№2 Издание основано в 1995 г. inf.1september.ru
УЧЕБНО-МЕТОДИЧЕСКИЙ ЖУРНАЛ ДЛЯ УЧИТЕЛЕЙ ИНФОРМАТИКИ

ИНФОРМАТИКА

4
Н.Н. Самылкина:
“...тема для меня является индикатором чувства юмора преподавателя: иначе о технике безопасности разговаривать невозможно”.

6
А.Г. Гейн:
“Мы будем учить информатике, т.е. умению работать с информацией. Например, умению обрабатывать информацию с помощью электронных таблиц, а не работе с электронными таблицами. Почувствуйте разницу!”

11
И.А. Калинин:
“Чтоб им всем монитор в кактусах поставили!”.

13
Д.П. Кириенко:
“...один из учебников начинается с главы “Моделирование как метод познания”. Вот этих слов я никогда не понимал”.

16
К.Ю. Поляков:
“Услышав выражения типа “владение умением понимать”, хочется отойти в сторону”.

20
И.Г. Семакин:
“Переход от “Арагац” к “БЭСМ-6” ощущался мной как прорыв в космос!”

22
А.Ю. Уваров:
“...пора вернуться к тому, что учитель — это Образованный Человек с большой буквы”.

электронная версия журнала
дополнительные материалы в Личном кабинете на сайте www.1september.ru

издательский дом 1september.ru
Первое сентября
февраль 2014
ИНФОРМАТИКА Подписка на сайте www.1september.ru или по каталогу «Почта России»: 79066 — бумажная версия, 12684 — CD-версия

НА ОБЛОЖКЕ

В этом году исполняется 50 лет легендарному семейству компьютеров — IBM 360. Сколько же всего связано с 360-ми! Несколько поколений программистов знали наизусть книжку Джермейна, отечественный вариант 360-х — ЕС ЭВМ на протяжении десятилетий были основной вычислительной платформой соцлагеря.

Но обложка не про это. Видите эту красную “кнопку” (это не кнопка — на нее не нажимали, ее вытягивали)? Это — вариант кнопки Reset у будущих персональных компьютеров. Вот как это было.

В НОМЕРЕ

3 ПАРА СЛОВ
- Что это будет?

4 ИНФОРМАТИКА В ЛИЦАХ
- Н.Н. Самылкина
- А.Г. Гейн
- И.А. Калинин
- Д.П. Кириенко
- К.Ю. Поляков
- И.Г. Семакин
- А.Ю. Уваров

28 УГЛУБЛЕНКА
- MongoDB: шаг от баз данных к базам знаний

46 ЗАНИМАТЕЛЬНЫЕ МАТЕРИАЛЫ ДЛЯ ПЫТЛИВЫХ УЧЕНИКОВ И ИХ ТАЛАНТЛИВЫХ УЧИТЕЛЕЙ
- “В мир информатики” № 195

В ЛИЧНОМ КАБИНЕТЕ

Облачные технологии от Издательского дома “Первое сентября”

Уважаемые подписчики бумажной версии журнала!

Дополнительные материалы к номеру и электронная версия журнала находятся в вашем Личном кабинете на сайте www.1september.ru.

Для доступа к материалам воспользуйтесь, пожалуйста, кодом доступа, вложенным в январский номер.

Срок действия кода: с 1 января по 30 июня 2014 года.

Для активации кода:
- зайдите на сайт www.1september.ru;
- откройте Личный кабинет (создайте, если у вас его еще нет);
- введите код доступа и выберите свое издание.

Справки: podpiska@1september.ru или через службу поддержки на портале “Первого сентября”.

ЭЛЕКТРОННЫЕ МАТЕРИАЛЫ

- Исходные файлы к статье “MongoDB: шаг от баз данных к базам знаний”.
- Презентации к статьям номера.

ИНФОРМАТИКА

ПОДПИСНЫЕ ИНДЕКСЫ
по каталогу “Почта России”: 79066 — бумажная версия, 12684 — электронная версия

http://inf.1september.ru

Учебно-методический журнал для учителей информатики
Основан в 1995 г.
Выходит один раз в месяц

РЕДАКЦИЯ:
гл. редактор С.Л. Островский
редакторы
Е.В. Андреева,
Д.М. Златопольский (редактор вкладки “В мир информатики”)
Дизайн макета И.Е. Лукьянов
верстка Н.И. Пронская
корректор Е.Л. Володина
секретарь Н.П. Медведева
Фото: фотобанк Shutterstock
Журнал распространяется по подписке
Цена свободная
Тираж 25 955 экз.
Тел. редакции: (499) 249-48-96
E-mail: inf@1september.ru
http://inf.1september.ru

ИЗДАТЕЛЬСКИЙ ДОМ “ПЕРВОЕ СЕНТЯБРЯ”
Главный редактор: Артем Соловейчик (генеральный директор)
Коммерческая деятельность: Константин Шмарковский (финансовый директор)
Развитие, IT и координация проектов: Сергей Островский (исполнительный директор)
Реклама, конференции и техническое обеспечение Издательского дома: Павел Кузнецов
Производство: Станислав Савельев
Административно-хозяйственное обеспечение: Андрей Ушков
Педагогический университет: Валерия Арсланьян (ректор)

ГАЗЕТА ИЗДАТЕЛЬСКОГО ДОМА
Первое сентября – Е.Бирюкова

ЖУРНАЛЫ ИЗДАТЕЛЬСКОГО ДОМА
Английский язык – А.Громушкина
Библиотека в школе – О.Громова
Биология – Н.Иванова
География – О.Коротова
Дошкольное образование – Д.Тюттерин
Здоровье детей – Н.Сёмина
Информатика – С.Островский
Искусство – О.Волкова
История – А.Савельев
Классное руководство и воспитание школьников – М.Битянова
Литература – С.Волков
Математика – Л.Рослова
Начальная школа – М.Соловейчик
Немецкий язык – М.Бузоева
ОБЖ – А.Митрофанов
Русский язык – Л.Гончар
Спорт в школе – О.Леонтьева
Технология – А.Митрофанов
Управление школой – Е.Рачевский
Физика – Н.Козлова
Французский язык – Г.Чесновицкая
Химия – О.Блохина
Школа для родителей – Д.Тюттерин
Школьный психолог – М.Чибисова

УЧРЕДИТЕЛЬ:
ООО “ЧИСТЫЕ ПРУДЫ”
Зарегистрировано ПИ № ФС77-44341 от 22.03.2011 в Министерстве РФ по делам печати
Подписано в печать: по графику 15.01.2014, фактически 15.01.2014
Заказ №
Отпечатано в ОАО “Первая Образцовая типография” Филиал “Чеховский Печатный Двор”
ул. Полиграфистов, д. 1, Московская область, г. Чехов, 142300
Сайт: www.chpd.ru
E-mail: sales@chpk.ru
Факс: 8 (495) 988-63-76

АДРЕС ИЗДАТЕЛЯ:
ул. Киевская, д. 24, Москва, 121165
Тел./факс: (499) 249-31-38
Отдел рекламы: (499) 249-98-70
http://1september.ru

ИЗДАТЕЛЬСКАЯ ПОДПИСКА:
Телефон: (499) 249-47-58
E-mail: podpiska@1september.ru

Что это будет?

В школьной информатике работает множество замечательных людей. Личное знакомство с ними — самая значимая награда для меня, как главного редактора профессионального издания

С этой идеей я носился давно. Первая запись в блокнотике “Идеи”: “Сделать номер “Информатики” про людей” — у меня появилась года три назад. На уровне ощущений я понимал, чего именно мне хочется: представить на страницах людей, которые связаны с преподаванием информатики в России. Интересных. Увлеченных. Увлекающихся. Толерантных. Категоричных. Дипломатичных. Осторожных. Резких. Сомневающихся. Мудрых. Разных.

В школьной информатике работает множество замечательных людей. Личное знакомство с ними — самая значимая награда для меня, как главного редактора профессионального издания. И мне очень хочется поделиться с читателями радостью этого человеческого общения.

Я помню момент, когда упомянутая выше страничка блокнотика была обновлена. Это было на одном из семинаров по характерологии, когда ведущий продемонстрировал простой, но эффектный и наглядный эксперимент. (В аудитории находились психологи, но все равно эксперимент получился ярким и запоминающимся.) Были предъявлены два женских портрета — Модильяни и Тропинина. Задание было примерно таким: поднимите, пожалуйста, руки те, кому более по душе картина Модильяни… пожалуйста, все “сфотографируйте” для себя эти поднятые руки… а теперь, пожалуйста, поднимите руки те, кому ближе картина Тропинина… и снова все “сфотографируйте”… Никаких прямых характерологических выводов из этого эксперимента не следует… просто отложите для себя две эти “фотографии” как иллюстрацию того, что люди *разные*.

Возможно, я еще долго придумывал бы, как мне сделать этот материал про *разных* людей, если бы не помог… Facebook. В конце прошлого года один из моих друзей (а я стараюсь придерживаться правила, что друзья в Facebook — это, как минимум, хорошие знакомые в жизни) выложил ссылку на “Руководство для начинающих”. Брошюрка имела красивую обложку и… ровно одну страницу внутри, на которой было написано… всего одно слово: “начните!”. Эта милая шутка стала последним “волшебным пинком”. Я составил небольшой перечень вопросов и разослал его коллегам с просьбой ответить — длинно ли, коротко ли, на все или на некоторые, живо или сухо, в общем — как душа захочет.

В этом номере мы представляем первые результаты этой задумки. Публикации будут продолжены и в следующих номерах. Сразу отвечу на вопрос про очередность и последовательность публикаций. Правило простое и чисто техническое — материалы ставятся в номера по мере готовности. Внутри номера используется алфавитный порядок, но дамы — вперед ☺.

Надеюсь, вам понравится!

С.Л. Островский,
главный редактор
(*so@1september.ru*)

Самылкина Надежда Николаевна

Профессор кафедры теории и методики обучения информатике математического факультета Московского педагогического государственного университета, кандидат педагогических наук, доцент ПО кафедры информатики и прикладной математики. Автор учебников, учебных пособий и плакатов для учащихся общеобразовательных школ и студентов педагогических вузов. Область научных интересов: методика преподавания информатики в школе и вузе, методика оценивания знаний учащихся, разработка и использование тестов в педагогической практике.

Как Вы пришли в информатику? Первый компьютер, на котором Вы работали? Первый язык программирования, с которым Вы познакомились? Первые задачи, которые Вы решали?

Можно сказать, что она сама ко мне пришла ☺ в лице директора школы, который в первых числах сентября 1985 года уговорил вести новый предмет ОИВТ в 9-х и 10-х классах. В этом году я как раз закончила школу и поступила в университет на новое направление — “Организация машинной обработки экономической информации”. Никто не хотел брать новый предмет, а мне “по младости лет” все новое давалось легко. К середине учебного года мы получили новые учебники, а к концу учебного года — новые компьютеры. Это действительно была новинка — компьютер на базе процессора Z-80, назывался ZX-Spectrum с цветным телевизором Шилялис. Но к тому времени мы уже успели поработать на японских Ямахах, которые были в компьютерных классах университета. На тот момент времени версии Бейсика были встроенными, использовались всеми для учебных целей. Там же поработали и на компьютерах ЕС, но использовали Фортран. Первыми серьезными задачами считаю те, что решались в курсовой (“Применение ПК для расчета углов разлета вторичных частиц”) и дипломной работе (“Центральное столкновение ядер He4 с ядрами фотоэмульсии при импульсе 1,6 ГЭВ/с”). По сути, это серьезные задачи моделирования, которые на тот момент решались средствами программирования. Надо сказать, достойно выглядят и сейчас (нашла старые распечатки), поскольку были подключены все имеющиеся графические возможности компьютеров. Из тематики задач понятно, что экономические задачи были скучноваты и я заканчивала физический факультет того же университета.

Первые пять фамилий из мира информатики, которые приходят Вам в голову? Если можете, пожалуйста, коротко расскажите, почему именно они.

Понятие “мир информатики” для меня не вполне определено, я не могу выделить границ. Все аспекты профессиональной, социальной и личной жизни связаны с информатикой. Могу ограничиться и тремя именами. Восхищают идеи Марвина Ли Минского (это о великих!), благодарна Александру Станиславовичу Лесневскому за уроки мудрости (это о коллегах), горжусь старшим сыном Самылкиным Александром — он программист в компании Microsoft.

Какая компьютерная техника и IT-сервисы окружают Вас сейчас? Насколько Вы “компьютерный человек” в обычной жизни? Это вопрос про все — про компьютеры и гаджеты, которые Вас окружают, про Ваш личный стиль социальной коммуникации в сети — насколько Вы живете в социальных сетях, насколько не можете жить без электронной почты и других средств коммуникации. Насколько бережете или не бережете свое личное пространство и время.

Трудно отказаться от удобств, которые предоставляют различные гаджеты, у меня есть все, что нужно для жизни и работы. “Умная” машина не только средство передвижения, когда большую часть времени проводишь за рулем. В машине тоже доступны все средства коммуникации — это то, что скорее бережет личное время. В соцсетях меня нет — этот круг общения для меня слишком велик, там трудно ограничить свое пространство, и это занимает слишком много личного времени.

Марвин Ли Минский

Если бы в Вашем распоряжении была лишь одна лекция (допустим — “пара”, полтора часа) и полная свобода рассказывать о том, что Вам интересно в информатике, чему бы Вы посвятили это время?

Ликвидации правовой безграмотности в информационной области. Этот разговор обычно возвращает студентам способность анализировать собственные поступки, а главное — их последствия.

Есть ли какой-то факт из информатики — может быть, какой-то алгоритм, — который произвел на Вас сильное, запоминающееся впечатление?

Обычно я довольно спокойно отношусь и к фактам, и к аргументам. Сейчас трудно выделить что-то одно, слишком много было фактов и оригинальных подходов к решению задач.

Есть ли у Вас любимая задачка или задачки?

До сих пор частенько возвращаюсь к задачнику Рэймонда Смаллиана “Принцесса или тигр?”.

Какую тему/темы школьного курса (будем считать, что мы находимся в пространстве ФГОС) Вы не хотели бы рассказывать детям? Ну не нравится она Вам.

Все годы эта тема для меня является индикатором чувства юмора преподавателя: иначе о технике безопасности разговаривать невозможно.

Надо ли сейчас вообще учить предмету “информатика” в школе? Ведь навыки работы на компьютерах, в сетях, с прикладным ПО дети получают и без этого и гораздо раньше, чем начинают изучать предмет. А специалистов вполне можно готовить в вузах. В школе-то зачем?

Думаю, что надо учить, но серьезнее отнестись к содержанию предмета и его возможностям. Многовато стало абсурда. Мне очень повезло, всегда давали возможность работать как мне хотелось. Информатика в школе — это мировоззрение, умный и пытливый взгляд на окружающий мир, поиск своего места в этом мире. Это начинается очень рано. К сожалению, бывает, что дети обходятся здесь без учителя — единомышленника.

Какой Вы представляете себе информатику в школе через пять лет? А через десять лет?

Современной. Именно об этом наши учебники информатики углубленного уровня.

Какой вопрос Вы хотели бы задать себе, чтобы на него ответить?

Пожалуй, что нет такого.

Гейн Александр Георгиевич

Руководитель авторского коллектива по разработке школьных учебников по информатике, профессор Уральского федерального университета, доктор педагогических наук.
Прямая речь: «Однажды я прочитал замечательную фразу: “От учителя, как и от врача, ждут чуда, но никто не удивляется, когда оно происходит”. И чем дольше я преподаю, тем ближе мне кажутся профессии учителя и врача. И тем чаще я повторяю себе: “Не навреди”».

Как Вы пришли в информатику? Первый компьютер, на котором Вы работали? Первый язык программирования, с которым Вы познакомились? Первые задачи, которые Вы решали?

Вопрос непростой, потому что я слишком давно живу на этом свете. Нет, это не жалоба на склероз — с памятью у меня все в порядке. Не совсем ясно, что назвать приходом в информатику.

Само слово “информатика” для меня появилось, по-видимому, с перспективой введения в школу курса ОИВТ. Но с алгоритмизацией, вычислительными методами и программированием я познакомился еще будучи учеником 9-го класса — проходя производственное обучение в вычислительном центре Уральского политехнического института. В то время один день в неделю отводился для так называемого производственного обучения, которое организовывалось на предприятии, шефствующем над данной школой. У нашей школы шефом был завод коммунального машиностроения. И обучение мы должны были получать в роли токарей и фрезеровщиков. Я с уважением отношусь к этим профессиям, но уже в то время понимал, что ни тем, ни другим скорее всего не стану — у меня вообще всегда было плохо с ручным трудом. Поскольку накануне я вернулся призером Всесоюзной олимпиады по математике, то, набравшись смелости, пошел к директору школы Зинаиде Калиновне Маковкиной и попросил найти мне более подходящий вид производственного обучения. И она, пользуясь своим статусом старого большевика, договорилась, что меня и еще двух моих одноклассников (один из них А.А. Романюха — ныне замдиректора по науке Института вычислительной математики РАН, д. физ.-мат. н.) направили в ВЦ физтеха УПИ.

Конечно, о программировании и профессии программиста я слышал и до этого, так что данный вид деятельности для меня не был открытием, и я точно понимал, что программистом не буду. Я любил (и люблю) решать математические задачи, но не любил, когда меня ограничивали в выборе средств для их решения. Вот один эпизод того времени. В 10-м классе к нам в школу пришел преподаватель математико-механического факультета Уральского госуниверситета агитировать на поступление. Я спросил, насколько сильно преподавание ориентировано на подготовку программистов. В ответ услышал, что для такой подготовки лучше поступать в МГУ на ВМК. “Нет, — сказал я, — в этом случае я предпочту поступать в УрГУ”. Это был 1972 год.

Поэтому мне кажется естественным считать, что в информатику я пришел в 1985 году. А пришел я со словами: “Я знаю, как написать учебник по информатике”. Дело в том, что Свердловская область благодаря усилиям академика Николая Николаевича Красовского получила право преподавать информатику в компьютерном варианте, в то время когда вся страна начинала учить информатику без компьютеров. Промышленные предприятия Свердловской области выделили миллион инвалютных рублей на покупку компьютеров (были закуплены Ямахи и Роботроны). Но нужен был учебник для такого преподавания. Был объявлен конкурс и было создано четыре авторских коллектива. Первый коллектив, в который я обратился с данным заявлением, был в моем родном университете, но мне ответили, что они тоже знают, как писать учебник и в моих знаниях не нуждаются. Тогда я пошел в пединститут, где создавался аналогичный коллектив, и предложил свое участие им. Этот коллектив возглавлял Владимир Габриэлевич Житомирский, с ним и с остальными участниками я тоже был давно знаком (по меркам того моего возраста — мне тогда исполнилось 35), как выпускник одного и того же факультета. По существу, это был еще один университетский коллектив. Наш учебник выиграл этот конкурс, а спустя два года он получил гриф Министерства просвещения РСФСР (два других учебника информатики, имевших хождение в то время, созданные коллективами под руководством В.А. Каймина и А.Г. Кушниренко, имели гриф Министерства просвещения СССР), так что мы по праву можем считать себя авторами первого российского учебника информатики.

Первый компьютер, на котором я работал. Если называть компьютерами наши крупногабаритные ЭВМ, то первым компьютером была одноадресная машина “Урал-14” с перфолентным вводом. Первый персональный компьютер, на котором мне довелось работать, была “Ямаха”.

Первый язык программирования, с которым я познакомился, был Алгол. Первый язык программирования, на котором мне довелось реально работать, был Фортран. Но до этого мне примерно 7 лет довелось программировать в машинных кодах. Мне удавалось писать довольно эффективные программы, и поэтому на языки программирования, которые после трансляции раздували и без того громоздкие программы еще больше, я смотрел без удовольствия.

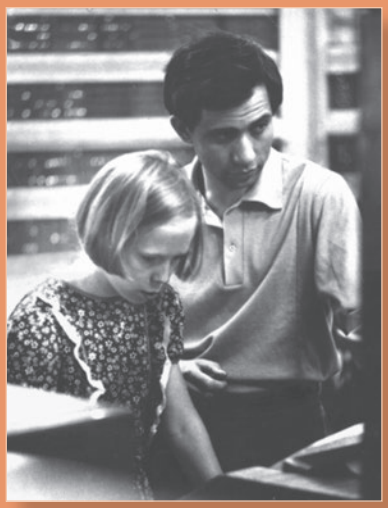
Первые задачи, которые мне довелось решать с помощью ЭВМ, — это были задачи статистической обработки экспериментальных данных. Мне это было интересно, поскольку меня привлекали не только как программиста (тогда еще не было статистических пакетов, объем памяти и быстродействие ЭВМ требовали достаточной виртуозности, чтобы расчеты закончились раньше, чем произойдет очередной сбой техники). В соответствующих научных группах я был если не единственным, то, по-видимому, наиболее опытным математиком, поэтому ответственность за выбор математического инструментария лежала на мне. И, надо думать, все было не так уж плохо — это не было моей профессиональной деятельностью, поскольку вся моя трудовая жизнь связана с преподаванием алгебраических дисциплин в родном университете. Меня просто приглашали порешать разные задачи — моделирование кинетики химических реакций, распространения загрязнений в атмосфере, восприятия музыкальных произведений слушателями и т.д. Сейчас, когда многие модели построены маститыми научными коллективами, все это выглядит, наверно, достаточно примитивно, но в начале 70-х все было новинкой. Остались научные статьи того времени, может быть, для кого-то они стали стартовой ступенькой. Возможно, поэтому мне особенно близки слова А.П. Ершова: “Как самостоятельная наука информатика вступает в права тогда, когда в рамках соответствующей частной науки строится информационная модель того или иного фрагмента действительности”.

Учебник 1991 г.

Первые пять фамилий из мира информатики, которые приходят Вам в голову? Если можете, пожалуйста, коротко расскажите, почему именно они.

Вот они в том порядке, в котором приходят в голову: Н.Винер, А.П. Ершов, Г.А. Звенигородский, С.Пейперт. Пятая фамилия мне пришла в голову не сразу. Зато потом пришли сразу две: К.Шеннон и Джон фон Нейман.

Приоритет Н.Винера понятен — знакомством с его работами по кибернетике началось мое вхождение в мир информатики. А.П. Ершов поражает глубиной, разносторонностью и прогностичностью своих трудов, Г.А. Звенигородский сыграл значительную роль в формировании моего понимания методики изучения информатики. Основные идеи С.Пейперта о связи развития мышления с алгоритмической организацией учебной деятельности явились выражением того, что весьма нечетко было в моем понимании этих вопросов. Иными словами, это те люди, которые оказали значительное мировоззренческое влияние. А К.Шеннон и Джон фон Нейман, конечно, великие ученые, и вклад их в развитие информатики нельзя преуменьшать, но это всего лишь знания. Принципиально важные и даже корректировавшие какие-то ментальные установки, но не более.



Геннадий Анатольевич Звенигородский

Какая компьютерная техника и IT-сервисы окружают Вас сейчас? Насколько Вы "компьютерный человек" в обычной жизни? Это вопрос про все — про компьютеры и гаджеты, которые Вас окружают, про Ваш личный стиль социальной коммуникации в сети — насколько Вы живете в социальных сетях, насколько не можете жить без электронной почты и других средств коммуникации. Насколько бережете или не бережете свое личное пространство и время.

Ноутбуку я предпочитаю стационарный компьютер. Конечно, есть сотовый телефон, но без наворотов. Вот и все. В сети я присутствую в основном электронной почтой, хотя есть скайп, аська и зарегистрирован в Фейсбуке. На обсуждения в Фейсбуке откликаюсь крайне редко, только на какие-то очень принципиальные вопросы. Впрочем, и на них я откликаюсь далеко не всегда, потому что моя точка зрения, как закоренелого информатика, не просто оказывается в меньшинстве, а в противоречии практически со всеми высказываемыми точками зрения. Иногда хочется, как в финале фильма "Иван Васильевич меняет профессию", сказать: "И тебя вылечат, и тебя вылечат, и меня, наверно, тоже вылечат" (цитата, как вы понимаете, специально не дословная).

Еще есть личная страничка на кафедральном сайте, в основном для общения со студентами, но есть там и раздел, посвященный школьной информатике с различными (но немногочисленными) методическими материалами.

Без электронной почты моя жизнь невозможна, с нее начинается утро, и потом довольно регулярно она проверяется в течение дня. Исключения составляют периоды, когда я бываю в отъезде, поскольку ноутбук с собой не вожу. А на телефоне мне пользоваться Интернетом не очень удобно.

Интернет тоже жизненно важен. Прежде всего информационной составляющей — бытовой, социальной, научной. В своей научной математической деятельности пользуюсь системами символьных вычислений, которые лежат где-то там, в "облаках".

Личное время практически совсем не берегу (за что получаю выговоры от тех, кто хотел бы получить его от меня побольше), а вот к личному пространству отношусь весьма трепетно.

Если бы в Вашем распоряжении была лишь одна лекция (допустим — "пара", полтора часа) и полная свобода рассказывать о том, что Вам интересно в информатике, чему бы Вы посвятили это время?

Вопрос оказался для меня очень непростым. Потому что не могу представить себе абстрактную аудиторию, которой читаю лекцию. А каждой конкретной аудитории я читал бы только ей предназначенную лекцию. Школьникам-гуманитариям — о применении информатики в исследовании литературного и музыкального творчества. Школьникам, склонным к естественным наукам, — о

моделировании случайных процессов. Школьникам математико-информационного направления — о доказательном программировании. Студентам-математикам — об алгоритмически неразрешимых задачах или о “ленивых” вычислениях. Учителям-математикам — об экспериментальной компьютерной математике.

Есть ли какой-то факт из информатики — может быть, какой-то алгоритм, — который произвел на Вас сильное, запоминающееся впечатление?

Такого факта нет. Возможно, сказывается моя привязка к математике. На мой взгляд, информатика — не фундаментальная, а прикладная наука. Так же, как технические науки опираются на теории и факты физики и химии, а не открывают их сами, информатика опирается на математику. Академик Н.Н. Красовский как-то сказал: “Нет информатики без математики”. Я понимаю, что в Российской академии наук есть отделение информатики, то есть она признана академической наукой. Но ведь все алгоритмы, начиная с алгоритма Евклида, придуманы математиками. Существование алгоритмически неразрешимых задач — это тоже математика. Шифрование с открытым ключом, которое сейчас является основой телекоммуникации, — это математическая теория. Построение кодов, исправляющих ошибки, — это красивая, но тоже математическая теория. Теоремы Шеннона и Котельникова о системах передачи информации — это математические теоремы. К информатике можно отнести создание языков программирования, но их создание снова базируется на математических или логико-математических результатах. Скажем, великолепные теорема Черча — Россера и комбинатор Карри, лежащие в основе функциональных языков программирования и ленивых вычислений, метод резолюций для узкого исчисления предикатов, лежащий в основе языка Пролог, теория формальных грамматик, лежащая в основе императивных языков программирования. Если же говорить о самой компьютерной технике, то там в основе опять физика.

Есть ли у Вас любимая задачка или задачки?

На этот вопрос отвечу как формальный исполнитель — есть.

Какую тему/темы школьного курса (будем считать, что мы находимся в пространстве ФГОС) Вы не хотели бы рассказывать детям? Ну не нравится она Вам.

Интернет (и это не связано с ФГОС).

Надо ли сейчас вообще учить предмету “информатика” в школе? Ведь навыки работы на компьютерах, в сетях, с прикладным ПО дети получают и без этого и гораздо раньше, чем начинают изучать предмет. А специалистов вполне можно готовить в вузах. В школе-то зачем?

Информатическая (я не сказал информационная) безграмотность обходится очень дорого. У меня довольно большая коллекция примеров такого негатива. И она пронизывает нашу систему снизу доверху — от менеджера магазина до депутата Государственной думы. Но, как и всякое массовое явление, оно воспринимается как должное, и большинству даже в голову не приходит, что с этим надо бороться. Впрочем, с информационной безграмотностью у нас обстоит примерно так же. Просто учить надо информатике, а не тому, что в этом вопросе объявлено информатикой.

Какой Вы представляете себе информатику в школе через пять лет? А через десять лет?

Никогда не ощущал себя провидцем (уж не говорю, пророком). Я уже говорил, что меня восхищает прогностика А.П. Ершова. В момент, когда информатика вводилась в школьное образование под флагом развития алгоритмического мышления и овладения компьютерной грамотностью, предвидеть ее поворот в сторону информационных технологий — это гениально. Мой соавтор по первым учебникам информатики, Марк Валентинович Сапир, в середине 90-х предсказывал, что в начале двухтысячных весь курс информатики будет ориентирован на изучение Интернета. Это предсказание не сбылось — Интернет занимает довольно скромное место в нынешнем курсе и вряд ли расширит свое присутствие, поскольку он вполне осваивается детьми в бытовом су-

ществовании. А обучение профессионально ориентированной работе в Интернете не является целью общеобразовательного курса. Я думаю, что за пять лет в школьной информатике изменения будут почти незаметны — слишком маленький срок. Во-первых, учебники, по которым будут через пять лет учиться школьники, пишутся сегодня. А в них изменений не заметно. Хотя бы уже потому, что утвержденная на сегодняшний день примерная программа, которой должны придерживаться авторы учебников, если хотят, чтобы их учебники получили гриф МОиН, мало отличается от обязательного минимума предшествующего госстандарта. Во-вторых, образование вообще весьма консервативная и медленно меняющаяся система. Когда в стране в середине 80-х разразился экономический кризис, а затем и политический, приведший к распаду СССР, система образования функционировала еще вполне пристойно, хотя востребованность образования была почти нулевая. А с конца 90-х начался развал системы образования как следствие кризиса общественного производства. Меня можно обвинять в марксистских взглядах, но я не знаю ни одного примера, опровергающего тезис, что базис определяет надстройку, а образование — это надстроечная структура (это вовсе не отрицает информатически очевидного тезиса, что надстройка по каналам обратной связи может влиять на базис). Принятие в 2004 году Федерального компонента государственного образовательного стандарта по информатике дало старт формированию кризисной ситуации в преподавании школьной информатики. Я писал об этом по крайней мере в трех статьях, правда, в журналах, которые школьным учителям вряд ли известны. Сейчас вроде бы производство восстанавливается (хотя единой точки зрения на этот вопрос нет даже в Государственной думе). Но ломать — не строить. Когда начнется восстановление системы образования и как оно будет происходить — кто ж знает? Мы видим нарывы на теле образования, например, в виде ЕГЭ, и начинаем хором обсуждать, что лучше — помазать мазью (т.е. как-то его улучшить) или хирургически удалить (т.е. отказаться от него вообще). А болезнь мы лечить не собираемся. А ЕГЭ тут почти ни при чем — подобный эффект мы имели бы при любой форме итогового контроля. Может быть, острота была бы чуть меньше. Ясно одно — за ближайшие пять лет система образования не восстановится. А значит, и информатика не изменится. За это время в лучшем случае станут ясными позитивные тенденции, если, конечно, таковые будут.

Десять лет — срок, когда уже можно ждать изменений. Вопрос в том, в каком направлении они будут происходить. Вполне вероятно, что деградация будет продолжаться. Как сказал недавно мой одноклассник, а ныне коллега и тоже профессор, только на другой кафедре, в результате очередной революции на поверхности снова оказались Шариковы с вытекающими отсюда последствиями. Я не разделяю такого пессимизма относительно последствий. Например, надеюсь, что вопрос о том, как надо писать то или иное слово, не будет решаться голосованием, исходя из пролетарского самосознания, но равнодушие к грамотному выражению своих мыслей у большинства не только студентов, но и кандидатов наук — подумаешь, неправильно написал слово, вы же меня поняли — удручает…

Но давайте верить в лучшее. Мы будем учить информатике. т.е. умению работать с информацией. Например, умению обрабатывать информацию с помощью электронных таблиц, а не работе с электронными таблицами. Почувствуйте разницу! А сегодня в нашем ФГОС информационную компетентность, которая, по мнению всех ведущих специалистов в области образования, является ключевой компетенцией выпускника, подменили ИКТ-компетентностью, т.е., как написано там же в расшифровке этого понятия, умением применять технологии.

Какой вопрос Вы хотели бы задать себе, чтобы на него ответить?

Чего ждет от нас школьная Информатика? Именно так — Информатика с большой буквы.

Калинин Илья Александрович

Кандидат педагогических наук, доцент кафедры информатики и прикладной математики Института математики и информатики Московского городского педагогического университета, начальник отдела серверной и телекоммуникационной инфраструктуры дирекции информационных систем Московского городского педагогического университета. Автор учебников, учебных пособий и плакатов для учащихся общеобразовательных школ. Область научных интересов: телекоммуникационные технологии, информационный поиск.

Как Вы пришли в информатику?

Я до сих пор не очень понимаю, что такое “информатика”, — из 25 лет работы это понимание только больше расплылось. На самом деле я пришел именно “к компьютеру”, как к какой-то потрясающе умной штуке, которая “почти все может”. Спасибо фантастике: “Press Enter”, “Витки”, “Встреча” — список длинный.

Первый компьютер, на котором Вы работали?

“Электроника МС-0511”, УКНЦ.

Первый язык программирования, с которым Вы познакомились?

Basic. Другого у нас не было, и только лет через пять мне удалось найти внятные компиляторы Pascal и C.

Первые задачи, которые Вы решали?

Если исключить учебные задачи типа “Нарисуйте робота”, то подготовка программы-теста по биологии. А потом — обработка статистических данных для детской научной работы (своей).

Первые пять фамилий из мира информатики, которые приходят Вам в голову? Если можете, пожалуйста, коротко расскажите, почему именно они.

Что такое “мир информатики”?.. Программы, интерфейсы, технология, применение?..

Перечисляя в произвольном порядке: Кнут, Гейтс, Джобс, Ершов, Тьюринг.

Наверное, это те люди, которые тем или иным способом стали “маркерами”, титульными лицами того, с чем я столкнулся в своей работе. Точно могу сказать только про двоих: Дональд Кнут, первый том “Искусства программирования для ЭВМ” — первая книга, которая перевела для меня алгоритмы из категории “магические пассы, которые надо выучить” в “сложный механизм, который строится таким образом...”, Тьюринг — “Однажды им стало известно, что на далеком острове за океаном живет ученый по имени Тьюринг, умеющий создавать автоматы” (Станислав Лем. “Магелланово облако”).

Какая компьютерная техника и IT-сервисы окружают Вас сейчас? Насколько Вы “компьютерный человек” в обычной жизни?

Пожалуй, почти весь спектр — от встроенных и носимых устройств до Blade-шасси, кластеров и СХД. “Тяжелыми” системами и специализированными вычислительными системами только не занимаюсь. У меня практически вся работа связана с компьютерами — так что, наверное, сильно компьютерный.

В социальных сетях почти не живу. Многое умею, но предпочитаю оставаться наблюдателем. Прожить без средств коммуникации — настолько, насколько могу без них работать. Сейчас с почтой, сотовой связью и сетью

связана вся моя работа. Если она изменится — что ж, попробую обойтись ☺. Я не разучился считать на бумажке (и даже пользоваться таблицами).

Если бы в Вашем распоряжении была лишь одна лекция (допустим — "пара", полтора часа) и полная свобода рассказывать о том, что Вам интересно в информатике, чему бы Вы посвятили это время?

О "поисках смысла". Почти "Грааль" всей области деятельности — создание искусственного интеллекта. О том, что же такое "думать"...

Есть ли какой-то факт из информатики — может быть, какой-то алгоритм, — который произвел на Вас сильное, запоминающееся впечатление?

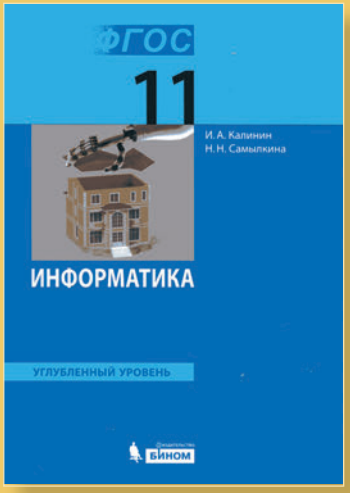
Есть несколько, самый забавный — алгоритм аль-Каши возведения в степень. Наверное, потому, что удалось его самому придумать — а уже потом выяснить, что он используется с XV века. А алгоритм эффектный — возведение, например, в степень 65 535-ю степень всего за 32 умножения.

Есть ли у Вас любимая задачка или задачки?

Как-то я не вступаю в личные отношения с задачами... На самом деле, не связанная с информатикой: "Арбуз весит два килограмма, да еще столько, сколько весит пол-арбуза. Сколько весит арбуз?" Так вот — не три и не пять.

Какую тему/темы школьного курса (будем считать, что мы находимся в пространстве ФГОС) Вы не хотели бы рассказывать детям? Ну не нравится она Вам.

Технику безопасности и санитарные нормы. Стартовое занятие каждого года, в котором исходный смысл полностью убит обязательно предписанным содержанием, эталонным по своей беспредметности, и утерей связи с действительностью. Ненавижу работу ради "постановки галочек" — как насчет ведра с песком в кабинете информатики (ЛИЧНО У МЕНЯ ТРЕБОВАЛИ — это не придумано!)? Или замера расстояния от глаз до экрана? А утверждение, что компьютер — опасное электрическое устройство, которое предлагается сказать современным детям? Страницы требований — устаревших еще до их издания, в которые предлагается верить, как в Бога (это при том, что я вообще агностик). И все это нам "впаривают" люди, за последние 20 лет не опубликовавшие для педагогов НИ ОДНОГО доказательного исследования с двойным слепым контролем, что соблюдение всех этих требований ***вообще на что-то влияет***. НИ ОДНОГО обоснованного, взвешенного курса. Вполне открыто говорящие, что не провели исследований по собственной методике даже для LCD-экранов и все нормы рассчитывали для кинескопных телевизоров... Ханжество, фарисейство и безграмотность. Чтоб им всем монитор в кактусах поставили!



Учебник 2013 г.

Надо ли сейчас вообще учить предмету "информатика" в школе?

Да.

Ведь навыки работы на компьютерах, в сетях, с прикладным ПО дети получают и без этого и гораздо раньше, чем начинают изучать предмет. А специалистов вполне можно готовить в вузах. В школе-то зачем?

Теоретически — можно. На практике — подавляющее большинство серьезных специалистов занимаются предметом лет с 10 — а то и раньше, вам это ни о чем не говорит?

В том-то и дело, что этот предмет не столько навыки, сколько образ действий, определяемых знаниями и опытом. Навыки "набить" можно — но очень хочется, чтобы информатику перестали воспринимать как набор навыков.

Какой Вы представляете себе информатику в школе через пять лет? А через десять лет?

"Хочешь рассмешить Бога — расскажи ему о своих планах". Футуристические прогнозы — крайне неблагодарная штука.

Представьте себе реакцию себя же 20-летней давности на фразу "Я пока еду, телефоном расписание на сегодня сфотографируй и мне на почту скинь"! Как же предсказывать, что станет важным через десять лет?

Мне кажется, как предмет информатика не очень меняется. Опишу впечатление: очень хотелось бы видеть ее предметом, который любят те, кто его преподает и учит.

Какой вопрос Вы хотели бы задать себе, чтобы на него ответить?
Почему вы вообще занимаетесь школьной информатикой?

Не считая того, что "это уже просто привычка" (старшее поколение, думаю, узнает довоенную еще шутку), — потому, что мне до сих пор хочется, чтобы люди вместе со мной удивлялись тому, что мы делаем с "умными машинами", а потом удивляли других.

Кириенко Денис Павлович

В школе увлекался математикой, любил участвовать в различных олимпиадах, был призером всероссийской олимпиады по математике. Педагогикой интересовался с самого окончания школы — преподавал в Тульской летней многопрофильной школе (*lmsh.edu.ru*), вел кружки, участвовал в проведении математических олимпиад. Окончил мехмат МГУ, занимался научной работой, программированием, но в итоге победило желание работать в школе. В 179-ю школу г. Москвы пришел в 2002 году практически случайно — куда раньше позвали. Некогда одна из самых известных в Москве математических школ в 1980-е годы была "развалена" до уровня средней районной школы, но за десять лет коллективу учителей-энтузиастов удалось вновь вернуть школу в число одних из первых в России не только в математике, но и в информатике. Затем втянулся в проведение олимпиад, но уже по информатике, вошел в число разработчиков ЕГЭ по информатике, разработал формат проведения ГИА по информатике, стал ездить преподавателем в Летнюю компьютерную школу (*lksh.ru*) — лучший летний лагерь России для юных программистов. Член коллектива разработчиков ALT Linux Team.

Как Вы пришли в информатику? Первый компьютер, на котором Вы работали? Первый язык программирования, с которым Вы познакомились? Первые задачи, которые Вы решали?

Интерес к компьютерам у меня появился еще когда я учился в школе, в конце 1980-х годов. Компьютеры тогда я видел только на картинках, ну или, может быть, по телевизору, но интерес уже был. Однажды в книжном магазине я увидел книжку голландца Каспера Боона "Паскаль для всех". На обложке был нарисован компьютер, а пролистав книжку, я понял, что эта книга для тех, кто повелевает компьютерами. Книжка была куплена, я начал ее читать и все понимал, пока не дошел до записи действительных чисел с плавающей точкой. Для шести-семиклассника, никогда не занимавшегося программированием, это оказалась сложная тема, и книжка была отложена до лучших времен.

Первый компьютер, за которым я работал, вернее, играл, кажется, был "Агат" в игровом клубе. Это было около 1990 года. Помню ныне забытую игру Xonix — где нужно отрезать куски территории, убегая от движущихся точек.

В 1991 году информатика началась у меня в школе, я учился в 8-м классе. У нас были компьютеры "Корвет", программировали на Бейсике. Помню, что после двух часов работы за "Корветом" начинала жутко болеть голова, но, несмотря на это, оторвать меня от компьютера было очень тяжело. Писали простые программы — помню программу, которая рисовала на экране картинку при помощи операторов "нарисовать линию" и "закрасить область", а большая часть программирования заключалась в перерисовывании картинки на миллиметровую бумагу и вводе координат точек в тело программы.

В 9-м классе уже в другой школе программировали на Бейсике на БК-0010, а летом после 9-го класса последнюю неделю каникул я проводил дома. Только вернувшись из летней математической школы,

я горел желанием узнать что-то новое, и тут нашлась книжка Боона. За неделю, лежа на диване, я прочитал ее целиком и, придя в школу после каникул, сразу стал писать на Паскале, полюбив этот язык на следующие два-три года. В школе к тому времени нас "пустили" в класс к серьезным компьютерам: IBM PC AT 286, на которых были дисководы, что существенно упрощало процесс создания и сохранения программ, а программировать можно было на Turbo Pascal. Ну и игрушки были совсем другого уровня — SimCity, "Цивилизация"...

В 11-м классе я пробовал изучать C, но язык мне не понравился, наверное, слишком сложным показался. Заставили понять и полюбить языки C и C++ меня уже в университете, и, действительно, много лет после этого я писал исключительно на C и C++, пока не открыл для себя Perl, а затем и Python.

Первые пять фамилий из мира информатики, которые приходят Вам в голову? Если можете, пожалуйста, коротко расскажите, почему именно они.

Дональд Кнут — автор самой уважаемой (хотя и довольно редко используемой) монографии "Искусство программирования" и замечательной системы верстки LaTeX.

Ричард Столлман — за идеи Free Software и создание GNU.

Линус Торвальдс — за создание ядра Linux.

Бьёрн Страуструп — за язык C++.

Гвидо Ван Россум — за наиболее любимый в последнее время язык Python.

На самом деле пять фамилий — это очень мало. Хочется назвать создателей Unix и C Денниса Ритчи, Брайана Кернигана, Кена Томпсона, Роба Пайка, классиков Computer Science Алана Тьюринга, Клода Шеннона, Эдсгера Дейкстру, Тони Хоара, Никлауса Вирта и еще многих других людей.

Какая компьютерная техника и IT-сервисы окружают Вас сейчас? Насколько Вы "компьютерный человек" в обычной жизни? Это вопрос про все — про компьютеры и гаджеты, которые Вас окружают, про Ваш личный стиль социальной коммуникации в сети — насколько Вы живете в социальных сетях, насколько не можете жить без электронной почты и других средств коммуникации. Насколько бережете или не бережете свое личное пространство и время.

Я весьма продвинутый пользователь. Дома есть компьютер, ноутбук, всегда с собой — планшет, без планшета, электронной почты, Интернета чувствую себя неуютно. Все рабочие файлы, преимущественно, хранятся в "облаке", чтобы были везде и всегда под рукой. Но все-таки до крайностей дело не доходит — например, Twitter, Facebook и "Одноклассниками" я практически не пользуюсь, предпочитая из социальных сетей ВКонтакте — преимущественно, из-за сложившегося круга общения. Или, например, у меня нет смартфона — период смартфонов для меня прошел, на смену смартфону пришел планшет, с которым удобней работать, а телефон у меня сейчас простой и небольшой. Да и вообще я не люблю телефонные звонки, предпочитая электронную почту.

Если бы в Вашем распоряжении была лишь одна лекция (допустим — "пара", полтора часа) и полная свобода рассказывать о том, что Вам интересно в информатике, чему бы Вы посвятили это время?

Я бы рассказал про язык Python и показал бы, как легко и просто на нем писать разные алгоритмы, программы получаются раза в три короче, чем на Pascal или C++.

Есть ли какой-то факт из информатики — может быть, какой-то алгоритм, — который произвел на Вас сильное, запоминающееся впечатление?

Криптография с открытым ключом — любой человек может зашифровать сообщение так, что никто, кроме получателя, его не прочтет, и не нужно при этом предварительно обмениваться с собеседником секретными ключами. А также все связанные с этим вопросы — электронная подпись, протокол бросания монеты, доказательство с нулевым разглашением. Во время учебы в университете я даже немного занимался научной работой в этом направлении.

Есть ли у Вас любимая задачка или задачки?

Например, задача про "Банкомат". Есть банкноты заданных номиналов, как выдать заданную сумму при помощи минимального числа банк-

нот? Сначала в голову приходит “жадное” решение — выдаем банкноты самого крупного номинала (например, по 5000 рублей), пока это возможно, затем переходим к банкнотам меньшего номинала (по 1000 рублей), затем — к следующим по величине (500 рублей) и т.д. Этот алгоритм работает, когда все номиналы банкнот — кратные, например, 10, 50, 100, 500, 1000, 5000 рублей. Но если номиналы банкнот некратны — например, 1, 30, 100 рублей — и нужно выдать, например, 120 рублей, “жадный” алгоритм предложит одну банкноту в 100 рублей и 20 банкнот по одному рублю, а оптимальное решение — выдать четыре банкноты по 30 рублей. На примере этой задачи легко объясняется идея динамического программирования.

Какую тему/темы школьного курса (будем считать, что мы находимся в пространстве ФГОС) Вы не хотели бы рассказывать детям? Ну не нравится она Вам.

Есть такая тема — “Моделирование и формализация”. Например, один из учебников начинается с главы “Моделирование как метод познания”. Вот этих слов я никогда не понимал. Конечно, решение любой задачи на компьютере есть создание модели, и, всякий раз решая задачу, мы создаем модель. Но при этом я ни разу не думал, что вот я сейчас буду создавать модель, чтобы решить задачу. Примерно с таким же успехом можно будущим архитекторам рассказывать, как важно архитектору уметь рисовать чертежи, вместо того чтобы учить черчению.

Надо ли сейчас вообще учить предмету “информатика” в школе? Ведь навыки работы на компьютерах, в сетях, с прикладным ПО дети получают и без этого и гораздно раньше, чем начинают изучать предмет. А специалистов вполне можно готовить в вузах. В школе-то зачем?

Изучение прикладного ПО, несомненно, должно быть интегрировано с другими предметами. Но есть некоторая информационная культура (культура работы с информацией), которой не то чтобы преимущественно не учат, но даже непонятно, как учить. Например, элементом этой культуры является заполнение поле “Тема” в письме электронной почты, подпись в конце письма, сопроводительный текст к письму, если основные документы пересылаются в виде вложения. И, наоборот, если основным содержанием письма является некоторая текстовая информация, то писать ее нужно в теле письма, а не во вложении. К сожалению, даже многие учителя не владеют этой культурой; например, на просьбу “прислать сообщение по электронной почте об окончании тура олимпиады” зачастую приходят письма, к которым приложен документ MS Word, в котором крупным шрифтом написано: “Тур в школе номер... окончен!”. Хоть авторы такого послания и преподают информатику в школе, но, похоже, им не приходилось за сутки получать и обрабатывать сотню сообщений по электронной почте. Это все — не использование информационных технологий, а культура использования информационных технологий, и как этому учить — пока непонятно.

Что касается теоретической информатики и программирования — этот предмет, несомненно, обязателен для изучения в школе. Во-первых, чтобы современные пользователи понимали принципы работы информационных технологий, что компьютер управляется программами, которые создаются человеком, и как человек создает эти программы. Во-вторых, поскольку спрос на IT-специалистов постоянно растет, то еще в школе нужно развивать интерес к IT-технологиям. Чем больше школьников познакомится с IT в школе, тем больше из них пойдет в вузы на соответствующие специальности. Развивая IT-индустрию, государство может выйти на лидирующие позиции в мире в IT-технологиях, а для этого вовсе необязательно строить заводы или сверлить скважины, достаточно только развивать IT-образование в школах и вузах и предоставлять льготы IT-компаниям.

Какой Вы представляете себе информатику в школе через пять лет? А через десять лет?

Мне кажется, что в российской школе существенно ничего не изменится.

Какой вопрос Вы хотели бы задать себе, чтобы на него ответить?

Не могу придумать такой.

Поляков Константин Юрьевич

Родился в 1968 году. В 1991 году закончил с отличием Ленинградский кораблестроительный институт по специальности “Автоматизация теплоэнергетических процессов”. Доктор технических наук, специалист по теории цифровых систем автоматического управления. С 1991 года работает учителем информатики в средней школе № 163 Санкт-Петербурга. Автор проектов “Методизмы” (обсуждение существующих учебников информатики) и “Материалы для подготовки к ЕГЭ по информатике” на сайте *http://kpolyakov.spb.ru*. Разработчик бесплатного программного обеспечения для поддержки школьного курса информатики (тренажер “Логика”, система “Исполнители”, сетевой тест NetTest, веб-редактор HEFS, учебная модель компьютера “ЛамПанель” и др.). Почетный работник общего образования РФ. Руководитель авторского коллектива по разработке учебника информатики углубленного уровня для 10–11-х классов. В свободное время увлекается водными и конными туристскими походами по России.

Как Вы пришли в информатику? Первый компьютер, на котором Вы работали? Первый язык программирования, с которым Вы познакомились? Первые задачи, которые Вы решали?

Первое знакомство с компьютерами произошло на первом курсе Ленинградского кораблестроительного института. Нас учили программировать на языке FORTRAN IV в пакетном режиме. Это означает, что программист пишет программу на специальном бумажном бланке, затем сдает ее в вычислительный центр (ВЦ) на подготовку данных. Через день получает пачку перфокарт, сдает ее снова в ВЦ на выполнение, еще через день получает распечатку, из которой следует, что перфокарты набиты с ошибками, сдает ошибочные строчки на повторную набивку и т.д. Поскольку учебные задания вроде табулирования значений функции были довольно скучными, любимым развлечением студентов было вывести на печатающее устройство какой-нибудь рисунок из символов.

С начала второго курса меня взяли на работу лаборантом в ВЦ. Поскольку свою основную работу (протирать спиртом головки магнитофонов, мыть полы в машинном зале и т.п.) удавалось делать достаточно быстро, оставшееся время можно было “помучить машину” ЕС-1022, которая была тут же рядом. Настольная книга того времени — К.Джермейн “Программирование на IBM/360” (М.: “Мир”, 1971).

Потом были PL/1, Бейсик, ассемблеры для разных процессоров, Паскаль, C, C++, Фокал, Delphi, Matlab, Javascript, Python. В разные периоды немного писал на других языках (Java, Perl, PHP, Ruby), скорее ради развлечения. По сей день любимый язык программирования — C, для программ с графическим интерфейсом — Delphi, для того чтобы что-то быстро проверить — Python.

Первые пять фамилий из мира информатики, которые приходят Вам в голову? Если можете, пожалуйста, коротко расскажите, почему именно они.

Д.Кнут, Н.Вирт, Ч.Хоар, Б.Гейтс, С.Джобс.

Дональд Кнут — патриарх современной отрасли *computer science*. Его главный труд — “Искусство программирования” — поражает фундаментальностью, строгостью и глубиной. В то же время Кнут в отличие от многих наших университетских преподавателей не только знает, как грамотно писать программы, но и умеет это делать: разработанная им система TeX — лучшее, что на данный момент придумали для подготовки математической литературы к изданию.

Никлаус Вирт — разработчик Паскаля, Модулы, Оберона. Автор замечательной книжки “Алгоритмы + структуры данных = программы”, где просто и понятно описаны достаточно сложные вопросы структурного программирования.

Чарльз Хоар — разработчик оригинального и эффективного алгоритма “быстрой сортировки”, который применяется повсюду.

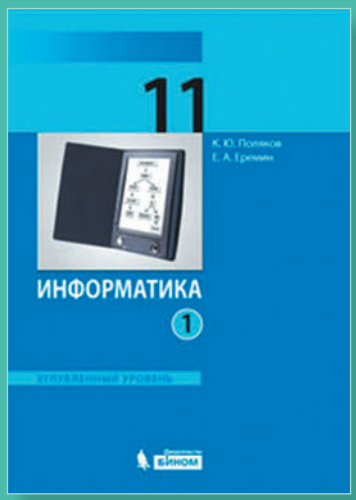
В отличие от предыдущих персон, которые относятся к университетской элите, Билл Гейтс и Стив Джобс — выдающиеся бизнесмены, сделавшие очень многое для того, чтобы достижения ученых пробили себе дорогу в повседневную жизнь. Они занимались тем, чем считали нужным заниматься, не обращая внимания на то, “что станет говорить Княгиня Марья Алексѣвна”.

Какая компьютерная техника и IT-сервисы окружают Вас сейчас? Насколько Вы “компьютерный человек” в обычной жизни? Это вопрос про все — про компьютеры и гаджеты, которые Вас окружают, про Ваш личный стиль социальной коммуникации в сети — насколько Вы живете в социальных сетях, насколько не можете жить без электронной почты и других средств коммуникации. Насколько бережете или не бережете свое личное пространство и время.

У меня есть настольный компьютер и ноутбук для поездок. Необходимость приобретения различных цифровых устройств определяется очень просто — в первую очередь нужно ответить на вопрос “Какую *твою* задачу позволит решать это устройство?”. Если ответа на этот вопрос нет, проблема отпадает сама собой. Это системный анализ. С этой же точки зрения, на мой взгляд, нужно подходить и к социальным сетям. Если они помогают решать какие-то реальные проблемы, нужно их использовать как инструмент. Подход “соцсети — это плохо, там столько грязи” следует рассматривать скорее с точки зрения религии. Каждый ищет и находит то, что ему нужно. Так, например, наиболее интересные вопросы и обсуждения, касающиеся нашего учебника, появляются в соцсети ВКонтакте: *http://vk.com/club54960632*. Совершенно спокойно обхожусь без компьютера длительное время. Такое случается, например, во время летнего отдыха. В походах, которые иногда длятся около месяца, нет не только Интернета, но временами и мобильной связи. В этом есть только один недостаток — после возвращения приходится долго “разгребать” накопившуюся электронную почту.

Если бы в Вашем распоряжении была лишь одна лекция (допустим — “пара”, полтора часа) и полная свобода рассказывать о том, что Вам интересно в информатике, чему бы Вы посвятили это время?

Я всегда стараюсь убедить талантливых и творческих молодых людей НЕ становиться программистами. Дело в том, что программист — всегда исполнитель. Это значит, что какой-то творческий человек что-то придумал и для реализации этой идеи нанял программиста, который обречен делать то, что ему скажут. Да, он может сделать свою работу хорошо или плохо, но суть от этого не меняется — программист всегда остается в тех рамках, которые



Учебник 2013 г.

установлены заданием. Исключения есть (например, успешные стартапы), но это капли в море. К сожалению, знаю талантливых людей, закончивших “программистские” специальности и ставших классными специалистами, которым через 10–15 лет работы настолько надоело “гонять байты”, что они теперь пытаются реализовать себя в совершенно других областях.

Нравится программировать? Отлично! Почему бы не выбрать профессию, в которой программирование необходимо, но не является основным делом. Примеры таких специальностей — автоматическое управление, робототехника. Здесь можно самому ставить задачи и самому их решать.

Есть ли какой-то факт из информатики — может быть, какой-то алгоритм, — который произвел на Вас сильное, запоминающееся впечатление?

Всегда очень нравилась битовая операция “исключающее ИЛИ” (XOR). На ней основано простейшее шифрование. С ее помощью можно быстро обнулить любое значение регистра процессора. Она позволяет выполнить обмен значений двух ячеек памяти без использования буфера (вспомогательной ячейки).

Замечательна идея программного стека применительно к вызовам подпрограмм и хранению локальных данных.

Есть ли у Вас любимая задачка или задачки?

Очень нравится задача о размещении туалета в вагонах поезда, которую приводил Э.Дейкстра в своих “Программистских баснях” (см., например, *http://koi8.pp.ru/dijkstra.html*). Действительно, человек, решивший ее, проявил типичный программистский подход. Кстати, это непосредственно относится к вопросу о применимости знаний на практике в свете ФГОС.

Какую тему/темы школьного курса (будем считать, что мы находимся в пространстве ФГОС) Вы не хотели бы рассказывать детям? Ну не нравится она Вам.

Мне всегда не нравились темы, которые сводятся к болтовне. Информатика — конкретная наука, где утверждения строго доказываются, а факты выражаются в числах. Услышав выражения типа “владение умением понимать”, хочется отойти в сторону.

Другая крайность — материал, который относится к университетскому курсу. Вспомним хотя бы “диагональное доказательство несуществования” в общеобразовательном стандарте 2004 года. Пример из современного варианта ЕГЭ — задача на решение системы логических уравнений (B15). С трудом можно представить себе практическую задачу, в которой это могло бы реально пригодиться.

Надо ли сейчас вообще учить предмету “информатика” в школе? Ведь навыки работы на компьютерах, в сетях, с прикладным ПО дети получают и без этого и гораздо раньше, чем начинают изучать предмет. А специалистов вполне можно готовить в вузах. В школе-то зачем?

Сейчас школьная информатика во многом выполняет профориентационную задачу — выявить людей, которые склонны к серьезной работе в этой области. Это что-то вроде “авиационного теста” Алана Купера из книги “Психбольница в руках пациентов”. Если человеку, поднявшемуся по трапу самолета, предложить выбор: пойти в кабину пилота или в салон для пассажиров, программист всегда выберет кабину пилота. Несмотря на то что ему потребуется изучить работу многочисленных приборов и органов управления, он получает полный контроль над ситуацией и принимает на себя ответственность за результат полета. Большинство “нормальных людей” выберет пассажирский салон.

Благодаря всеобщему обучению информатике на момент окончания школы выпускники уже ясно понимают, кому из них нужно идти “в кабину пилота”, а кому — нет.

Какой Вы представляете себе информатику в школе через пять лет? А через десять лет?

Через 5–10 лет скорее всего ничего принципиально не изменится. Дальше — видно будет. Компьютеры уже становятся такими же бытовыми приборами, как холодильники, телевизоры и стиральные машины. Поэтому вполне вероятно, что информатики и программирования как школьного предмета вскоре не будет. Теоретические основы и принципы работы цифровой техники будут изучаться в курсах математики и физики.

Какой вопрос Вы хотели бы задать себе, чтобы на него ответить?

Вечный вопрос, который задает себе каждый: есть ли какой-то смысл в том, что я делаю? Тут можно только верить, что ответ на этот вопрос положительный.

Задача Дейкстры о поездах*

Много лет назад в одной только что основанной железнодорожной компании кто-то из директоров — вероятно, тот, что заведовал финансовым отделом, — обнаружил, что можно существенно сократить начальные капиталовложения, если оборудовать туалетами только пятьдесят процентов вагонов в поездах дальнего следования. Так они и решили сделать. Однако как только компания начала перевозить пассажиров, стали стремительным потоком поступать жалобы насчет туалетов. Расследование, организованное компанией, показало, что случилось самое естественное: несмотря на молодость, компания уже начала страдать от неслаженности действий между своими подразделениями. Решение директора о туалетах не было доведено до персонала сортировочной станции, и они со всеми вагонами обращались одинаково. В результате некоторые поезда компоновались так, что в них, можно сказать, и вообще не было туалетов.

Чтобы разрешить это затруднение, с каждым вагоном связали бит информации, определявший, есть ли в нем туалет или нет, и сортировочной станции приказали компоновать поезда так, чтобы вагонов двух типов в них было по возможности одинаковое количество. Это было не совсем просто для сортировочной, но когда они все-таки разобрались, то ответственные за процесс сортировки сотрудники очень гордились собой.

Несмотря на то что теперь поезда компоновались по-новому, поток жалоб не иссякал. Организовали новое расследование, и выяснилось, что, хотя примерно в половине вагонов каждого поезда теперь и были туалеты, иногда все такие вагоны оказывались в одном конце поезда. Это положение решили исправить, составив новые инструкции для сортировочной станции, согласно которым при компоновке вагоны с туалетами и вагоны без туалетов следовало чередовать. Это доставило сортировщикам известные трудности, но, поворчав сперва, они справились и с ними.

Тем не менее жалобы продолжались. Оказалось, что, поскольку туалеты были у вагонов с одной стороны, расстояние между двумя соседними туалетами могло достигать трех вагонов. У мамаш с детьми, испытывавшими острую нужду, это зачастую приводило к катастрофическим последствиям, особенно когда проходы были заставлены багажом. На все вагоны с туалетами тогда повесили еще один бит информации, так что они превратились в направленные объекты, и новые инструкции гласили, что в каждом поезде все вагоны нужно ориентировать в одну и ту же сторону. На этот раз новые инструкции встретили на сортировочной станции, мягко говоря, без всякого энтузиазма, так как поворотных кругов едва хватало; чтобы быть совершенно справедливым по отношению к работникам сортировочной, надо сказать, что их не хватало вовсе, и только благодаря хитроумию и смекалке этим людям удалось разрешить проблему.

Теперь, когда туалеты распределялись равномерно по каждому поезду, в компании решили, что все затруднения преодолены. Но пассажиры продолжали жаловаться: хотя от любого места расстояние до ближайшего туалета не превышало длины одного вагона, пассажиры (испытывавшие нужду) не знали, в какую сторону следовало им отправляться ковылять по коридору! Чтобы устранить это новое затруднение, во всех проходах нарисовали стрелки с надписью “ТУАЛЕТ”. В результате вагоны без туалетов тоже стали асимметричными объектами, которые нужно было ориентировать в правильную сторону при сортировке.

Когда на сортировочной станции прочитали новую инструкцию, там установилась атмосфера отчаяния и назревавшего бунта: выполнить предписания было физически невозможно.

В этой критической ситуации некто, чье имя впоследствии было утеряно, так что теперь его уже невозможно установить, сделал следующее наблюдение. Если к каждому вагону с туалетом с того конца, где располагался туалет, прицепить отныне и навеки вагон без туалета, то сортировочной станции пришлось бы иметь дело уже не с N направленных вагонов двух типов, а с N / 2 одинаковых объектов, которые во всех случаях и для любой задачи можно было рассматривать как симметричные. И это решило все проблемы сортировки разом; цена решения была невелика: во-первых, в поездах должно было быть только четное количество вагонов (немногие недостающие вагоны были куплены на деньги, сэкономленные дурнем-финансистом), а во-вторых, пришлось слегка жульничать насчет равных расстояний между туалетами. Но в конце концов последняя пара метров — не в счет.

Во времена, когда случилась эта история, человечество не было еще облагодетельствовано автоматическими компьютерами, однако наш аноним, придумавший это решение, заслуживает права считаться первым в мире компетентным программистом.

Рассказывая эту историю, Дейкстра добавлял:

“Я рассказывал эту байку в разной аудитории. Как правило, программисты приходят в восторг, а управляющие неизменно по ходу повествования все более и более раздражаются; что касается настоящих математиков, то они не могут понять, в чем же здесь соль”.

* Та самая легендарная задача “О туалетах”, которую любит К.Ю. Поляков.

Семакин Игорь Геннадьевич

Доктор педагогических наук, кандидат физико-математических наук, профессор кафедры прикладной математики и информатики Пермского государственного национального исследовательского университета. Ведущий автор и руководитель авторского коллектива общеобразовательного курса информатики для средней школы. Под его руководством и при его участии выпущено более 30 учебников и учебных пособий для общеобразовательной школы, системы высшего и среднего специального образования. Область научных интересов: содержание и методика преподавания информатики, информатизация образования. Но это можно прочитать и на последней обложке учебника. А вот чего там прочитать нельзя ☺.

Люблю свой университет, с которым связано полвека моей жизни; люблю горы и песни под гитару — рецидив студенческой молодости; люблю корпеть над текстами; люблю преподавать и вообще — общаться со студентами; люблю программировать — чистит мозги; люблю ходить на лыжах и плавать; люблю жарить шашлыки на даче; люблю возиться с внуками; люблю хорошую музыку и хорошие книги.

Как Вы пришли в информатику? Первый компьютер, на котором Вы работали? Первый язык программирования, с которым Вы познакомились? Первые задачи, которые Вы решали?

В 1961 году, в пермской школе № 102 был открыт первый в городе класс с профильной подготовкой по специальности “программист-математик”. Я учился в нем с 9-го по 11-й классы. Обучение проходило на базе Вычислительного центра Пермского госуниверситета (ВЦ ПГУ). ЭВМ, на которой нас учили программированию, называлась “Арагац”. Это была ламповая машина, совместимая по архитектуре с ЭВМ “М-20” конструкции С.А. Лебедева. “Арагац” проработала в ПГУ 10 лет: с 1960-го по 1970 г. Программировали на ней только в машинных кодах. До сих пор помню тему своей выпускной программы в 11-м классе: решение систем линейных алгебраических уравнений методом Зейделя.

В 1964 году я поступил на физический факультет ПГУ. С 3-го курса стал учиться по специальности “Теоретическая физика”. Расчеты по своим курсовым работам выполнял все на той же “Арагац”. Это были задачи из области теории гидродинамической устойчивости. Сейчас невозможно поверить в то, что такие серьезные научные задачи решались на ЭВМ, оперативная память которой составляла около 12 килобайт, а быстродействие — 10 тысяч операций в секунду.

На 5-м курсе университета (1968–1969 г.) я был направлен на выполнение дипломной работы в Вычислительный центр Сибирского отделения Академии наук СССР, в г. Новосибирск. Вот там мне впервые пришлось осваивать язык программирования высокого уровня. Это был АЛГОЛ. На отечественных ЭВМ 2-го поколения АЛГОЛ был одним из основных языков. В ВЦ СОАН была создана одна из версий АЛГОЛа, которая называлась языком АЛЬФА. Главным разработчиком языка АЛЬФА был Андрей Петрович Ершов. В языке АЛЬФА использовалась русскоязычная нотация. Позже, по аналогии с АЛЬФА, А.П. Ершовым был разработан учебный Алгоритмический язык для школьной информатики.

В ВЦ СОАН я решал достаточно трудоемкую задачу по расчету вихревых потоков в атмосфере. Отладку программы проводил на ЭВМ 2-го поколения “М-220”. А когда дело дошло до рабочих расчетов, то меня “пустили” на “БЭСМ-6”! Это была первая отечественная (советская) машина с быстродействием 1 млн. оп./сек. Переход от “Арагац” к “БЭСМ-6” ощущался мной как прорыв в космос!

Первые пять фамилий из мира информатики, которые приходят Вам в голову? Если можете, пожалуйста, коротко расскажите, почему именно они.

Первыми мне могут прийти в голову 2–3 фамилии. Для пяти уже надо напрягаться ☺. Безусловно, первая фамилия — это Андрей Петрович Ершов. Выше я уже о нем говорил. Хотя в ВЦ СОАН один год я работал рядом с А.П. Ершовым, но непосредственно с ним общаться не приходилось. Андрей Петрович был завотделом программирования. А я дипломную работу делал в другом отделе — динамической метеорологии, которым руководил академик Гурий Иванович Марчук. Новосибирская школа программирования, которую возглавлял А.П. Ершов, была одной из ведущих в СССР. И первоначально Андрея Петровича я знал именно в этом качестве. Уже позже, в 1980-х годах, произошло мое второе пересечение с А.П. Ершовым на почве школьной информатики. В 1985 году я был направлен в МГУ на учебу, организованную для вузовских преподавателей, которым предстояло летом этого же года подготовить на курсах первых школьных учителей информатики. Организатором этой учебы был А.П. Ершов. Было совершенно очевидно, что включение в школьную программу курса информатики произошло во многом благодаря его энергии и авторитету.

Первые ЭВМ использовались главным образом для решения научно-технических задач. Успешность здесь зависела не только от быстродействия машин, но и от качества применяемых численных методов. В тот период советская школа математиков-численников была одной из ведущих в мире. Ее лидеры — академики А.А. Самарский, А.Н. Тихонов, Н.Н. Яненко, Г.И. Марчук и др.

Какая компьютерная техника и IT-сервисы окружают Вас сейчас? Насколько Вы “компьютерный человек” в обычной жизни? Это вопрос про все — про компьютеры и гаджеты, которые Вас окружают, про Ваш личный стиль социальной коммуникации в сети — насколько Вы живете в социальных сетях, насколько не можете жить без электронной почты и других средств коммуникации. Насколько бережете или не бережете свое личное пространство и время.

Без ПК, без электронной почты, без Интернета свою работу представить не могу. Наверное, не написал бы и десятой части книг, статей и пр. из того, что удалось написать за последние годы.

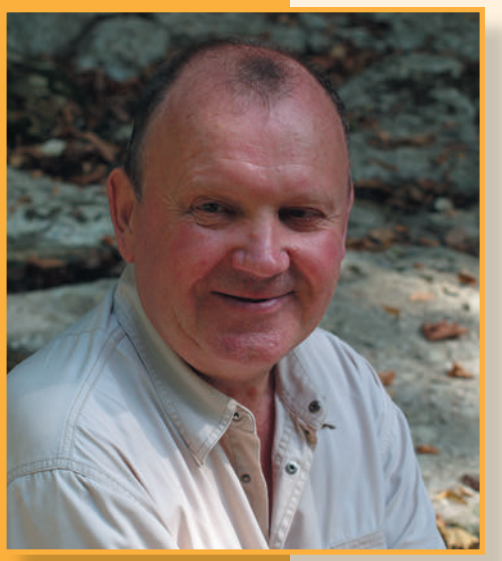
Что касается социальных сетей, то я тут консерватор ☹. Мне хватает той доли публичности, с которой связана моя работа. Все необходимые мне контакты обеспечивают телефон, e-mail, форум на сайте издательства “БИНОМ”, Skype. Поэтому ни блогов, ни твиттеров не веду, “ВКонтактах” не участвую.

Если бы в Вашем распоряжении была лишь одна лекция (допустим — “пара”, полтора часа) и полная свобода рассказывать о том, что Вам интересно в информатике, чему бы Вы посвятили это время?

Наверное, истории ЭВМ и ИКТ. Ясно, что за полтора часа всего не расскажешь! Но это самая подходящая тема, с помощью которой можно заинтересовать учеников в изучении предмета. И хорошо бы еще успеть затронуть социальные последствия информатизации.

Надо ли сейчас вообще учить предмету “информатика” в школе? Ведь навыки работы на компьютерах, в сетях, с прикладным ПО дети получают и без этого и гораздо раньше, чем начинают изучать предмет. А специалистов вполне можно готовить в вузах. В школе-то зачем? Какой Вы представляете себе информатику в школе через пять лет? А через десять лет?

Эти вопросы связаны между собой. Если в содержании школьной информатики видеть только технологии и программирование (как это обозначено в ФГОС) и выделять на преподавание предмета один урок в неделю, то ее изучение потеряет смысл. Если же идти по пути усиления фундаментального и методологического (метапредметного) содержания информатики, то предмет имеет шанс не только выжить в школе, но и развиваться в сторону системообразующей основы всего школьного образования.



Уваров Александр Юрьевич

Доктор педагогических наук, ведущий научный сотрудник Вычислительного центра Российской академии наук (ВЦ РАН) и главный научный сотрудник Федерального института развития образования (ФИРО). Он лауреат премии Правительства РФ в области образования, эксперт Института ЮНЕСКО по информационным технологиям в образовании, действительный член международного профессионального партнерства Education Impact (*http://www.educationimpact.net/*), член редакционной коллегии журнала “Образовательная политика”, эксперт международных образовательных программ “Партнерство в образовании” (Microsoft) и “Обучение для будущего” (Intel).

А.Ю. Уваров работал школьным учителем, был научным сотрудником и заведовал лабораторией в Академии педагогических наук СССР. Возглавлял Управление информатики и вычислительной техники Министерства просвещения СССР на первом этапе внедрения компьютеров в образование (1985–1988), был проректором Университета Российской академии образования. Он является инициатором и научным руководителем многих национальных и международных исследовательских проектов в области информатизации образования.

А.Ю. Уваров является автором и соавтором более 200 опубликованных работ по проблемам дидактики, разработки учебных программ, педагогического дизайна, использования ИКТ в образовании. Среди них книги “Учебные телекоммуникационные проекты в классе”, “Электронный учебник: теория и практика”, “Телекоммуникация, образование, профессионализм”, “Информатизация школы: вчера, сегодня, завтра”, “Кооперация в обучении: групповая работа в классе”, “Российская школа и новые информационные технологии: взгляд в следующее десятилетие”, “Распространение инновационных учебно-методических материалов” и др.

Как Вы пришли в информатику? Первый компьютер, на котором Вы работали? Первый язык программирования, с которым Вы познакомились? Первые задачи, которые Вы решали?

Полвека назад, когда я был студентом, на первом этаже в нашем корпусе установили компьютер “БЭСМ-2М”. Для этого снесли несколько стен и в зале размером с две поточных аудитории разместили настоящее чудо техники из фантастических романов. “За стенами зданий кипела работа: мигали, гудели, считали приборы” — так поэт из журнала “Юность” тех лет описывал вычислительный центр на космодроме одной из инопланетных цивилизаций ☺.

В то время мы с моим товарищем Ю.Ф. Лукьянцом, который позднее стал одним из разработчиков интеллектуальной составляющей отечественной ПРО, увлекались чтением книг по кибернетике. К научно-технической информации в СССР относились серьезно. Достаточно вспомнить, что знаменитый сборник “Автоматы” под редакцией К.Шеннона и Дж. Маккарти (*http://www.twirpx.com/file/538508/*) увидел свет на русском языке раньше, чем на английском.

На занятиях нас учили работе с аналоговыми вычислителями, а о цифровых машинах лишь упоминали на лекциях. Однако расчет устойчивости летательных аппаратов казался нам тогда слишком обычным занятием. Мы “желали странного”, и когда у нас появилась возмож-

ность выполнить самостоятельную научно-исследовательскую работу по автоматизации проектирования технологических процессов, мы с увлечением занялись программированием “автомата-конструктора” на БЭСМ.

Наш старший товарищ, Ю.А. Багрянцев, помогал мне познать искусство, радость и печаль программирования. Единственными доступными средствами программирования в то время были язык команд БЭСМ и библиотека стандартных программ. Такое программирование требовало недюжинной изобретательности. Это было время, о котором А.И. Ситковский, позднее — разработчик одной из оборонных систем на “Эльбрусах”, написал:

“Заплетут, где надо, розы
Программисты-виртуозы”.

Доступ к ЭВМ был ограничен, и мы всерьез осознали простую истину тех времен: наше время ничего не стоит, а машинное время стоит дорого. Потребовалось огромное количество дешевого личного и много часов дорогостоящего машинного времени, прежде чем несложные по современным понятиям программы оптимизации раскроя листового материала и составления карт обработки деталей на токарном станке заработали. Это был триумф. Его результатом стала публикация статьи студентов в одном из научных изданий.

Первые пять фамилий из мира информатики, которые приходят Вам в голову? Если можете, пожалуйста, коротко расскажите, почему именно они.

Все помнят, что слово “информатика” приобрело свое современное значение немногим более четверти века назад, когда в АН СССР появилось Отделение информатики. В середине 60-х годов, когда речь шла о теории информации, теории библиотечного дела и информационной работы (информатике), вычислительной математике, программировании, проектировании автоматизированных систем, использовании ЭВМ и т.п. в качестве вмещающего термина, как правило, использовалось слово “кибернетика”.

А.П. Ершов и Дж. Маккарти на Летней школе юных программистов в Новосибирске, 1987 г. (http://ershov.iis.nsk.su/ershov/russian/pict.html)

Поэтому, когда говорят об информатике, я прежде всего вспоминаю К.Шеннона и Н.Винера, Джона фон Неймана и Дж. Маккарти. Книги первых трех сформировали мой интерес к информатике, а с Дж. Маккарти мне довелось встречаться лично. Этот замечательный человек не только изобрел язык Лисп и ввел термин “искусственный интеллект”. Он был большим другом нашей страны, часто приезжал в СССР, выступал перед участниками Школы юных программистов. Впервые мне посчастливилось общаться с ним в 1966 году на семинаре, который проходил на кафедре высшей нервной деятельности биофака МГУ.

Говоря об отечественных ученых, я прежде всего вспоминаю академиков А.И. Берга и А.П. Ершова.

Без А.И. Берга в нашей стране кибернетика еще долго считалась бы “лжеучением” и “продажной девкой империализма”, как писали тогда в энциклопедиях. Без личного мужества и гражданской позиции академика А.И. Берга книга “Кибернетика” Н.Винера не появилась бы на русском языке. Будучи заместителем министра, он приказал подчиненному ему издательству “Советское радио” перевести и выпустить эту книгу массовым тиражом.

Помимо множества заслуг в области радиоэлектроники, академик А.И. Берг создал и много лет возглавлял в АН СССР Совет по комплексной проблеме “Кибернетика”, из которого отпочковались несколько связанных с информатикой институтов Академии. Он потратил немало усилий, чтобы привлечь внимание отечественных педагогов к программированному обучению, поддержал создание в АПН СССР лаборатории количественных и инструментальных методов дидактических исследований, в которой мне довелось работать. Представлять на страницах этого изда-

Источник: http://maidano.ru/razdel2/desiatya/desiatvtor.html

ния “отца школьной информатики” академика А.П. Ершова было бы странно. Однако, возможно, не все знают, что Андрей Петрович был не только разработчиком трансляторов, но и настоящим визионером. Четверть века назад он ясно видел все этапы продолжающейся и сегодня информационной революции, предсказывал возникновение облачных вычислений. Заметим, что необходимые для этого научные представления в то время уже формировались, и он приложил к этому немало усилий. Для меня большой школой стала совместная с академиком А.П. Ершовым работа над первой отечественной концепцией информатизации образования. Концепция сделала термин общеупотребительным.

Знаменитую метафору А.П. Ершова “Программирование — вторая грамотность”, которая была сформулирована в докладе на конгрессе IFIP три десятилетия назад, уместно помнить и сегодня. Для А.П. Ершова программирование было не профессией, а способом видения мира. С текстом его выступления (*http://ershov.iis.nsk.su/russian/second_literacy/article.html*), как мне кажется, должен познакомиться каждый образованный человек.

Задача, которую ученый сформулировал много десятилетий назад, актуальна и сегодня.

Недаром законодатели в США предложили недавно придать языкам программирования статус “критических иностранных языков”, с тем, чтобы у местных властей был стимул вводить уроки программирования в начальных классах школ и в старших группах детского сада (*http://www.golos-ameriki.ru/content/computing-languages-status-congress/1811464.html*). Эксперименты по обучению программированию маленьких детей, которые проводит один из авторов первого школьного учебника по информатике Анатолий Георгиевич Кушниренко, показывают, что это совсем не пустая фантазия.

Новые образовательные стандарты в нашей стране также требуют формирования “информационной культуры” у каждого школьника. Самое ясное и простое объяснение этому феномену содержится в работах А.П. Ершова.

Какая компьютерная техника и IT-сервисы окружают Вас сейчас? Насколько Вы “компьютерный человек” в обычной жизни? Это вопрос про все — про компьютеры и гаджеты, которые Вас окружают, про Ваш личный стиль социальной коммуникации в сети — насколько Вы живете в социальных сетях, насколько не можете жить без электронной почты и других средств коммуникации. Насколько бережете или не бережете свое личное пространство и время.

Сегодня большинство россиян живет в окружении цифровых устройств. Кто-то подсчитал, что сегодня на каждого приходится по два мобильных телефона. Почти четверть века назад у меня появился первый портативный компьютер Toshiba 1000, который тогда весил меньше двух килограмм. С тех пор я постоянно пользуюсь носимыми компьютерами. Пару лет назад мне подарили планшет, но я им практически не пользуюсь: выйти в Интернет можно и с привычного компьютера, а ничего более это устройство не может. Слушать музыку и смотреть кино проще на смартфоне.

Каждого из нас, как и обещал четверть века назад академик А.П. Ершов, дома окружают микропроцессоры и компьютеры. В этих условиях, как мне кажется, все большее значение приобретают скорость и цена доступа в Интернет. Это уже важнее, чем мощность носимого компьютера. Если у вас доступ в сеть ограничена скоростью 10 Мбит/с, то жить еще можно, но лучше, чтобы она была не менее 100 Мбит/с. Эта очевидная истина признана всеми, кроме тех, кто определяет скорость доступа в Интернет в школах. Если канал на школу пропускает всего 1 Мбит/с, то о каком реальном использовании Интернета в учебном процессе можно говорить? Эти мысли приходят ко мне каждый раз, когда во время поездок я попадаю в места, где скорость доступа в Интернет сильно ограничена.

Лет пятнадцать назад ко мне на консультацию пришла одна из моих студенток. Я предложил ей скопировать ее работу на стоящий в аудитории компьютер, но она сказала, что не носит с собой сменных

носителей. У нее, как у сисадмина, все хранится на сервере и доступно через Интернет. В те времена это мог позволить себе только сисадмин. Сегодня это норма для каждого. Например, все мои архивы с фотографиями лежат на SkyDrive. Там же хранятся все рабочие материалы проектов, в которых я участвую с коллегами. Использование коллективного информационного пространства у нас давно стало нормой. Это резко повышает производительность труда. Поэтому нормально работать без доступа в Интернет я сегодня уже вряд ли смогу ☹.

Если бы в Вашем распоряжении была лишь одна лекция (допустим — "пара", полтора часа) и полная свобода рассказывать о том, что Вам интересно в информатике, чему бы Вы посвятили это время?

В различные времена мне были интересны различные темы. Сегодня меня очень интересуют облачные вычисления. Я даже написал об этом статью для педагогов, которая опубликована в первом номере "Информатики" за 2014 год.

Есть ли какой-то факт из информатики — может быть, какой-то алгоритм, — который произвел на Вас сильное, запоминающееся впечатление?

В свое время меня поразило, какой глубокой может оказаться такая простая на первый взгляд задача, как сортировка массивов. Меня восхитила мощь рекурсивных описаний, когда запись программы достаточно сложных вычислений мне удалось свернуть буквально в несколько строк. Совершенно гениальна идея коммутации пакетов, которая лежит в основе TCP/IP. Чтобы не продолжать этот список, упомяну лишь об идее облачных вычислений, где множество самовосстанавливающихся параллельно выполняемых процессов оперируют с общей памятью, обмениваясь между собою транзакциями.

Какую тему/темы школьного курса (будем считать, что мы находимся в пространстве ФГОС) Вы не хотели бы рассказывать детям? Ну не нравится она Вам.

Таких много ☹.

Жаль, что ОБЯЗАТЕЛЬНЫЙ МИНИМУМ СОДЕРЖАНИЯ ОБРАЗОВАНИЯ ПО ИНФОРМАТИКЕ обязывает детей и учителей "изучать" (причем за пару часов ☺) темы поистине необъятные.

• Понятие информации. Информационные процессы в живой природе, обществе и технике: получение, передача, преобразование, хранение и использование информации.

• Информационная деятельность человека. Информационные основы процессов управления.

• Вещество, энергия, информация — основные понятия науки.

• Информационные основы процессов управления.

• Информационная культура человека. Информационное общество.

Здесь столько тайн. Понимание информационных процессов — глубочайшая философская тема. Однако минимумы не оставляют учителям и школьникам никакой иной возможности, кроме повторения банальностей. За каждой из этих тем много непознанного, требующего глубоких размышлений, практической работы. Такое пренебрежительное, формальное отношение к очень глубоким вопросам заставляет подозревать, что авторы подобных минимумов слабо знакомы с предметом, не в полной мере владеют информационной культурой.

Я бы не хотел учить и информационным технологиям.

• Понятие текста и его обработки. Текстовый редактор: назначение и основные возможности. Редактирование и форматирование текста.

• Представление графической информации. Графический редактор: назначение и основные возможности.

Технологии обработки числовой информации:

• Электронные таблицы: назначение и основные возможности. Ввод чисел, формул и текста. Стандартные функции. Построение диаграмм. Использование электронных таблиц для решения задач.

Технологии хранения, поиска и сортировки информации:

• Базы данных: назначение и основные возможности. Системы управления базами данных. Ввод и редактирование записей. Сортировка и поиск записей.

Компьютерные коммуникации:

• Локальные и глобальные компьютерные информационные сети. Основные информационные ресурсы: электронная почта, телеконференции, файловые архивы. Сеть Интернет. Технология World Wide Web (WWW). Поиск информации.

Для этого есть достаточно много различных компьютерных учебных курсов, которые любой школьник может освоить самостоятельно…

Надо ли сейчас вообще учить предмету "информатика" в школе? Ведь навыки работы на компьютерах, в сетях, с прикладным ПО дети получают и без этого и гораздо раньше, чем начинают изучать предмет. А специалистов вполне можно готовить в вузах. В школе-то зачем?

Этот вопрос сформулирован так, что сразу хочется заявить: освоение навыков работы на компьютере и изучение информатики имеют так же мало общего, как освоение каллиграфии с помощью гусиного пера и изучение русского языка. Вопрос не в том, изучать ли информатику в школе, а в том, что именно надо изучать. Я попытался ответить на этот вопрос пятнадцать лет назад в своей статье "Чему учить на уроках информатики" (*http://inf.1september.ru/1999/art/uvar1.htm*), и мое мнение с тех пор мало изменилось.

Сегодня, как мне кажется, неплохо бы вообще отойти от изучения в школе отдельных предметов. Я знаю немало думающих учителей, которые считают, что разорванное на предметы школьное образование не позволяет учащимся формировать для себя единую картину мира. Предметное образование появилось как результат уплощения педагогической профессии. Теперь стало считаться нормальным, что учитель русского языка может не знать географии, а учитель химии плохо разбирается в математике. В этих условиях требовать от учащихся осваивать школьную программу, которая включает в себя все эти предметы, — нонсенс!

Мне кажется, что пора вернуться к тому, что учитель — это Образованный Человек с большой буквы. Именно по культуре учителей надо судить о культуре общества. Информационная культура, знакомство с основами информатики важны для жизни в современном мире не меньше, чем личная гигиена и физическая культура (здоровье).

Возможно, нас ждет постепенный отход от чисто предметного преподавания, переход к проектной работе учащихся, в рамках которой они будут не только усваивать основы соответствующих наук, но также приобретать общенаучные, общеучебные и гражданские компетенции. За рубежом их часто называют "компетенции XXI века"*.

Какой Вы представляете себе информатику в школе через пять лет? А через десять лет?

В книге "Ярче тысячи солнц" описано, как в канун Второй мировой войны группа известных физиков обсуждала политические события. Ее участники регулярно давали свои предсказания о возможных изменениях на ближайшую неделю. В результате всегда выигрывал тот, кто делал простую интерполяцию текущего состояния на будущее. Эта стратегия не позволила предсказать только два события: начало Второй мировой войны в Европе и атаку японцев на Пёрл-Харбор.

Нам всем кажется, что за следующие 5–10 лет вряд ли что-то кардинально изменится в школе. И для этого есть достаточно оснований. Однако информатизация образования продолжается. Современные дети не хотят мириться с традиционной школьной рутиной. Они "голосуют ногами". Изменения неизбежны, но только в том случае, если мы сами их готовим и проводим в жизнь. Так поступают, например, учителя информатики школы № 550 в Санкт-Петербурге. Возможно, их пример окажется заразительным и для других школ, где думают о личностно ориентированном обучении, реальной индивидуализации учебного процесса, где хотят, чтобы современное представление об информатике получил КАЖДЫЙ учащийся школы.

---

* Их описание можно найти, например, в журнале "Образовательная политика" № 1 за 2014 год.

ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ г. МОСКВЫ
ИЗДАТЕЛЬСКИЙ ДОМ «ПЕРВОЕ СЕНТЯБРЯ»
МОСКОВСКИЙ ПЕДАГОГИЧЕСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

МОСКОВСКИЙ ПЕДАГОГИЧЕСКИЙ МАРАФОН УЧЕБНЫХ ПРЕДМЕТОВ 2014

24 МАРТА – 18 АПРЕЛЯ

РАСПИСАНИЕ ДНЕЙ ПЕДАГОГИЧЕСКОГО МАРАФОНА

24 марта	День учителя технологии *
25 марта	**Открытие Марафона** День классного руководителя
26 марта	День школьного психолога День учителя ОБЖ
27 марта	День здоровья детей, коррекционной педагогики, логопеда, инклюзивного образования и лечебной физической культуры
28 марта	День учителя начальной школы (день первый)
29 марта	День учителя начальной школы (день второй)
30 марта	День дошкольного образования
1 апреля	День учителя географии
2 апреля	День учителя химии
3 апреля	День учителя биологии
4 апреля	День учителя информатики
5 апреля	День учителя физики
6 апреля	День учителя математики
8 апреля	День учителя истории и обществознания
9 апреля	День учителя МХК, музыки и ИЗО
10 апреля	День школьного и детского библиотекаря
11 апреля	День учителя литературы
12 апреля	День учителя русского языка
13 апреля	День учителя английского языка
15 апреля	День учителя французского языка
16 апреля	День школьной администрации
17 апреля	День учителя физической культуры
18 апреля	День учителя немецкого языка **Закрытие**

marathon.1september.ru

! Обязательная предварительная регистрация на все дни Марафона с 20 февраля 2014 года на сайте marathon.1september.ru

! Каждый участник Марафона, посетивший три мероприятия одного дня, получает официальный именной сертификат (6 часов)

В дни Марафона ведущие издательства страны представляют книги для учителей

Начало работы каждого дня – 9.00. Завершение работы – 15.00

УЧАСТИЕ БЕСПЛАТНОЕ. ВХОД ПО БИЛЕТАМ

РЕГИСТРИРУЙТЕСЬ, РАСПЕЧАТЫВАЙТЕ СВОЙ БИЛЕТ И ПРИХОДИТЕ!

Место проведения Марафона: МПГУ, ул. Малая Пироговская, дом 1, стр. 1 (в 5 минутах ходьбы от ст. метро «Фрунзенская»)
* Место проведения Дня учителя технологии: ЦО № 293, ул. Касаткина, 1а (ст. метро «ВДНХ»)

По всем вопросам обращайтесь, пожалуйста, по телефону **8-499-249-3138** или по электронной почте marathon@1september.ru

MongoDB: шаг от баз данных к базам знаний

Е.А. Еремин,
г. Пермь

Никто пока точно не описал, как человек хранит свои знания. Но понять это очень хочется, хотя бы исходя из прикладной цели — передать накопленные людьми знания компьютеру и научить его работать не с базами данных, а с базами знаний.

Согласно последним теориям, вплотную подходящим к объяснению общей логики работы мозга, внутренняя модель мира у человека построена существенно иерархическим образом. В замечательной книге [1] по этому поводу читаем следующее:

“Почему кора головного мозга устроена иерархически?

Вы обладаете способностью размышлять о мире, передвигаться в окружающем пространстве и прогнозировать будущее, потому что в вашем неокортексе сформирована модель внешнего мира. …Иерархическая структура коры головного мозга хранит модель иерархического строения внешнего мира. Вложенная структура реального мира отображается вложенной структурой коры вашего головного мозга.

<…>

В каждый отдельно взятый момент времени вы можете воспринимать только определенную часть внешнего мира. Вы можете находиться только в одной комнате и смотреть только в одном направлении. Благодаря иерархическому строению коры головного мозга вы в состоянии осознать, что находитесь дома, в своей гостиной, и что вы смотрите в окно, даже если в данный момент ваш взгляд остановился на щеколде. Высшие зоны коры головного мозга поддерживают инвариантную репрезентацию вашего дома, более низкие — репрезентацию комнаты, а первичные — анализируют непосредственные визуальные символы.

<…>

Кора головного мозга обладает иерархической структурой, она состоит из миллиардов нейронов и триллионов синапсов. Нам трудно представить,

как такая логически простая, но по численности обширная система памяти может создавать наше сознание, язык, культуру, искусство, книги, науку и технологии. Однако это происходит, потому что наше интуитивное представление о способностях коры головного мозга и мощностях его иерархической структуры является неадекватным. Кора головного мозга на самом деле работает. В этом нет никакой магии. Это доступно вашему пониманию".

В свою очередь, системы искусственного интеллекта, т.е. программы, обрабатывающие определенным образом организованные компьютерные данные о внешнем мире, тоже так или иначе "мигрируют" в сторону иерархических (например, онтологии) или сетевых (карты предметной области, включая хорошо известные в образовании Concept Map) структур.

Исходя из сказанного о структуре знаний, становится понятно, что классические базы данных выглядят для представления знаний не очень подходяще. Мало того, что система связанных таблиц с жесткой структурой не очень-то похожа на естественную для человека иерархическую структуру с ветвями различного размера; при большом количестве таблиц (сущностей, разновидностей объектов в системе) все это еще и работает очень неэффективно. Те, кто знает и понимает программирование, вероятно, сейчас вспомнили очень похожую разницу между массивами и динамическими структурами.

Что надо изменить в базах данных, чтобы приблизить их к базам знаний? Очевидно, что необходимо прежде всего поменять их структуру и сделать ее менее жесткой. В связи с этим сразу вспоминается набирающая популярность система MongoDB [2]. Не берусь утверждать, что ее создатели, разрабатывая свое детище, рассуждали так, как описано выше; но факт остается фактом: Mongo гораздо лучше соответствует структуре человеческих знаний, нежели стандартная реляционная база.

Итак, главная цель данной публикации — наглядно продемонстрировать на несложных примерах, что Mongo является простым и естественным средством зафиксировать наши "разнородные" знания. Но уже в ходе подготовки материалов возникло и еще одно параллельное стратегическое направление. Оно состояло в том, чтобы попутно показать полезность умения программировать при работе с хорошей, но в чем-то не устраивающей нас системой: поскольку Mongo явно "молодой и развивающийся" продукт, он *пока* имеет весьма "аскетический" интерфейс. В связи с этим в разделе 3 будет рассказано, как, используя Delphi, можно самому сделать простейшие программные инструменты, которые существенно упростят общение с Mongo. Причем написать такую программу вполне по силам среднестатистическому школьнику.

Публикуемый материал специально разрабатывался для данной статьи. Даже если вы уже что-то знаете о Mongo, например, читали общепризнанную интернет-книгу [3] или познакомились в нашем журнале с практикумом по этому продукту [4], вы все равно найдете для себя что-то новое. Вместе с тем, статья написана так, что не требует никаких предварительных знаний о Mongo.

Итак, давайте посмотрим, насколько приспособлена Mongo для хранения наших знаний об окружающем мире.

1. Некоторые предварительные сведения

1.1. Место Mongo среди других баз данных

Как известно, классические реляционные базы данных устроены следующим образом. Заранее тщательно продумывается организация будущих данных, а затем сведения о получившейся структуре сохраняются в базу. Благодаря этим "данным о будущих данных" (часто их называют *метаданными*) все конкретные значения в базе получают строго определенное место и их можно хранить без всяких разделителей "друг за другом". Поскольку размеры всех данных предопределены, для доступа к ним достаточно очень простых вычислений. Скажем, если размер одной строки реляционной таблицы (*записи*) равен 400 байт, то третья запись начинается с 801-го байта (две предшествующие занимают ровно 800 байт). Зная начало записи и размеры каждого из столбцов (*полей*), немногим сложнее вычислить начало нужного поля и извлечь в качестве его значения заранее известное число байт. Как именно интерпретировать эти байты (как число, строку, дату и т.п.), тоже заранее известно из описания структуры.

Принятый способ организации данных позволяет получить высокое быстродействие, но зато неэффективно расходует память. Например, человек может носить очень короткую фамилию Ким, но все равно под поле с ней в базе данных будет отведено столько байт, сколько требуется для *максимального* количества символов в фамилиях. Кроме того, даже если у какого-нибудь поля есть значение всего в одной записи (допустим, речь идет о звании "мастер спорта" у школьников), то под него все равно приходится везде предусматривать место.

Реальные базы строятся не из одной, а из нескольких таблиц, но сути дела это существенно не меняет. Попутно заметим, что для выборки данных из системы связанных таблиц разработан специальный язык SQL, который часто отождествляют с реляционной организацией данных.

Таким образом, для реляционных баз данных характерно раздельное хранение данных о структуре таблицы и собственно данных и, как следствие, жесткая заранее предопределенная организация информации о каждом объекте (см. *рис.* 1 на с. 30, сравнивающий разные способы организации данных).

Рис. 1. Универсальность структуры данных в Mongo

Противоположным подходом к хранению данных служат базы “ключ/значение”, где каждое значение имеет собственное имя (ключ) и хранится вместе с ним независимо от других. Обычно ключ — это строка, а значение — набор байтов, причем размер обеих частей может выбираться произвольно. Названия ключей могут иметь общую часть, что позволяет производить простейшую группировку пар.

Конечно, в чистом виде такой способ организации данных годится только для самых примитивных баз вроде словарей и телефонных справочников. Для расширения возможностей подхода разрешают байты, лежащие в поле значения, интерпретировать как несколько значений, в том числе как ссылки на другие ключи. В итоге тем или иным способом можно объединять пары “ключ/значение” в группы, благодаря чему появляется возможность описать некоторый объект, обладающий набором свойств. Но все равно при таком подходе роль структуры данных сведена к минимуму.

Обсуждаемая нами сегодня база MongoDB занимает промежуточное положение: в отличие от реляционной базы у нее нет заранее предопределенной структуры, но зато по сравнению с базой “ключ/значение” в ней предусмотрены развитые средства для формирования структур данных. Главная разница между структурами в Mongo и в реляционной базе заключается в том, что в Mongo структура определяется не на уровне всей таблицы, а внутри каждой отдельной строки (строки в Mongo принято называть *документами*). В результате появляется возможность иметь строки разной структуры, что невозможно при реляционном подходе. Набор документов, связанных по смыслу, объединяется в общую *коллекцию* — некоторый аналог таблицы (на *рис.* 1 этот факт изображается пунктирным контуром).

Вот как об этом говорится в книге [3], которая является великолепным введением в Mongo.

“Вы можете спросить — зачем придумывать новые термины (коллекция вместо таблицы, документ вместо записи и поле вместо колонки)? Не излишнее ли это усложнение? Ответ в том, что эти термины, хоть и близки своим “реляционным” аналогам, но не полностью идентичны им. Основное различие в том, что реляционные базы данных определяют “колонки” на уровне “таблицы”, в то время как документ-ориентированные базы данных определяют “поля” на уровне “документа”. Это значит, что любой документ внутри коллекции может иметь свой собственный уникальный набор полей. В этом смысле коллекция “глупее”, чем таблица, тогда как документ имеет намного больше информации, чем строка”.

Нетрудно видеть, что при определенных условиях Mongo может “имитировать” своих конкурентов. Так, если потребовать от всех документов одинаковую структуру, то получается организация, весьма близкая к реляционной (не забывайте смотреть на *рис.* 1). Если, напротив, каждый документ формировать строго из двух полей — ключ и значение, то получим аналог противоположного предельного случая. Но, разумеется, ценность Mongo заключается не в повторении того, что уже умеют другие информационные системы, а в возможности создать некоторую промежуточную организацию данных, которая остальным средствам недоступна. Как мы увидим далее, именно такая промежуточная организация оказывается наиболее удобной для представления реальных знаний.

Особо отметим, что из-за более широких, чем у реляционных баз, возможностей Mongo называют *NoSQL* (***N**ot **o**nly **SQL*** — не только SQL) базой. Или более полно — *документ-ориентированной NoSQL* базой.

1.2. JSON — основа Mongo

JSON расшифровывается как ***J**ava **S**cript **O**bject **N**otation*, т.е. что-то вроде[1] нотации (правил) описания объектов на Java Script. Это несложная система правил, которая позволяет описывать характеристики объектов в форме текста, понятного человеку, и в то же время легко разбираемого на составляющие компьютерной программой.

Базовым элементом JSON является пара вида

<имя> : <значение>

Имя представляет собой некоторую последовательность символов, сформированную по обычным правилам для имен переменных. Значения могут быть *числовыми* (2, 2.3, 3e8) и *текстовыми* ('изучаем JSON', '38 попугаев'), а также объектами или массивами. *Объект* — это синтаксически правильная конструкция JSON, заключенная в фигурные скобки, например, трамвай 12-го маршрута может быть описан так:

transport : {type : 'трамвай', route : 12}

Массив может содержать несколько, одно или даже ни одного значения, которые заключены в квадратные скобки:

cities : ['Пермь', 'London', 'Варна']

Приятно сообщить, что для наших упражнений этих сведений о JSON вполне достаточно.

[1] Названия из четырех существительных с английского всегда нелегко перевести; поэтому авторы чаще всего стараются русский вариант не указывать (тем более что все слова по отдельности и так понятны!).

Таблица 1

	name	Процессор			Монитор		Дисковод		WiFi	
		name	f (в гГц)	внут./ внеш.	size (в дюймах)	внут./ внеш.	d (в дюймах)	внут./ внеш.	card	внут./ внеш.
1	desktop	AMD Sempron	2.5	внут.	17	внеш.	3.5	внут.		
2	notebook	Intel Celeron	2.4	внут.	14	внут.	3.5	внеш.		
3	netbook	Intel Atom	1.6	внут.	10	внут.		?	Intel link 5150	внут.

1.3. Установка MongoDB

Для практической реализации рассматриваемых ниже примеров вам потребуется работающая система Mongo. Ее дистрибутив необходимо загрузить с официального сайта [2], а затем разместить все файлы системы в желаемой папке. После этого останется только создать текстовой файл конфигурации, в котором прописать путь к папке, где вы хотите хранить свои базы[2]. Как это делается, подробно описано в [3] и [4], так что позвольте здесь не повторяться.

Не забывайте, что для работы с системой необходимо сначала запустить программу-сервер **mingod** и только потом программу-клиент **mongo**.

Надеюсь, что у вас, как и у меня, все заработает с первого раза.

2. Пример “Мои компьютеры”

2.1. Постановка задачи

Удачный выбор учебной задачи играет очень большую роль. Сошлюсь на забавный пример, неизменно кочующий из одной статьи про моделирование структуры знаний в другую, — это классификация вин. Речь там идет о том, какие сорта вин лучше сочетаются с устрицами, и прочих весьма специфических вопросах. Впервые такая предметная область была выбрана в известном отчете [5] (замечу, что на сайте можно найти и его русский перевод). И хотя в настоящее время без ссылок на этот первоисточник не обходится практически ни одна работа по онтологиям, думается, для нашего обсуждения стоит попробовать поискать что-нибудь менее изысканное.

Чтобы найти более естественный пример, я просто посмотрел вокруг себя и тут же увидел, что искал. У нас дома имеется три достаточно разных по конструкции компьютера: это настольный компьютер, довольно старый ноутбук и недавно приобретенный ради поездок маленький нетбук, весом менее 1 кг. Почему бы не занести сведения о них в базу? Подчеркну, что для нашего разговора о компьютерном представлении знаний очень полезно, что все компьютеры так по-разному скомпонованы. В частности, у настольного компьютера монитор и клавиатура являются отдельными устройствами, а у остальных (переносных) они по понятным причинам встроены в общий корпус. Напротив, у “десктопа” есть встроенный накопитель на гибких магнитных дисках, а к ноутбуку аналогичный дисковод прилагался в качестве внешнего USB-устройства. Ну а нетбук и вовсе “все свое несет с собой” — к нему при продаже не было никаких внешних устройств[3].

Итак, в самом общем виде задача уже может быть сформулирована: пользуясь MongoDB, создать базу сведений об имеющихся трех компьютерах.

Известно, что при выборе или описании компьютера мы пользуемся основными характеристиками его функциональных узлов: процессора, памяти и устройств ввода-вывода. Подобных характеристик довольно много, поэтому, стараясь сохранить лишь те, что отличаются наибольшим разнообразием, я после некоторых размышлений построил табл. 1.

Кратко поясню свой выбор. Процессор я взял в качестве элемента, который всегда располагается внутри корпуса компьютера. Аналогичным образом в корпусе любого из трех моих компьютеров есть ОЗУ и жесткий диск, поэтому с конструктивной точки зрения здесь нет принципиальных особенностей, и я их в таблицу не включил. Так же я поступил и с накопителями на оптических дисках — они если были, то все были встроенными. Напротив, я сохранил сведения о мониторе и дисководе по причинам, о которых уже говорилось ранее. А клавиатуру, блок питания, мышь и тачпад можно, опять-таки без особого ущерба, пропустить. Наконец, в качестве примера “редкого” узла я выбрал адаптер WiFi — он есть всего у одного из моих компьютеров.

Мне кажется, что данные, приведенные в таблице, понятны всем читателям журнала. Пожалуй, поясню только столбец с сокращенным (ради приемлемой ширины таблицы) названием “внут./внеш.”, в котором отмечено, является ли данный функциональный узел встроенным в корпус или нет. Ячейки таблицы, соответствующие внешним устройствам, для наглядности закрашены, а знак вопроса в последней строке отражает некоторую неопределенность ситуации: хотя в комплекте нетбука и нет дисковода, но что мне мешает воспользоваться дисководом от ноутбука? Об этой особенности ситуации мы поговорим позднее (см. 2.4), а сейчас перейдем к конструированию базы данных согласно составленной таблице.

[2] Убедитесь, что на рабочем диске свободно минимум 100 Мб — даже очень небольшие базы Mongo сразу резервируют много места.

[3] Ах да, блок зарядки все-таки дали!

2.2. Создаем базу из полных записей

Внимательно проанализируем полученную выше таблицу с параметрами компьютеров.

Сразу договоримся упростить представление признака внешних/внутренних устройств за счет возможности Mongo “вкладывать” данные друг в друга. Предлагается ввести поле под названием box (ящик, корпус) и сделать его значение массивом; тогда все внутренние устройства станут элементами массива, а все внешние будут размещаться вне этого поля-массива. Предложенная мера позволит исключить из базы целых четыре поля, сохранив при этом их информацию.

Теперь можно начать формировать записи о каждом из трех имеющихся компьютерах.

Добавление записи к Mongo имеет вид вызова метода `insert` [3, 4]:

```
db.comps.insert()
```

Здесь db — это формальное обозначение текущей базы данных, оно всегда добавляется к каждой команде. Стоящее после точки comps — это имя нашей коллекции данных (я выбрал его сам, несколько сократив для скорости набора слово во множественном числе “computers”). Наконец, insert() — это метод, в скобки которого нам предстоит вставить содержимое конкретной записи. Замечу, что нотация с точками явно заимствована из ООП.

Начнем с первого компьютера. Его описание на JSON, четко структурированное с применением пробелов, будет выглядеть следующим образом:

```
db.comps.insert({name: ‚desktop',
                 units:
                   [{box:
                       [{processor:
                           {name: ‚AMD Sempron',
                            f: 2.5
                           }
                        }
                      , {floppy_drive:
                           {d: 3.5
                           }
                        }
                       ]
                    }
                  , {monitor:
                       {size: 17
                       }
                    }
                   ]
                });
```

Внимательно изучите приведенный выше текст. Как он устроен? Имеется объект, у которого есть всего два крупных поля: name (имя) и units (блоки, узлы). Содержание первого из них очевидно. Второе является массивом, так как его значение написано в квадратных скобках; в нашем случае массив содержит два внутренних поля с понятными именами: box и monitor (последний, как нетрудно видеть, 17 дюймов). Каждое из этих полей в листинге обведено рамкой. Что касается поля box, то оно, в свою очередь, тоже массив, в который входят processor и floppy_drive со внутренними параметрами устройств.

Прежде чем включать компьютер[4], давайте подготовим две оставшиеся записи.

```
db.comps.insert({name: 'notebook',
                 units:
                   [{box:
                       [{processor:
                           {name: 'Intel Celeron',
                            f: 2.4
                           }
                        }
                      , {monitor:
                           {size: 14
                           }
                        }
                       ]
                    }
```

[4] Я бы вообще посоветовал сначала прочитать разделы 2 и 3 целиком: возможно, вы выберете для себя какой-нибудь другой способ ввода данных.

```
                    , {floppy_drive:
                          {d: 3.5
                          }
                      }
                    ]
                 });

db.comps.insert({name: 'netbook',
                 units:
                    [{box:
                        [{processor:
                            {name: 'Intel Atom',
                             f: 1.6
                            }
                          }
                        , {monitor:
                            {size: 10
                            }
                          }
                        , {wifi:
                            {card: 'Intel link 5150'
                            }
                          }
                          ]
                      }
                    ]

                 });
```

Надеюсь, вам удастся разобраться в них самостоятельно.

Теперь остается запустить Mongo (сервер и клиент!) и, создав новую базу командой

```
use my_computers
```

ввести приготовленную информацию обо всех компьютерах.

При вводе обратите внимание на предусмотренную помощь в проверке правильности расстановки скобок: при установке курсора под скобку клиент подсвечивает парную ей.

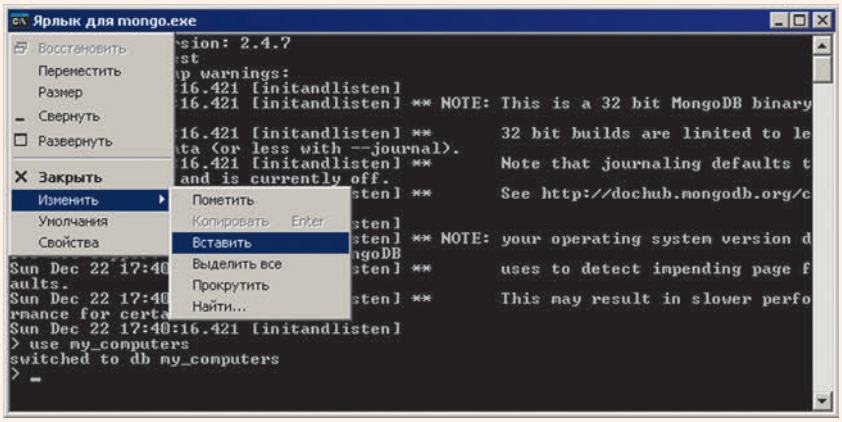


Рис. 2. Вставка текста из буфера обмена в консольное окно

Как теперь проверить, насколько удачно сформировалась база?

Можно, например, попросить найти (метод find) компьютер по размеру его монитора. Соответствующие запросы будут выглядеть так.

```
db.comps.find({'units.monitor.size': 17})
db.comps.find({'units.box.monitor.size': 14})
db.comps.find({'units.box.monitor.size': 10})
```

Обратите внимание, как прописаны “пути” к монитору у разных компьютеров: у стационарного монитор является самостоятельным блоком, а у портативных компьютеров — частью поля box (т.е. располагаются внутри корпуса).

Можно дополнительно поинтересоваться, у каких компьютеров поле monitor существует ($exists) вне корпуса, а у каких — внутри.

```
db.comps.find({'units.monitor': {$exists: true}})
db.comps.find({'units.box.monitor': {$exists: true}})
```

Примеры запросов по мониторам, превышающим заданный размер, можно найти в файле my_computers.txt в приложении на диске. Напомню читателям, что текстовые строки из этого или других файлов можно перенести в клиент Mongo через буфер обмена, воспользовавшись главным меню в левом верхнем углу окна (*рис.* 2).

2.3. Создаем базу, дополняя содержание записей

Первые эксперименты с набором команд в Mongo почти наверняка изрядно утомят вас из-за необходимости тщательно расставлять многочисленные скобки. Поэтому, возможно, вас заинтересует альтернативный способ создания записей о наших компьютерах. Его суть сводится к тому, что сначала в записи создаются "пустые" объекты самого верхнего уровня, а затем в них вставляется содержимое. В результате команд получается больше, но зато они короче и имеют более простую структуру.

Рассмотрим данный способ на примере первого компьютера с именем desktop. Чтобы не портить созданную ранее коллекцию comps, будем работать с новой, допустим, comps2. Прежде всего создадим запись, в которой есть только имя и пустой пока массив units (одни скобки [] безо всякого содержимого) для хранения информации о блоках:

```
db.comps2.insert({name: 'desktop',
units: []});
```

Теперь с помощью метода update() найдем запись с именем desktop и вставим ($push) в массив units в качестве элементов коробку системного блока (box) и монитор.

```
db.comps2.update({name: 'desktop'},
{$push: {units: {box: [],
monitor: {size: 17}}}});
```

Наконец, пополним box процессором и дисководом. Адресоваться внутрь массива units теперь несколько сложнее, поскольку он уже не пустой, а содержит два элемента: units[0]=box[] и units[1]=monitor. При этом по принятому в Mongo несколько странному синтаксису[5] вместо units[0] принято писать units.0. В итоге полное обращение к массиву box принимает вид units.0.box, а добавление в него внутренних компонентов выглядит так:

```
db.comps2.update({name: 'desktop'},
{$push: {'units.0.box': {processor:
{name: 'AMD Sempron', f: 2.5}}}});
db.comps2.update({name: 'desktop'},
{$push: {'units.0.box':
{floppy_drive: {d: 2.5}}}});
```

Используя метод find(), как уже описывалось в 2.2, можно убедиться, что итоговый результат получился такой же, как и раньше.

2.4. Ссылки на внешний дисковод

А теперь вспомним, что у нас в комплекте одного из компьютеров был внешний USB-дисковод. Для простоты мы включили его в состав ноутбука, что, видимо, не самое лучшее решение; в самом деле, а как, скажем, нетбук "узнает", что можно воспользоваться этим внешним дисководом?

Попробуем организовать базу по-другому. Снова предлагаю экспериментировать с новой коллекцией; назовем ее comps3.

Сначала добавим к пустой пока коллекции дисковод:

```
db.comps3.insert({floppy_drive:{d:
3.5}})
db.comps3.find()
```

На вторую команду Mongo ответит:

```
{ "_id" : ObjectId("52b0672a71848fef7ba
0481a"), "floppy_drive" : { "d" : 3.5 } }
```

Тем самым, мы видим, что объект создан. Но главное для нас сейчас не это, а то, что Mongo сообщила нам значение идентификатора _id; это "длинное" шестнадцатеричное число (вполне вероятно, что у вас получится не такое значение, возьмите, естественно, свое!) надо скопировать и подставить в поле external (выделено жирным шрифтом):

```
db.comps3.insert({name: 'netbook',
units: [{box: [{processor:
{name: 'Intel Atom', f: 1.6} },
{monitor: {size: 10}},
{wifi: {card: 'Intel link 5150'}}]},
{external:
ObjectId("52b0672a71848fef7ba0481a")}]});
```

Так что теперь к прежнему описанию компьютера добавилось еще одно (внешнее) устройство. Попробуем спросить у системы, какое оно. Один из возможных вариантов решения этой проблемы состоит в следующем. Сначала поинтересуемся, у каких компьютеров есть поле external (т.е. существуют внешние устройства):

```
db.comps3.findOne({'units.external':
{$exists: true}}).units[1].external
```

Вы, вероятно, заметили, что вместо ставшего уже привычным метода find() в запросе использован другой — findOne(). Во вводных описаниях [3,4] о нем вообще не упоминается, а в прилагаемом к дистрибутиву руководстве [6] сказано, что этот метод всегда находит только один (one) документ — тот, который встретился первым. Но, на мой взгляд, гораздо важнее, что, пользуясь findOne(), легко получить не только всю найденную запись целиком, но и ее отдельное поле (удивительно, почему в документации такая полезная возможность нигде не подчеркивается; мы еще вернемся к ее полезности в разделе 4). Какое именно поле требуется выделить, пишется после обращения к методу[6]: в нашем случае это units[1].external.

В результате обработки нашего запроса на экране увидим значение:

```
ObjectId("52b0672a71848fef7ba0481a")
```

[5] Запись через точки уже "занята" для объектов; более того, такая же запись используется еще для последовательного применения методов, например, find().sort().

[6] Перед нами еще один пример, когда в запросе Mongo неожиданно "вылезает" синтаксис JavaScript: индекс написан в квадратных скобках!

Это не что иное, как искомый идентификатор внешнего устройства.

Теперь подставим все наше “удачное” выражение, извлекшее идентификатор, в новый запрос (там оно выделено жирным шрифтом):

```
db.comps3.find({_id: db.comps3.
findOne({'units.external':
{$exists: true}}).units[1].external})
```

Фактически вместо значения идентификатора в последнюю команду подставлен запрос, его получающий. Таким образом, мы имеем дело с вложенным запросом. Но он работает, показывая интересующий нас внешний дисковод:

```
{ "_id" :
ObjectId("52b0672a71848fef7ba0481a"),
"floppy_drive" : { "d" : 3.5 } }
```

Итак, мы убедились, что к нашему компьютеру можно вместо внешнего устройства подсоединить дисковод.

Тот же результат можно получить, используя переменные языка JavaScript. Этот вариант решения приведен в приложении в текстовом файле comps3.txt; возможно, кому-то из читателей он покажется понятнее, чем вложенный запрос.

2.5. Заглянем внутрь базы

Mongo содержит в своем составе специальную утилиту **mongoexport**, которая позволяет просмотреть содержимое любой интересующей нас коллекции. Сразу подчеркну, что она запускается *вместо* клиента и взаимодействует с работающим сервером **mongod**. Утилита способна преобразовать данные указанной коллекции из внутреннего бинарного представления (BSON) в текст JSON. Например, чтобы узнать, что есть в нашей коллекции comps2, надо написать команду

bin\mongoexport --db my_computers --collection comps2 --out file.json

В командной строке, кроме имени исполняемого файла с двойным знаком минус, пишутся ключи, задающие дополнительную информацию. О содержании первых двух вы, несомненно, догадались, а последний определяет имя файла, в который будет сохранен результат экспорта. Вместо --out можно указать другую коллекцию, тогда она пополнится из первой; если же параметр вообще не задан, результаты выводятся на экран.

На всякий случай приведу краткую инструкцию, как получить содержимое коллекции.

- Запускаем сервер **mongod**.
- Через кнопку “Пуск” в главном меню Windows выбираем пункт “Выполнить...” и вводим cmd — при нажатии Enter будет запущен сеанс MS DOS.
- Устанавливаем текущий каталог в папке Mongo: я набирал команду cd c:\mongodb, возможно, вы используете другой путь.
- Вводим команду экспорта (см. выше).
- Загружаем файл, который должен получиться в папке Mongo, в **Блокнот** и убеждаемся, что там действительно хранится полное содержимое нашей коллекции.

Анализируя содержимое получившегося файла, обратите внимание, как необычно Mongo хранит внутри базы ObjectID().

3. Инструменты

Вдоволь намучившись со скобками при наборе команд Mongo, поневоле начинаешь задумываться, нельзя ли это дело как-то автоматизировать. Наверно, если немного подождать, то какие-нибудь удобные интерфейсные оболочки для Mongo обязательно появятся[7]. Можно, с другой стороны, уже сейчас попробовать написать нечто подобное на “си-образных” языках, для которых хорошо “налажена” связь с Mongo, — но это потребует от нас глубоких знаний и длительной работы. Видимо, и здесь, как обычно, истина где-то посредине. Поэтому вашему вниманию предлагается два простых инструмента, написанных в системе визуального программирования Delphi. Первый из них прост настолько, что, по-моему, доступен для повторения даже школьнику с весьма поверхностными навыками работы в Delphi. Второй более сложен: он опирается на мощный компонент TTreeView, но зато очень удобен в употреблении.

Оба инструмента являются автономными приложениями с графическим интерфейсом. В ходе диалога с пользователем они вырабатывают текстовую строку с командой для Mongo и отправляют ее в буфер обмена. Вставить текст из буфера в клиент Mongo уже не составляет труда (вспомните *рис.* 2). Выбранная технология хороша тем, что не требует прямого контакта с Mongo, а значит, не нужны какие-либо дополнительные знания по программированию, связанные с особенностями Mongo.

3.1. Mongo Wrapper Tool (MWT)

Данный инструмент (tool) позволяет автоматически создавать вокруг названий полей и их значений “обертку” (wrapper) из служебных символов. В итоге формируется командная строка для Mongo, в которой уже не надо, например, проверять сбалансированность скобок. Тем не менее, поскольку точки вставки для “обертываемых” текстов указывает человек, ошибки все же возможны; так что MWT правильнее назвать полуавтоматическим диалоговым инструментом.

Вид диалогового окна MWT приведен на *рис.* 3. Его рассмотрение, как и ввод команды, надо на-

[7] Например, на сайте *json.org* есть ссылки на три библиотеки для работы с JSON в Delphi — осталось дождаться умельца, который “приделает” к ним интерфейс.

чинать снизу вверх. Самая последняя строчка электронной формы создает по правилам JSON пару из имени поля и его значения. Между ними автоматически вставляется двоеточие с пробелом; в случае, когда флажок проверки ‘текст’ выбран, значение автоматически заключается в апострофы. Считывание значений и формирование пары запускается кнопкой “ввод”, расположенной в конце рассматриваемого ряда органов управления.

Обработчик кнопки “ввод” несложен и устроен следующим образом.

```
s := param.Text;
if textCB.Checked
   then s := ''''+s+''''; {заключаем в '}
s := field.Text+': '+s; {формируем пару}
pair.Text := s;
```

Здесь даже нечего особенно комментировать, кроме, может быть, редко встречающейся комбинации из четырех апострофов подряд. Крайние из них обычным образом заключают в себе текст, а два внутренних, по принятому правилу, говорят, что это собственно символ “апостроф”. Таким образом, здесь так специфически записан “апостроф внутри апострофов”, т.е., проще говоря, символ “апостроф”. Кому такая запись кажется слишком мудреной, может заменить ее числовым кодом апострофа, написав chr(39).

Результатом работы приведенного обработчика служит текст, который заносится в поле ввода pair, расположенное в строке выше. Далее пользователю предлагается вставить сформированную пару в текст одним из трех способов: заключив его в фигурные скобки, приписав после запятой или приписать после запятой в скобках (см. *рис.* 3). Вставка производится в то место поля ввода stringEdit (снова ищите его строкой выше), где установлен курсор. Поскольку вставка текста во всех трех случаях различается лишь символами, добавляемыми до и после текста, имеет смысл сделать общую процедуру.

```
procedure form_text(s1,s2:string);
{формирует и вставляет в текущую
 позицию stringEdit
 текст s1+pair+s2}
var spos:integer; s:string;
begin with Form1.stringEdit do
          begin
            spos := SelStart;
            {позиция курсора}
            s := Text;
            insert(s1+Form1.pair.
                   Text+s2,s,spos+1);
            Text := s
          end;
end;
```

Здесь в переменную spos запоминается текущая позиция курсора, а затем в содержимое строки вставляется текст пары, “обернутый” нужными служебными знаками. Заметим, что прибавление единицы к номеру позиции необходимо потому, что в компоненте TEdit символы нумеруются с 0, а в строковых переменных — с 1.

Чтобы вызвать процедуру, например, для кнопки “{вставить}”, надо обратиться к ней так:

```
form_text('{','}')
```

После того когда в строку будут вставлены все необходимые пары, можно нажать кнопку “Готово”. При этом будет сформирован полный текст команды для коллекции с указанным именем; затем он будет выделен и отправлен в буфер обмена.

```
s := stringEdit.Text;
s := 'db.'+collection.Text+'.'
   +comboBox1.Items[comboBox1.
                    Itemindex]
   +s+');';
resEdit.Text := s; {полный текст}
{заносим в буфер обмена
 (clipboard)}
resEdit.SelectAll;
resEdit.CopyToClipboard;
```

Полный проект MWT находится в приложении на диске. Кроме программ-обработчиков, описанных в тексте статьи, вы дополнительно найдете там возможность сохранения истории набора, что позволяет использовать ранее введенные тексты.

Для тех, кому освоение работы с инструментом покажется трудным, в файле “тест MWT.doc” детально показано, как получить строку, которая изображена на *рис.* 3.

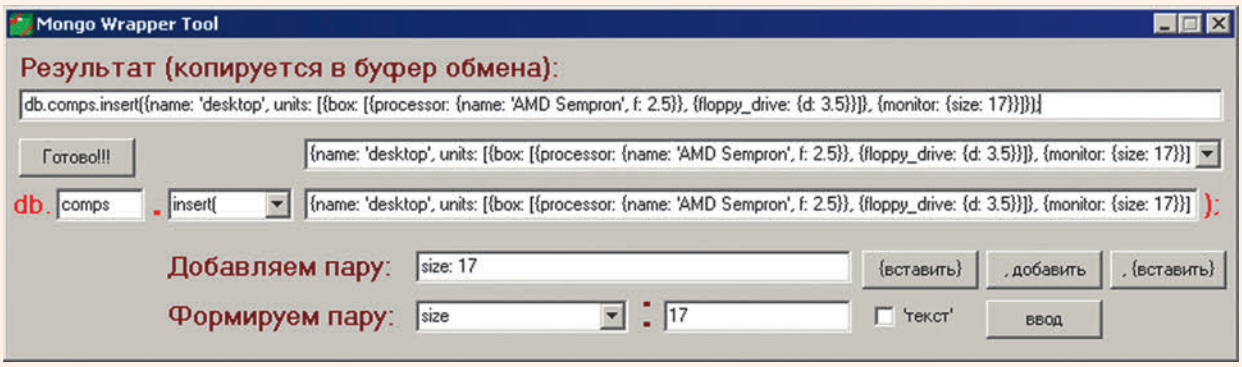


Рис. 3. Общий вид инструмента Mongo Wrapper Tool

3.2. Mongo Tree Tool (MTT)

Простейший инструмент MWT, который мы изготовили, уже заметно облегчает формирование синтаксически правильных текстов команд для MongoDB. Тем не менее работать с ним недостаточно комфортно: требуется постоянное внимание, да и ошибки не исключены. Попробуем сделать инструмент получше, такой, чтобы текст командной строки формировался полностью автоматически.

Как наглядно и просто изобразить данные о наших компьютерах? Поскольку их устройства иерархически вложены (box в units, processor в box и т.п.), то такая структура данных есть дерево (несколько забегая вперед, можно заглянуть на *рис.* 4). Отсюда следует очевидный план работы нового инструмента: а) создать визуализированную структуру дерева, а затем б) описать полученную структуру в формате JSON. В итоге мы имеем две относительно независимых подзадачи, которые будем решать последовательно.

Улучшенный инструмент, к созданию которого мы приступаем, уместно назвать MTT — ***M**ongo **T**ree **T**ool* (*tree* по-английски "дерево").

Наша задача существенно облегчается тем, что палитра компонентов Delphi на странице Win32 содержит замечательный готовый к употреблению компонент TreeView (его описание можно почитать, например, в [7][8]), в котором предусмотрена масса удобных методов работы с деревом и совершенные средства его рисования. На его базе мы и будем разрабатывать свой инструмент MTT.

Начнем с построения визуального образа дерева на экране. Посмотрим, для примера, на структуру, описывающую знания о нашем компьютере с именем notebook (*рис.* 4). Видно, что дерево состоит из отдельных элементов, причем каждому типу элемента соответствует определенный (назначенный нами) графический значок.

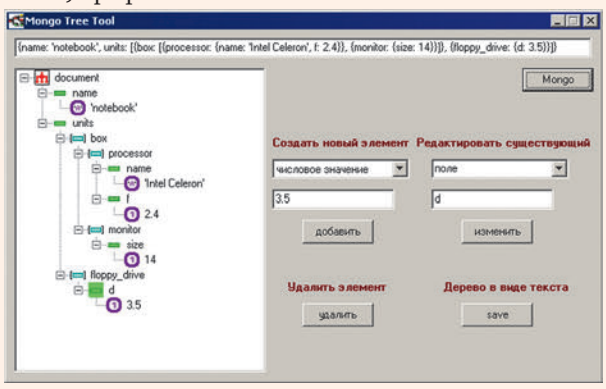


Рис. 4. Общий вид инструмента Mongo Tree Tool

Всего мы видим на рисунке пять разновидностей значков: документ (обязательный корень дерева, всегда существующий в единственном экземпляре), простое поле, поле-массив, текстовое и числовое значения. Помимо этих "иконок", нам потребуются еще пять — они будут изображать каждый из перечисленных элементов в выделенном (выбранном) состоянии. В нашем MTT выделенный элемент отображается на зеленом фоне — найдите на рисунке элемент d в нижней части дерева. В дальнейшем вы почувствуете, что выделение текущего элемента при работе с программой очень удобно, поскольку легко понять, с каким из узлов мы сейчас работаем (в частности, на *рис.* 4 мы сразу видим, что новый числовой элемент со значением 3.5 был добавлен именно к полю d).

Компонент TreeView "не умеет" сам хранить свои рисунки. Для этой цели с помощью свойства images присоединим к нему дополнительный компонент ImageList. В последний мы и загрузим все подготовленные заранее (в простом графическом формате, например, BMP) рисунки. Диалоговое окно нашей "картинной галереи" изображено на *рис.* 5. Обязательно обратите внимание на тот факт, что номера рисунков начинаются с нуля: это пригодится нам при установке нужного номера образа. И еще одно заслуживающее внимания обстоятельство: значки выделенных элементов должны идти строго в том же самом порядке, что и соответствующие им невыделенные. Такое расположение рисунков позволит, зная номер невыделенного образа i, вычислить для него номер выделенного образа по элементарной формуле i+5.

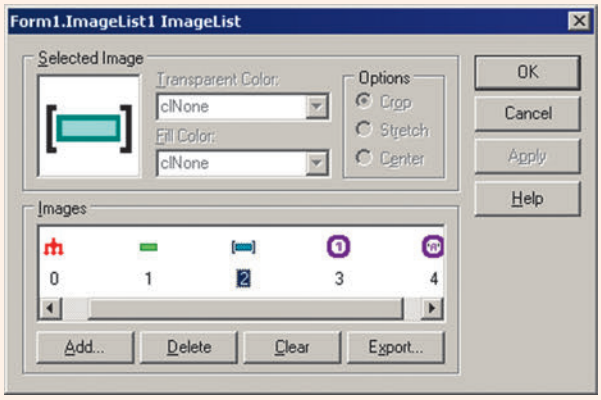


Рис. 5. Задаем картинки для элементов дерева

Наконец, не забудьте в свойство items, пользуясь стандартным диалоговым окном, добавить корень дерева с именем document и номерами образов ImageIndex=0 и SelectedIndex=5.

После этого создадим на правой половине формы необходимые стандартные компоненты ввода (снова возвращаемся к *рис.* 4) и перейдем к написанию обработчиков для них.

Начнем с добавления к дереву (создания) нового элемента. Для этого служат визуальные компоненты ComboBox1 (тип поля), Edit1 (значение поля) и кнопка Button1 ("добавить"), обработчик которой и добавляет к выделенному узлу дерева описанный вами элемент. Проследим подробнее, как это происходит.

[8] Очень полезно также посмотреть примеры работы с компонентом в справочной системе Delphi.

```
{считываем значение}
      s := edit1.Text;
{текстовое значение заключаем в ''}
      if comboBox1.ItemIndex = 3
         then s := ''''+s+'''';
{выделенный узел}
      tn := treeView1.Selected;
      {создаем к нему потомка}
      tn := treeView1.Items.
      AddChild(tn,s);
{устанавливаем картинку согласно
 выбранному пункту списка}
      {номер}
      i := comboBox1.ItemIndex+1;
      tn.ImageIndex := i;
      tn.SelectedIndex := i + 5;
      treeView1.FullExpand
```

Сначала считываем в строковую переменную s введенное пользователем значение (на *рис.* 4 это 3.5). Если в ComboBox1 выбран четвертый (ItemIndex=3) тип элемента, т.е. текстовое значение, то значение переменной заключается в апострофы. Далее в переменную tn[9] запоминается выделенный узел, и функция AddChild() добавляет к нему новый узел с текстом из переменной s. Функция устроена так, что возвращает ссылку на вновь созданный узел, так что после вставки наша переменная tn уже будет содержать новый узел. Затем по выбранному типу элемента ему ставится в соответствие номер невыделенного (ImageIndex) и выделенного (SelectedIndex) образов и дерево полностью раскрывается методом FullExpand() (слово “expand” имеет значения расширить, растянуть, расправить, раскинуться, распускаться и т.п.).

Аналогичным образом работает и редактирование уже существующего компонента. При каждом выделении элемента дерева запускается обработчик OnChange, и значения ComboBox2 и Edit2 устанавливаются в соответствии с типом и названием выбранного элемента. Значения эти можно поменять и, нажав кнопку Button2, “изменить”, занести исправления в дерево. Мы не будем здесь приводить программы этих двух простых обработчиков, тем более что обработчик для Button2 очень похож на таковой для Button1. Желающие могут заглянуть в приложение на диске и изучить соответствующие программные фрагменты.

А вот обработчик удаления компонента дерева, несмотря на его простоту, выписать стоит. Дело в том, что в книгах охотно описывается, как добавить узел, но, как правило, умалчивается, как его удалить. Между тем сделать это несложно:

```
{выделенный узел}
      tn := treeView1.Selected;
{не корневой узел!}
      if tn.AbsoluteIndex <> 0
      then treeView1.Items.Delete(tn);
```

Условие на AbsoluteIndex не дает удалить корневой узел с нулевым номером, а с ним заодно и все прикрепленное к нему остальное дерево[10].

Последняя кнопка “save” не столько полезна практически, сколько служит для познавательных целей: она позволяет сохранить структуру дерева в текстовой файл, а затем еще раз обдумать, в какой последовательности хранятся узлы дерева. Мы не будем разбирать обработчик этой явно вспомогательной кнопки, тем более что программа сохранения дерева не содержит каких-либо особенностей по сравнению с сохранением текста из других компонентов вроде Memo.

Таким образом, первая подзадача реализована: мы можем создавать и редактировать любое дерево по нашему желанию. Прежде чем переходить ко второй подзадаче, насладитесь результатами! (Во всяком случае, я с удовольствием этим побаловался!)

А теперь займемся преобразованием созданного дерева в Mongo-команду (вы еще не забыли, зачем нам это дерево понадобилось?). Это будет более серьезная задача, чем предыдущая.

Итак, создаем кнопку “Mongo” и поле редактирования resEdit, в котором будет формироваться результат. Все, что мы будем писать далее, — это один большой обработчик срабатывания кнопки.

Для дальнейших действий надо знать, что все узлы дерева хранятся в специальном массиве items, причем они всегда упорядочены согласно визуальному изображению дерева. Как учит теория [8], если нет множественного наследования (проще говоря, пересечения ветвей дерева), то однозначное преобразование дерева в массив всегда возможно. Для нас это обстоятельство удобно тем, что вместо сложного движения “вверх-вниз” по ветвям дерева можно просто перебирать элементы линейного массива.

И еще несколько полезных фактов. У любого выбранного нами узла автоматически формируется массив item (не путать с items!), в котором хранятся все его узлы-наследники (дети, потомки). Их число можно узнать, посмотрев свойство count, а булевское свойство HasChildren отвечает на вопрос, есть ли у данного узла дети. Родитель для данного узла указан в свойстве parent.

Заметим, что все узлы дерева, имеющие общего родителя, принято называть *братьями*. В английском языке, в том числе и в помощи к Delphi, используется несколько иной термин *sibling*, который переводится “дети одних родителей”, или, проще говоря, “братья и сестры”.

Как обработать дерево и сформировать Mongo-запрос? Можно начать от корня дерева и построить алгоритм, который, используя рекурсию, постепенно добавит в итоговую строку все нижележащие узлы. Но мне показалось естественнее воспользо-

[9] Узлы дерева имеют специальный тип TTreeNode (*node* по-английски это и есть узел).

[10] Это предусмотрительность, а не печальный опыт!

ваться “противоположным” алгоритмом, т.е. таким, который постепенно “сворачивает” дерево начиная с “концевых” узлов (в теории графов их часто называют *листьями*). Итак, логика алгоритма будет следующей. Перебираем узлы в поисках такого, ***все*** дети которого являются листьями дерева. Это означает, что ни у одного из детей уже нет потомков, а значит, все они уже приобрели окончательный вид и могут быть добавлены к родителю. Пользуясь введенными ранее терминами, можно сказать короче: *дети присоединяются к родителю только тогда, когда* ***все*** *их узлы-братья оказываются листьями дерева*.

После присоединения узлов-детей к родителю с добавлением необходимых скобок и прочих служебных символов обработанные узлы могут быть удалены. А сформированный узел-родитель теперь сам становится листом дерева и при последующем просмотре дерева, в свою очередь, тоже будет присоединен к своему родителю. Процесс завершится, когда останется единственный корневой узел.

Чтобы лучше понять, как это работает, обратитесь еще раз к дереву, изображенному на *рис.* 4. Очевидно, что при первом его обходе листьями будут только узлы-значения[11] и они будут добавлены к своим полям (через двоеточие). При повторном проходе к узлу processor будут подсоединены листья-поля name и f (конечно, с необходимыми внешними фигурными скобками и запятой между ними) и т.д. В конечном счете все узлы постепенно “свернутся” в корневой узел.

Поскольку портить отображаемое на экране дерево нельзя, создадим еще одно рабочее дерево d и сделаем его невидимым (установив visible = false). Именно в d и будут выполняться все преобразования.

Программа обработки дерева имеет следующий вид.

```
d.Items := treeView1.Items;
while d.Items[0].HasChildren do
begin
      i := 0; n := d.Items.Count; {число узлов}
      while i < n do with d do
        begin parentTN := Items[i];
              k := parentTN.Count; {количество детей у родителя}
              nado := k > 0; {когда детей 0 — точно не надо!}
              if nado then {есть дети, проверим, листья ли они}
                 begin child0 := parentTN.Item[0];
                 {наивно полагаем, что все дети одинакового типа!}
                       notValue := child0.ImageIndex < 3;
                       {3 и 4 — числовое и текстовое значения}
                       massiv := (child0.ImageIndex = 2) or
                                 (not notValue and (k > 1));
                      {т.е. либо указано, что это массив (тип 2),
                       либо это НЕСКОЛЬКО значений (типы 3 или 4)}
                      for j := 0 to k - 1 do {перебираем всех детей}
                          nado := nado AND
                      {нет детей} NOT parentTN.Item[j].HasChildren;
                 {nado = true только когда ни у кого нет детей!}
                 end;
              if nado {пора объединять!} then
                 begin s := parentTN.Text+': ';
                       if massiv then s := s+'[';  {array}
                       if notValue then s := s+'{'; {поле}
                       s := s + parentTN.Item[0].Text;
                       {добавили имя 1-го поля}
                       j := 1;
                       while j < k do {есть еще? добавим!}
                             begin if massiv and notValue
                                       then s := s+'}, {'
                                       else s := s+', ';
                                   s := s + parentTN.Item[j].Text;
                                   j := j + 1
                             end;
                       if notValue then s := s+'}'; {поле}
                       if massiv   then s := s+']';  {array}
```

[11] Это косвенно свидетельствует о правильности нашего дерева: все поля получат свои значения; в то же время, полный контроль корректности составленного дерева далеко не так тривиален (скажем, возможны пустые массивы без значений), поэтому в рамках данной статьи не рассматривается.

```
                        parentTN.Text := s; {итоговый текст}
                        {удаляем обработанное}
                        for j := 0 to k - 1 do
                            begin d.Items.Delete(parentTN.Item[0]);
                                  n := n - 1
                            end;
                    end
                else i := i + 1
        end
end;
        s := d.Items[0].Text;
        k := pos(' ',s); delete(s,1,k); {удаляем 'document: '}
        with ResEdit do
             begin Text := s;
                   SelectAll;
                   CopyToClipboard
             end
end;
```

Алгоритм начинается с копирования сформированного пользователем дерева в рабочее дерево d, с которым дальше и выполняются все манипуляции.

Внешний цикл организует итерационный просмотр и упрощение рабочего дерева до тех пор, пока оно не “выродится” в единственный узел (без потомков).

Следующий вложенный в него цикл по переменной i организует просмотр всех узлов дерева. Заметим, что число узлов n по мере обработки дерева будет уменьшаться.

Что делается с каждым из узлов?

Очередной узел запоминается в переменную parentTN, а в k заносится число его детей. Если k = 0, т.е. детей у узла нет, то переменная nado немедленно получает значение false; в этом случае оба следующих условных оператора пропускаются, мы наращиваем значение i и переходим к следующему узлу. Гораздо чаще дети имеются, тогда мы входим внутрь следующего if и проверяем, все ли узлы-братья являются листьями.

Но прежде формируются две логические переменные, которые потребуются при дальнейшем анализе (а именно, при расстановке квадратных и фигурных скобок). В переменную child0 запоминается первый из узлов-детей. Предполагается, что все его братья имеют тот же тип, так что здесь алгоритм “надеется” на аккуратность пользователя (желающие могут добавить соответствующие проверки и выдачу сообщения при обнаружении ошибки). Далее анализируется целочисленное свойство child0.ImageIndex, которое характеризует тип узла. В результате булевская переменная notValue принимает значение true, если текущий узел не является текстовым или числовым значением. Далее вывод о том, является ли конструкция массивом, заносится в булевскую переменную massiv: массив либо помечен явно (тип узла с номером 2), либо мы имеем дело[12] со значениями, которых несколько (k > 1).

Далее, наконец, определяется, можно ли свернуть узлы-братья к родителю. Для этого все они перебираются в цикле и проверяется, являются ли они листьями дерева (HasChildren = false). Цикл устроен так, что если хотя бы один из рассмотренных узлов имеет потомка, то (благодаря отрицанию перед логическим И) переменная nado получит значение false, что запретит последующие преобразования узла-родителя.

Пусть теперь в результате проверки выяснилось, что у узла-родителя все потомки являются листьями. Тогда nado = true, следующее условие “срабатывает”, и запускается большой программный кусок, который присоединяет содержимое всех листьев к родительскому узлу.

Работает он так. Сначала к тексту родителя формально приписывается двоеточие. Далее для массива добавляется квадратная скобка и, если узел-лист является не значением (а объектом!), то еще и фигурная скобка. После этого добавляется содержимое первого узла-листа. Затем, если есть еще узлы-листы, то они приписываются через запятую аналогичным образом и, наконец, закрываются нужные скобки. Теперь объединенный текст для узла-родителя готов и мы помещаем его в parentTN.text. Остается удалить все обработанные узлы-потомки. В этом цикле обратите внимание, что удаляется все время лист-элемент с индексом 0: как только самый первый узел удаляется, оставшиеся перенумеровываются, начиная с нуля, заново! Удаляя узел, важно не забывать уменьшать общее число узлов n.

Когда все дерево “свернется” в единственный корневой узел, цикл завершается. Остается только

[12] В наших примерах такая ситуация не встречается, но почему бы ее не предусмотреть?

удалить ненужное имя корневого узла “document: ” и отправить окончательную строку с командой в буфер обмена.

Таким образом, достаточно нажать кнопку “Mongo”, и строка с командой будет сформирована полностью автоматически! Имея такой мощный инструмент, можно рискнуть записать в базу параметры всех узлов компьютера, а не только те значения, которые были отобраны ранее в сокращенную таблицу. Проделайте это в качестве закрепляющего упражнения.

4. Пример “Устройство компьютера”

Помимо уже рассмотренных средств для представления знаний, в Mongo есть и некоторые более специализированные. В частности, в одном из файлов документации [6], который входит в дистрибутив, в пункте 3.3.2 обсуждается создание структур данных в виде деревьев. Как следует из описания, для создания иерархических структур в Mongo заложено пять(!) различных способов[13] решения этой важной задачи. Каждый из них имеет свои особенности и, соответственно, преимущества. Мы рассмотрим только один, который замечателен двумя достоинствами: во-первых, он наиболее компактный и наглядный, а во-вторых, что еще более важно, он “может также обеспечивать подходящее решение для хранения графов, где узел может иметь множественных родителей” [6]. Для нашей задачи последнее свойство окажется принципиальным.

4.1. Постановка задачи

Подробно обсуждавшийся ранее пример, сформулированный в 2.1, служит образцом сохранения некоторого набора частных фактов. С точки зрения представления знаний это простейшая задача. Теперь рассмотрим чуть более сложную проблему — как описать устройство “абстрактного” компьютера (а не трех моих конкретных).

Задача эта сама по себе заслуживает отдельного обсуждения [9, 10]. Причина состоит в том, что популярная и, к сожалению, учебная литература очень часто трактуют данный вопрос с чисто ***конструктивной*** точки зрения, называя характерными частями компьютера системный блок, клавиатуру, монитор и т.д.[14] Тем не менее серьезные авторы прямо подчеркивают, что “определение ликбезовских курсов, согласно которому компьютер состоит из системного блока, монитора, клавиатуры и манипулятора “мышь”, естественно, никуда не годится — под него не подходят даже портативные компьютеры или монтируемые в стойку серверы с IBM PC-совместимой архитектурой” [11].

[13] В руководстве используется термин *pattern* — образец, модель, шаблон.

[14] Такой подход имеет право на существование, например, для покупателя, которому важно знать, сколько коробок он должен получить в магазине.

При подготовке статьи, хотя это и было не обязательно, я решил посмотреть, изменилось ли что-нибудь в описании данного вопроса за последнее время, для чего обратился к Google. Лучше бы я этого не делал… В ответ на мой запрос честный поисковик нашел мне в авторитетной Википедии (в русской статье “Семантические сети”) знакомую фразу: “Например, компьютер состоит из системного блока, монитора, клавиатуры, мыши и т.д.”. Более того, когда я скопировал эту строку целиком, добавив для верности еще характерное начало следующего предложения “Важным отношением является HasPart…”, и заключил все это в кавычки, то, к своему ужасу, увидел следующую реакцию Google: “Результатов: примерно 363 (0,42 сек.)”[15]. Воистину компьютер многократно усиливает некомпетентность человека!

Справедливости ради отмечу, что попадались и интересные находки. Например, на страничке с красивым названием “Понимание компьютеров как систем” [12] изображена очень интересная схема, базирующаяся на информационных процессах, которые столь уважаемы сейчас в методике школьной информатики. Но даже из такой обобщенной схемы выводилась вполне пристойная структура компьютера безо всяких упоминаний о системном блоке.

Прошу прощения читателей за некоторое отступление, но без него будет непонятна схема, приводимая далее на *рис.* 6.

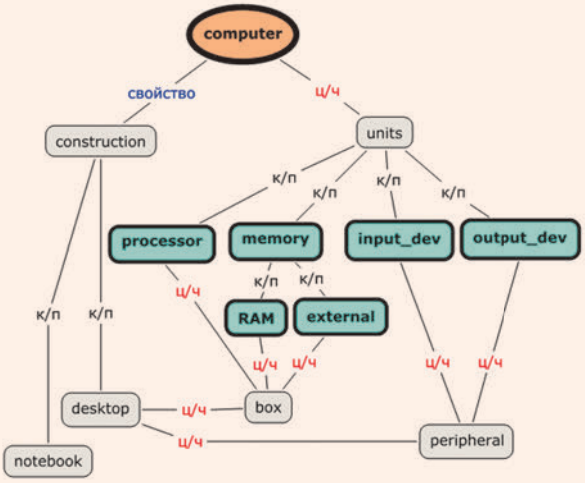


Рис. 6. Упрощенная семантическая сеть “Устройство компьютера”

На схеме изображено, что компьютер состоит из *функциональных узлов* (*units*), разновидностями которых являются *процессор*, *память* (для простоты только двух видов — *ОЗУ* и *внешняя*), а также *устройства ввода* и *вывода*. Кроме того, у

[15] Еще одно “приключение” состояло в том, что сия мудрость предоставлялась на одном из сайтов как цитата из Большой Советской Энциклопедии, хотя БСЭ, как оказалось, вообще не содержит статьи “Семантические сети”.

компьютера есть некоторое частное свойство, которое именуется *конструкцией*. Одной (и далеко не единственной!) из таких конструкций является *стационарный ПК* (*desktop*), который состоит из *корпуса* (*box*) и *отдельных устройств* (*peripheral*). Для конструкции обозначенного типа в корпус входят *процессор*, *ОЗУ* и *внешняя память*, остальное в системный блок не попадает (внешние дисководы сейчас не рассматриваем). На рисунке также выделена еще одна из возможных конструкций — *ноутбук*, которая устроена по-другому, но, чтобы не загромождать схему, эта структура не разворачивается.

Рисунок уже подготовлен к занесению информации в Mongo, поэтому все термины даются на английском языке (они будут идентификаторами), зато типы связей между ними записываются на русском (они будут текстовыми значениями). Заметим, что для удобства будущего набора названия стандартных связей "*часть/целое*" (подписаны красным цветом) и "*класс/подкласс*" (черные подписи) на *рис.* 6 сокращены.

Читатели, несомненно, поняли, что *рис.* 6 есть очень схематичное описание, которое нуждается в серьезном расширении. Определенно не хватает конкретных внешних устройств (они присоединятся с помощью отношения *класс/экземпляр*), нуждаются в доработке другие варианты конструкций, и т.д. и т.п. Но для целей данной публикации — научиться строить в Mongo простейшую семантическую сеть, имеющейся степени детализации вполне достаточно.

А теперь можно непосредственно приступать к формированию подготовленного графа в Mongo.

4.2. Граф без типов связей

Сначала опишем получившуюся структуру (граф, семантическую сеть) без учета типов связей. В основу описания положим строки вида

_id: "parent", children: ["child1", "child2", ...]

Здесь parent — это идентификатор узла-родителя, например, *memory*, а child1 и т.д. — узлы-потомки, например, для нашего случая, *RAM* и *external*. Важно отметить, что введение строковых (смысловых) идентификаторов вместо числовых в нашем случае явно удобнее, хотя это и снижает эффективность обработки данных в базе.

Описание графа, изображенного на *рис.* 6, пригодное для сохранения в Mongo, имеет следующий вид.

```
db.graph1.insert({_id: "RAM", children: []})
db.graph1.insert({_id: "external", children: []})
db.graph1.insert({_id: "processor", children: []})
db.graph1.insert({_id: "memory", children: ["RAM", "external"]})
db.graph1.insert({_id: "input_devices", children: []})
db.graph1.insert({_id: "output_devices", children: []})
db.graph1.insert({_id: "box", children: ["processor", "RAM", "external"]})
db.graph1.insert({_id: "peripheral", children: ["input_devices", "output_devices"]})
db.graph1.insert({_id: "desktop", children: ["box", "peripheral"]})
db.graph1.insert({_id: "construction", children: ["desktop"]})
db.graph1.insert({_id: "units", children: ["processor", "memory",
                        "input_devices", "output_devices"]})
db.graph1.insert({_id: "computer", children: ["units", "construction"]})
```

Здесь мы видим команды вставки (insert) в новую коллекцию graph1 узлов и связей между ними. Например, узлы *RAM* и *external* не имеют потомков, но сами они являются потомками узла *memory* (см. четвертую строку).

Подчеркнем, что список потомков (children) с формальной точки зрения является самым обычным массивом из текстовых значений, хотя эти значения и совпадают с выбранными ранее строковыми идентификаторами.

Судя "по отсутствию возражений" со стороны системы, данные приняты. Но хочется проверить более тщательно. Как это можно сделать?

Начнем с того, что убедимся в правильности нахождения потомков. Введем запрос

```
db.graph1.findOne({_id: "computer"}).
children
```

На который Mongo справедливо напишет

```
[ "units", "construction" ]
```

Обсудим подробнее, как получился такой ответ. В нашем запросе стоит метод findOne(), с которым мы уже встречались в 2.4. Механизм работы метода в руководстве [6], к сожалению, не раскрывается, но по логике можно догадаться, как все это примерно работает. Более того, как мы скоро увидим, правильность нашего понимания будет косвенно подтверждена последующими экспериментами в 4.3. Итак, по-видимому, метод findOne() сначала, как обычно, ищет первый попавшийся документ по заданному признаку (в нашем случае по идентификатору), после чего выделяет из него поле children, указанное после точки. Сразу хочу подчеркнуть, что понятие потомков, несмотря на его важность, не является чем-то исключительным — просто ищется поле указанного документа.

Метод проб и ошибок показывает, что findOne() позволяет выделить не только весь массив children,

но и любой его элемент, например, можно узнать второго потомка:

```
db.graph1.findOne({_id: "computer"}).
children[1]
construction
```

Для закрепления попробуйте подправить запрос и спросить потомков узла *unit* (их четверо) и *processor* (а здесь нет совсем!).

Теперь рассмотрим обратную задачу: для некоторого заданного узла определим его родителя. Например, для узла *input_devices* запрос будет выглядеть достаточно стандартно:

```
db.graph1.find({children: "input_devices"})
```

В ответ Mongo выдаст документы про обоих родителей *peripheral* и *units*:

```
{ "_id" : "peripheral", "children" :
[ "input_devices", "output_devices" ] }
{ "_id" : "units", "children" :
[ "processor", "memory",
"input_devices", "output_devices" ] }
```

Это, конечно, правильно — см. *рис.* 6[16]. Хочется подчеркнуть, что с позиций Mongo поиск родителя ничем не отличается от стандартного поиска документа по известному значению строкового поля. Не случайно в запросе используется обычный метод find().

Руководство [6] мудро обращает наше внимание на то, что ответ на данный запрос не только находит родителя для данного узла, но и содержит сведения о его узлах-братьях по каждому из родителей. В частности, по родителю *peripheral* узел *input_devices* имеет всего одного брата — *output_devices*, а по *units* дополнительно добавляются *processor* и *memory* (проверьте по *рис.* 6).

Таким образом, мы научились строить семантическую сеть и находить в ней потомков и предков.

4.3. Граф с учетом типов связей

А теперь добавим дополнительно к каждой связи ее тип. Именно такая ситуация часто встречается в базах знаний. Как видно из *рис.* 6, в нашей семантической сети использовано всего три типа связей: *целое/часть* (сокращенно ц/ч), *класс/подкласс* (к/п) и *свойство*.

Для решения задачи достаточно добавить еще одно поле — links, которое содержит перечень типов связей в виде массива. В большинстве случаев все типы одинаковые, так что особо внимания при наборе не требуется. Но для последнего узла *computer* типы различны, так что требуется проследить, чтобы их порядок соответствовал порядку узлов. Как обычно, будем использовать новую коллекцию graph2.

```
db.graph2.insert({_id: "RAM", children: []})
db.graph2.insert({_id: "external", children: []})
db.graph2.insert({_id: "processor", children: []})
db.graph2.insert({_id: "memory", children: ["RAM", "external"], links: ["к/п", "к/п"]})
db.graph2.insert({_id: "input_devices", children: []})
db.graph2.insert({_id: "output_devices", children: []})
db.graph2.insert({_id: "box", children: ["processor", "RAM", "external"],
                  links: ["ц/ч", "ц/ч", "ц/ч"]})
db.graph2.insert({_id: "peripheral", children: ["input_devices", "output_devices"],
                  links: ["ц/ч", "ц/ч"]})
db.graph2.insert({_id: "desktop", children: ["box", "peripheral"],
                  links: ["ц/ч", "ц/ч"]})
db.graph2.insert({_id: "construction", children: ["desktop"], links: ["к/п"]})
db.graph2.insert({_id: "units", children: ["processor", "memory", "input_devices",
                  "output_devices"], links: ["к/п", "к/п", "к/п", "к/п"]})
db.graph2.insert({_id: "computer", children: ["units", "construction"],
                  links: ["ц/ч", "свойство"]})
```

В качестве проверки можете для начала повторить запросы о потомках и предках (их образцы есть также на диске в файле graph.txt). А теперь посмотрим, как узнавать “соседей” по семантической сети с учетом типов связей.

Попробуем ввести запрос аналогично тому, как мы делали при поиске потомков, только вместо children в нем напишем links.

```
db.graph2.findOne({_id: "computer"}).links
```

Система отвечает предсказуемо, т.е. особой разницы между children и links она не видит:

```
[ "ц/ч", "свойство" ]
```

В заключение выполним более сложный поиск: найдем потомков узла с заданным типом связи. Наиболее интересно сделать это для узла computer, у которого потомки имеют разный тип. Мне удалось найти решение только с использованием JavaScript.

[16] На *рис.* 6 ради экономии места слово “devices” везде написано сокращенно!

```
var c = db.graph2.findOne({_id:
"computer"}).children
var l = db.graph2.findOne({_id:
"computer"}).links
for (i = 0; i < 2; i++) {
if (l[i]=="свойство") {printjson(c[i])}}
```

Первые две строчки трудностей не вызывают: одна заносит в переменную c массив с узлами-потомками, а вторая в l — соответствующий массив с типами ссылок. Наконец, последняя строка перебирает в цикле `for` все элементы массива типов и в случае совпадения с заданным образцом при помощи printjson выводит из массива узлов элемент с тем же номером.

Замечу, что попытка подставить вместо числа 2 (фиксированной верхней границы цикла) автоматическое определение размерности массива l.length() мне почему-то не удалась. Интерпретатор, как следовало из сообщения об ошибке, упорно пытался применять метод length() не ко всей переменной, а к ее первому элементу. С большой вероятностью это недоработка самого интерпретатора.

Подведем итоги.

• База данных Mongo имеет определенные преимущества по сравнению с общепринятыми реляционными БД. Ее принципы устройства лучше подходят для представления множества *разнообразных* фактов предметной области.

• Mongo основана на простой и наглядной системе записи JSON и тесно интегрирована с языком JavaScript. Порой последнее обстоятельство приводит к синтаксической путанице (например, иногда тексты помещаются в апострофы, а иногда в кавычки, индексы могут обозначаться где x[0], а где x.0 и т.д.).

• Mongo хорошо передает вложенность данных, позволяя, например, элементарно описывать, что из чего состоит. Чем больше уровень вложенности, тем ощутимее данное достоинство (но тем, соответственно, сложнее сохранить синтаксическую правильность записи).

• Mongo обладает мощными механизмами произвольного объединения (связывания) простых данных в более сложные через _id. Добавим, что подобные ссылки могут быть организованы не только внутри одной коллекции, как это показано в статье, но и между разными коллекциями и даже БД.

• Mongo прекрасно приспособлена к представлению строго иерархических структур данных (деревьев): предусмотрено пять(!) “штатных” способов такого представления. Некоторые из них поддерживают и сетевые структуры.

• Стандартные возможности системы позволяют к связям между данными элементарно добавлять сведения об их типах. Такая способность Mongo дает основания смотреть на эту систему как на конкурента (по крайней мере потенциального) систем для хранения семантических карт предметных областей вроде Topic Map или Concept Map.

• Клиент Mongo хорошо взаимодействует с буфером обмена, что позволяет легко сохранять в текстовых файлах наиболее важные команды с целью их последующего использования.

• С помощью современных систем программирования можно легко написать удобные инструменты для облегчения взаимодействия с Mongo.

Ссылки

1. *Хокинс Д., Блейксли С.* Об интеллекте. М.: Издательский дом “Вильямс”, 2007.
2. *MongoDB. http://mongodb.org*.
3. *Сегуин К.* Маленькая книга о MongoDB. *http://jsman.ru/mongo-book/index.html*.
4. *Поляков К.Ю.* Нереляционные базы данных: практикум. Информатика, 2013, № 6, с. 34–40.
5. *Noy N.F., McGuinnnes D.L.* Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, 2001. *http://protege.stanford.edu/publications/ontology_development/ontology101.html*.
6. *MongoDB* Documentation. Release 2.4.7. MongoDB Documentation Project.
7. *Архангельский А.Я.* Программирование в Delphi 6. М.: ООО Бином-Пресс, 2004.
8. *Ахо А.В., Хопкрофт Д., Ульман Д.Д.* Структуры данных и алгоритмы. М.: Издательский дом “Вильямс”, 2007.
9. *Еремин Е.А.* Популярные лекции об устройстве компьютера. СПб.: BHV-Петербург, 2003.
10. *Еремин Е.А.* Функциональные блоки ЭВМ. Информатика, 2005, № 3.
11. *Иртегов Д.В.* Введение в операционные системы. СПб.: БХВ-Петербург, 2002.
12. Understanding Computers as Systems *http://www.salinas.k12.ca.us/washington/beta/technology/comp_system/system.html*.

Интерактивная доска, проектор или приставка?

MimioClassroom

Выбирайте интерактивное оборудование, которое нужно именно вам!

Интерактивная приставка
MimioTeach

Интерактивный короткофокусный проектор
MimioProjector

Есть интерактивная версия с возможностью одновременной работы и обычная версия без интерактивных функций

Интерактивная доска
MimioBoard

ME78 — диагональ 78”
ME87 — диагональ 87”

Приобретайте интерактивные устройства для своего класса, исходя из ваших педагогических потребностей, удачно дополняя тот набор оборудования, который уже есть у вас.

У вас уже есть:	Дополните следующим оборудованием:
Меловая доска	MimioTeach, обычный проектор MimioProjector и накладка FlexCover*
	Интерактивный проектор MimioProjector и накладка FlexCover*
Меловая доска и проектор	Приставка MimioTeach и накладка FlexCover*
Маркерная доска	Интерактивный проектор MimioProjector
	Приставка MimioTeach и обычный проектор MimioProjector
Маркерная доска и проектор	Приставка MimioTeach

Если в вашем классе есть только меловая доска и достаточно места для установки — рекомендуем обратить внимание на интерактивную доску MimioBoard.

** FlexCover — гибкий магнитный лист с антибликовой белой поверхностью и покрытием для рисования маркером*

Продажа оборудования, консультации и обучение:

http://www.mimioclass.ru

8 (800) 5555-33-0

Звонок по России бесплатный

ООО «Рене» — генеральный дистрибьютор Mimio в России

mimio
a better way to learn

СЕМИНАР

Круги Эйлера (диаграммы Эйлера — Венна)

Д.М. Златопольский,
Москва

Данная статья подготовлена в связи с публикацией в сентябрьском выпуске “В мир информатики” задачи “На каникулах”[1].

Напомним ее условие: “После летних каникул 2013 года учитель опросил ребят, кто из них ходил в театр, кино или цирк. Оказалось, что из 36 учеников двое не были ни в кино, ни в театре, ни в цирке. В кино побывали 25 человек, в театре — 11, в цирке — 17, и в кино и в театре — 6, и в кино и в цирке — 10, и в театре и в цирке — 4. Сколько человек побывали в театре, кино и цирке одновременно?”.

Обсудим методику решения задачи.

Задачи подобного типа удобно решать, используя так называемые “круги Эйлера”. Прежде чем подробно рассказывать о том, как именно это делать, рассмотрим и решим другую, более простую, задачу: “В классе 30 учеников. Все они являются читателями школьной и районной библиотек. Из них 20 ребят берут книги в школьной библиотеке, 15 — в районной. Сколько учеников не являются читателями школьной библиотеки?”.

В каком случае общее число учеников, пользующихся школьной библиотекой или районной библиотекой — 20 + 15 = 35, больше, чем число учащихся в классе? Так бывает, когда некоторые ученики берут книги и в одной библиотеке, и в другой. Сколько таких учеников в классе?

Пусть круг *Ш* изображает множество читателей школьной библиотеки, круг *Р* — районной. Тогда *ШР* — условное изображение множества читателей и районной, и школьной библиотек одновременно:

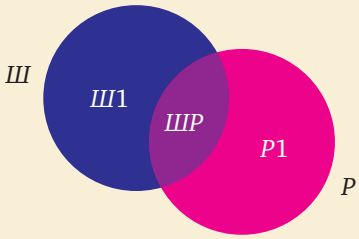


Рис. 1

Ясно также, что части кругов *Ш* и *Р*, обозначенные *Ш*1 и *Р*1, — это множество читателей, посещающих только школьную библиотеку и только районную библиотеку, соответственно.

Значение *ШР* может быть найдено так.

$Ш = 20$ человек, $Р = 15$ человек.

$$Ш1 = Ш - ШР = 20 - ШР \quad (1)$$

$$Р1 = Р - ШР = 15 - ШР \quad (2)$$

$Ш1 + ШР + Р1 = 30$ (общее число учеников в классе).

Подставляя вместо *Ш*1 и *Р*1 их значения из (1) и (2), получим:

$$20 - ШР + ШР + 15 - ШР = 30,$$

откуда $ШР = 5$, то есть пять учеников являются читателями школьной и районной библиотек одновременно.

Из *рис.* 1 следует, что число учеников, не являющихся читателями школьной библиотеки, равно

$$Р - ШР = 15 - 5 = 10.$$

Ответ: десять учеников не являются читателями школьной библиотеки.

Из приведенного решения мы получили две важные формулы:

$$Всего = A + B - AB \quad (3)$$

$$AB = A + B - Всего \quad (4)$$

— где *A* — количество (множество) людей или предметов, связанных с некоторым отдельным признаком *A*, *B* — то же, с признаком *B*, *AB* — количество людей или предметов, обладающих как признаком *A*, так и признаком *B*, *Всего* — общее число людей (или предметов) с признаками *A* или *B*.

Обсудим также вопрос — а могло ли в рассмотренной задаче быть так, что при известных значениях $Ш = 20$, $Р = 15$ и $ШР = 5$ общее число учени-

[1] Списки читателей, приславших правильные ответы на задачу, опубликованы в данном выпуске в рубрике “Задачник”.

ков в классе было равно 32? Да, могло. Определив по формуле (3) значение *Всего* (число ребят, посещающих школьную или районную библиотеки):

$$Всего = 15 + 20 - 5 = 30,$$

— мы можем сделать вывод, что 32 – 30 = 2 ученика класса не пользуются ни одной из указанных библиотек ☹.

При решении рассмотренной задачи *рис.* 1 играл существенную роль. В нем была удачно использована идея изображения множеств с помощью кругов. Эта идея полезна и при решении целого ряда других задач.

Выдающийся математик, физик, механик и астроном XVIII века Леонард Эйлер (он родился в Швейцарии, но много лет прожил в России) широко пользовался такими рисунками, и в его честь этот метод был назван “методом кругов Эйлера”. Такой метод математики применяли и до Эйлера. Им пользовался, например, выдающийся немецкий математик и философ Готфрид Вильгельм Лейбниц (1646–1716). В его черновых набросках были обнаружены рисунки с такими кругами. Но, как уже говорилось, достаточно основательно развил этот метод только Л.Эйлер. После Эйлера этот же метод разрабатывал чешский математик Бернард Больцано (1781–1848), только в отличие от Эйлера он рисовал не круглые, а прямоугольные схемы. Методом кругов Эйлера пользовался и немецкий математик Эрнст Шредер (1841–1902) в книге “Алгебра логики”. Но особенного расцвета графические методы достигли в сочинениях английского логика Джона Венна (1843–1923), подробно изложившего их в книге “Символическая логика”, изданной в Лондоне в 1881 году. Поэтому такие схемы также называют иногда “диаграммами Венна” или “диаграммами Эйлера — Венна”.

Можно сказать, что круги Эйлера (диаграммы Эйлера — Венна) — это наглядная геометрическая иллюстрация объемов понятий и отношений между ними.

Леонард Эйлер

Вернемся к “основной” задаче.

Прежде всего обратим внимание на то, что так как по условию из 36 учеников двое не были ни в кино, ни в театре, ни в цирке, то “где-то” побывали 34 человека.

Пусть круг *К* (*К*1, *КЦ*, *КТ*, *КТЦ*) изображает учеников, побывавших в кино, круг *Т* (*Т*1, *КТ*, *ТЦ*, *КТЦ*) — посетивших театр, круг *Ц* (*Ц*1, *КЦ*, *ТЦ*, *КТЦ*) — побывавших в цирке:

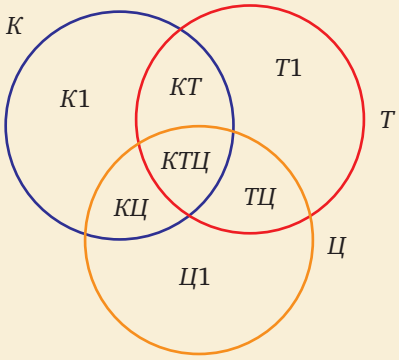


Рис. 2

Примечание. *К*1 — ученики, побывавшие на каникулах только в кино, *Т*1 — только в театре, *Ц*1 — только в цирке.

Рассмотрим два круга — *К* и *Т* (для экономии места отдельно вырисовывать их не будем). Согласно условию, *К* = 25 человек, *Т* = 11 человек, *КТ* + *КТЦ* = 6 человек (число ребят, посетивших два “места” — кино и театр).

Если вспомнить формулу (3), то можем сказать, что общее число учеников в кругах *К* (кино) и *Т* (театр) равно 25 + 11 – 6 = 30. Значит, только в цирке (*Ц*1) побывали 34 – 30 = 4 человека (вспомните тех, кто не попал во множество посещающих библиотеки).

Рассматривая круги *Т* и *Ц* и рассуждая аналогично, находим, что *К*1 = 34 – (11 + 17 – 4) = = 10 учеников (были только в кино).

Аналогично для кругов *К* и *Ц*:

$$Т1 = 34 - (25 + 17 - 10) = 2$$

(два ученика посетили только театр).

Далее, из *рис.* 2 следует, что количество ребят, посетивших не менее чем два “культурных учреждения”, равно:

$$34 - (К1 + Т1 + Ц1) = 34 - (10 + 2 + 4) = 18,$$

то есть:

$$КТ + КЦ + ТЦ + КТЦ = 18. \qquad (5)$$

С другой стороны, по условию, побывали:

— в кино и театре — 6 человек (это *КТ* + *КТЦ*);

— в кино и цирке — 10 человек (*КЦ* + *КТЦ*);

— в театре и цирке — 4 человека (*ТЦ* + *КТЦ*).

Сложив все значения, имеем:

$$КТ + КТЦ + КЦ + КТЦ + ТЦ + КТЦ = 20 \ (6).$$

Вычтя (5) из (6), получим:

КТЦ + *КТЦ* = 20 – 18 = 2, то есть *КТЦ* = 1 (только один ученик был на каникулах и в кино, и в театре, и в цирке).

Ответ: 1.

Новые задачи

1. На фирме работают 57 человек. Из них 26 знают английский язык, 21 — немецкий язык, а 6 — оба языка. Сколько сотрудников фирмы:

1) не знают ни английского, ни немецкого языков;
2) знают хотя бы один из указанных языков;
3) знают только один английский язык;
4) знают только по одному иностранному языку?

2. В трех классах 35 мальчиков. 24 из них играют в футбол, 18 — в волейбол, 12 — в баскетбол. 10 ребят одновременно играют в футбол и в волейбол, 8 — в футбол и в баскетбол, а 5 — в волейбол и в баскетбол. Сколько мальчиков играют и в футбол, и в волейбол, и в баскетбол одновременно?

Круги Эйлера и логические операции

Д.М. Златопольский,
Москва

Вспомним задачу об учениках и библиотеках (см. выше), но при этом изменим количественные показатели. Пусть информация об учениках, посещающих ту или иную библиотеку, представлена в табл. 1:

Таблица 1

Посещает	**Ученики**																			
	А	*Б*	*В*	*Г*	*Д*	*Е*	*Ж*	*З*	*И*	*К*	*Л*	*М*	*Н*	*О*	*П*	*Р*	*С*	*Т*	*У*	*Ф*
Школьную																				
Районную																				

Фамилии учеников во второй строке условно обозначены буквами. Запишем в третьей строке букву “И” (от слова “истина”) для тех учеников, которые посещают школьную библиотеку, в четвертой строке — районную:

Таблица 2

Посещает	**Ученики**																				**Всего**
	А	*Б*	*В*	*Г*	*Д*	*Е*	*Ж*	*З*	*И*	*К*	*Л*	*М*	*Н*	*О*	*П*	*Р*	*С*	*Т*	*У*	*Ф*	
Школьную		И	И	И	И	И	И	И				И	И	И		И	И		И		13
Районную	И	И			И		И		И	И		И		И				И	И	И	11

Дополним таблицу еще одной строкой и запишем в нее букву “И” для тех учеников, которые посещают как школьную, так и районную библиотеки:

Таблица 3

	Ученики																				
Посещает	*А*	*Б*	*В*	*Г*	*Д*	*Е*	*Ж*	*З*	*И*	*К*	*Л*	*М*	*Н*	*О*	*П*	*Р*	*С*	*Т*	*У*	*Ф*	**Всего**
Школьную		И	И	И	И	И	И	И				И	И	И		И	И		И		13
Районную	И	И			И		И		И	И		И		И				И	И	И	11
Обе		И			И		И					И		И					И		6

Видно, что заполнение новой строки в табл. 3 проводилось по правилам, аналогичным правилам определения результата логической операции конъюнкции, или логического умножения (с логической связкой **И**):

1-е значение	Ложь	Ложь	Истина	Истина
2-е значение	Ложь	Истина	Ложь	Истина
(1-е значение) **И** (2-е значение)	Ложь	Ложь	Ложь	Истина

Нетрудно убедиться также, что значение числа учеников, посещающих обе библиотеки, соответствует формуле 4 в статье “Круги Эйлера (диаграммы Эйлера — Венна)” с учетом того, что некоторые ученики вообще не посещают указанные библиотеки (правда, может, они пользуются городской библиотекой).

Поэтому можно сказать, что область, выделенная на *рис.* 1 в только что указанной статье, соответствует логической операции конъюнкции двух множеств (кругов) — учеников, которые посещают школьную библиотеку, и учеников, которые пользуются районной библиотекой. Заметим также, что множество, являющееся результатом конъюнкции двух других множеств, называют “пересечением исходных множеств”.

Дополним табл. 2 еще одной строкой и запишем в нее букву “И” для тех учеников, которые посещают только какую-то одну библиотеку:

Таблица 4

	Ученики																				
Посещает	*А*	*Б*	*В*	*Г*	*Д*	*Е*	*Ж*	*З*	*И*	*К*	*Л*	*М*	*Н*	*О*	*П*	*Р*	*С*	*Т*	*У*	*Ф*	**Всего**
Школьную		И	И	И	И	И	И	И				И	И	И		И	И		И		13
Районную	И	И			И		И		И	И		И		И				И	И	И	11
Какую-то одну	И		И	И		И		И	И	И			И			И	И	И		И	12

Здесь результат в четвертой строке определяется по правилам для логической операции строгой дизъюнкции (с логической связкой **XOR — исключающего ИЛИ**):

1-е значение	Ложь	Ложь	Истина	Истина
2-е значение	Ложь	Истина	Ложь	Истина
(1-е значение) **XOR** (2-е значение)	Ложь	Истина	Истина	Ложь

В кругах Эйлера область, соответствующая множеству учеников, посещающих только какую-то одну библиотеку, на *рис*. 1 ниже выделена желтым цветом, и она соответствует логической операции исключающего **ИЛИ** двух заданных множеств (кругов).

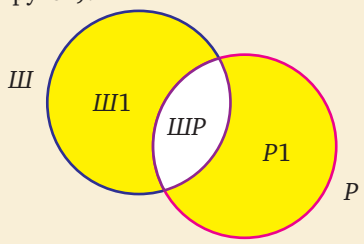


Рис. 1

Определим также число учеников, пользующихся по крайней мере одной библиотекой:

Таблица 5

Посещает	Ученики																				**Всего**
	А	*Б*	*В*	*Г*	*Д*	*Е*	*Ж*	*З*	*И*	*К*	*Л*	*М*	*Н*	*О*	*П*	*Р*	*С*	*Т*	*У*	*Ф*	
Школьную		И	И	И	И	И	И	И				И	И	И		И	И		И		13
Районную	И	И			И		И		И	И		И		И				И	И	И	11
Одну или две	И	И	И	И	И	И	И	И	И	И		И	И	И		И	И	И	И	И	18

Результат (18) соответствует формуле (3) в статье “Круги Эйлера (диаграммы Эйлера — Венна)”, а в кругах Эйлера соответствующее множество на *рис*. 2 выделено серым цветом.

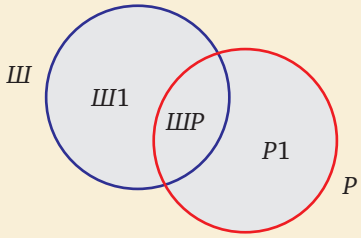


Рис. 2

Это множество называют “объединением исходных множеств”, или “соединением исходных множеств”.

Задания для самостоятельной работы

1. Нарисуйте круги Эйлера, выделив на них область, соответствующую множеству учеников:

1) не посещающих районную библиотеку, а пользующихся только школьной;

2) посещающих только городскую библиотеку (допустив, что среди 20 рассматриваемых учеников такие есть).

2. Составьте список учеников (используя вместо фамилий буквы), образующих пересечение множеств, заданных в табл. 2 и соответствующих отдельным библиотекам.

3. Составьте список учеников (используя вместо фамилий буквы), образующих объединение указанных выше множеств.

4. В таблице представлена информация о 16 учениках:

2	**Посещает**	*А*	*Б*	*В*	*Г*	*Д*	*Е*	*Ж*	*З*	*И*	*К*	*Л*	*М*	*Н*	*О*	*П*	*Р*
3	Школьную	И		И	И		И		И			И	И		И	И	И
4	Районную		И	И	И	И		И		И	И		И		И		

— где буква “И” говорит о том, что данный ученик посещает соответствующую библиотеку.

Оформите лист электронной таблицы для определения числовых значений в столбце R:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1		**Фамилия**																
2	**Посещает**	*А*	*Б*	*В*	*Г*	*Д*	*Е*	*Ж*	*З*	*И*	*К*	*Л*	*М*	*Н*	*О*	*П*	*Р*	**Всего**
3	Школьную																	
4	Районную																	
5	Одну или две																	
6	Только одну районную																	
7	Обе																	
8	Какую-то одну																	
9	Ни одной																	

Указания по выполнению. Используйте значения ИСТИНА и ЛОЖЬ, логические функции, а также функцию СУММ (SUM) или СЧЕТЕСЛИ (COUNTIF).

В ответе на задание 2 укажите только формулы в ячейках B5:B9 (листы присылать не нужно).

“ЛОМАЕМ” ГОЛОВУ

Числовые ребусы в троичной системе

В приведенных ниже ребусах зашифрованы числа, записанные в троичной системе счисления. Одинаковым буквам соответствуют одинаковые цифры. Звездочкой (“*”) может быть любая цифра.

1.

```
   *
+  *
   *
```

2.

```
   A
+  *
A  *
```

3.

```
   B
+  B
*  *
```

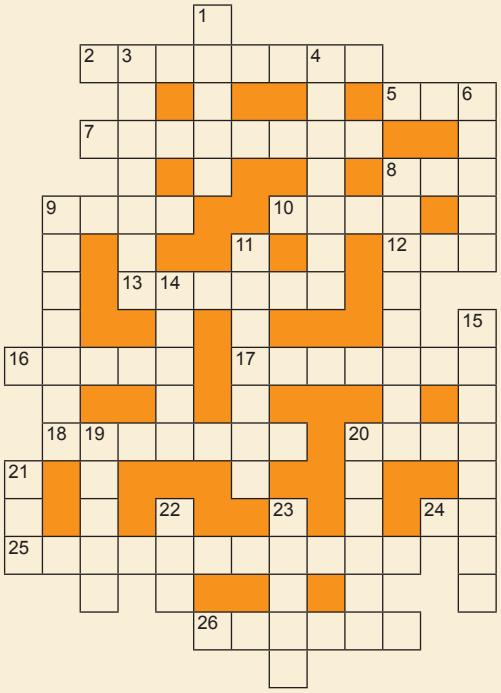
Шарики в коробочках

Имеется пять коробочек: белая, черная, красная, синяя и зеленая. Также есть по два шарика для каждого из указанных цветов. В каждой коробочке лежит по два шарика, причем цвета коробочки и шариков могут и не совпадать. Известно, что:

1. Ни один шарик не лежит в коробочке того же цвета, что и он сам;
2. В красной коробочке нет синих шариков;
3. В коробочке нейтрального цвета (то есть белого или черного) лежит один красный и один зеленый шарик;
4. В черной коробочке лежат шарики холодных тонов (зеленый и синий цвета);
5. В одной из коробочек лежат один белый и один синий шарик;
6. В синей коробочке находится один черный шарик. Какого цвета шарики лежат в каждой коробочке?

Кроссворд

Решите, пожалуйста, кроссворд.



По горизонтали

2. Совокупность четко определенных правил для решения задачи за определенное число шагов.
5. То, без чего не может работать исправный и включенный в сеть настольный персональный компьютер.
7. Логическая операция.
8. Буква греческого алфавита, которой, как правило, обозначают неизвестную величину.
9. Поименованная совокупность данных на носителе.
10. Популярный вид компьютерных программ.
12. Число в системе условных обозначений символов.
13. Устройство управления работой шины персонального компьютера, а также спортивный судья.

16. Единица измерения массы.
17. Поиск и исправление ошибок в разрабатываемой программе.
18. Действие, проводимое с клавишей.
20. …данных.
24. Вторая нота.
25. Одна из основных профессий в области информационных технологий.
26. Наука о законах и формах мышления.

По вертикали
1. .ru, .com, .edu, .py, …
3. Часть окна текстового редактора, используемая для установки полей, отступов и т.п.
4. Электронная схема, применяемая в регистрах компьютера для запоминания одного бита информации.
6. Электрод электронной лампы.
8. Средство, облегчающее доступ к нужной странице в книге или в WWW.
9. Один из первых языков программирования высокого уровня.
11. Единица измерения информации, равная 1024 бита.
14. Прямоугольник, ограничивающий меню или т.п.
15. Поиск и исправление ошибок в разрабатываемой программе.
19. Разработчик программы.
20. Язык программирования — рекордсмен по числу различных вариантов.
21. Характеристика величины, определяющая множество ее допустимых значений и применимых к ней операций.
22. Цифра восьмеричной системы счисления.
23. Буква греческого алфавита.

Получить число 1957

Как известно, сумма десяти цифр равна 45:

$$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 0 = 45.$$

А как надо расставить знаки сложения и умножения между этими же цифрами, не изменяя порядка их последовательности, чтобы результат арифметических действий равнялся бы числу 1957? Между некоторыми цифрами можно не ставить знак, объединяя их в “многозначное” число.

Переправа на остров

Трое мальчиков пошли на рыбалку, взяв с собой лодку, выдерживающую нагрузку до 100 кг. Как перебраться мальчикам с берега реки на остров, если их массы равны 40, 50 и 70 кг?

Фальшивая гирька

Имеются 6 гирек массой 1, 2, 3, 4, 5 и 6 г. На них нанесена соответствующая маркировка. Однако есть основания считать, что при маркировке гирь допущена одна ошибка. Как при помощи двух взвешиваний на чашечных весах, на которых можно сравнить веса любых групп гирь, определить, верна ли имеющаяся на гирях маркировка?

Чему равны три ТРЕТИ?

Решите, пожалуйста, числовой ребус, в котором, как обычно, одинаковым буквам соответствуют одинаковые цифры, разным буквам — разные цифры:

ТРЕТЬ + ТРЕТЬ + ТРЕТЬ = ЦЕЛОЕ

Абсурдный результат

Возьмем произвольное, отличное от 0, число a. Обозначим его буквой x, $x = a$. Обе части этого равенства умножим на $-4a$. Получим

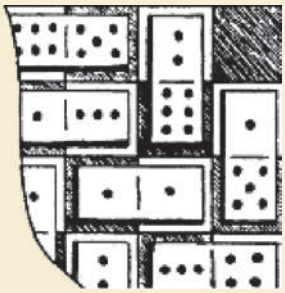
$$-4ax = -a^2, \text{ или } -4ax + 4a^2 = 0.$$

К обеим частям этого равенства прибавим x^2. Получим $x^2 - 4ax + 4a^2 = x^2$, или $(x - 2a)^2 = x^2$. Значит, $x - 2a = x$, но $x = a$, поэтому $a - 2a = a$, или $-a = a$.

Как такое могло получиться?

Доминошки на шахматной доске

Можно ли покрыть шахматную доску доминошками так, чтобы свободными остались только клетки $a1$ и $h8$?



Песочные часы

Предлагаем учащимся 1–7-х классов решить две задачи, связанные с песочными часами.

Задача 1

Есть двое песочных часов: на 4 минуты и на 9 минут. Как определить промежуток времени в 7 минут?

Задача 2

Есть двое песочных часов: на 4 минуты и на 11 минут. Как определить промежуток времени в 2 минуты?

Время, затрачиваемое на переворачивание часов, не учитывать. Задачи надо решить за минимально возможное число операций (установок часов и их переворачиваний).

Электронные таблицы и… прыжки в высоту

А.И. Азевич,
Москва

Предлагаю читателям решить ряд задач в среде электронных таблиц Microsoft Excel, Oracle OpenOffice.org Calc или др.

1. На соревнованиях по прыжкам в высоту судейская коллегия решила использовать электронную таблицу для учета хода соревнований и подведения их итогов:

	A	B	C	D	E	…	J
1			**Высота, см**				
2	**№**	**Фамилия**	150	153	156	…	171
3	1		–	–	–	…	–
4	2		–	–	–	…	–
…	…	…	…	…	…	…	…
17	15		–	–	–	…	–
18							

Если какой-то спортсмен преодолел ту или иную высоту, то в соответствующей ячейке проставляется символ “+”.

	A	B	C	D	E	…	J
1			**Высота, см**				
2	**№**	**Фамилия**	150	153	156	…	171
3	1	Абрамов	+	+	–	…	–
4	2	Бойко	+	–	–	…	–
5	3	Варданян	–	–	–		–
…	…	…	…	…	…	…	…
17	15	Ямпольский	+	+	–	…	–
18							

Подготовьте лист для определения в столбце K наибольшей высоты, взятой каждым спортсменом.

Указания по выполнению

1. Используйте функции ЕСЛИ и НАИБОЛЬШИЙ.
2. Вспомогательные расчеты проведите вне зоны видимости листа[2].

2. Решите задачу 1 при условии, что вместо символов “–” и “+” в ячейках проставляются, соответственно, символы “0” и “1”.

3. На соревнованиях по прыжкам в высоту судейская коллегия решила применить электронную таблицу для учета хода соревнований и подведения их итогов:

	A	B	C	D	E	F	G	H	I	J	K		X	Y	Z
1		**Высота, см**	150			153			156			…	171		
2		Попытка	*1*	*2*	*3*	*1*	*2*	*3*	*1*	*2*	*3*	…	*1*	*2*	*3*
3	**№**	**Фамилия**													
4	1	Абрамов	1	0	0	0	1	0	0	1	0	…	0	0	0
5	2	Бойко	0	0	1	0	1	0	0	0	1	…	0	0	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
17	15	Ямпольский	0	1	0	0	0	1	0	1	0	…	0	0	0
18															

Если какой-то спортсмен преодолел ту или иную высоту, используя ту или иную попытку, то в соответствующей ячейке проставляется символ “1”.

Подготовьте лист для определения в столбце AA наибольшей высоты, взятой каждым спортсменом.

Указание по выполнению. Используйте функции ЕСЛИ, СУММ и НАИБОЛЬШИЙ.

4. Решите задачу 3 при условии, что вместо символов “0” и “1” в ячейках проставляются, соответственно, символы “–” и “+”:

[2] Аналогично оформите лист и в других задачах.

	A	B	C	D	E	F	G	H	I	J	K		X	Y	Z
1		**Высота, см**	150			153			156			…	171		
2		Попытка	*1*	*2*	*3*	*1*	*2*	*3*	*1*	*2*	*3*	…	*1*	*2*	*3*
3	**№**	**Фамилия**													
4	1	Абрамов	+	–	–	–	+	–	–	+	–	…	–	–	–
5	2	Бойко	–	–	+	–	+	–	–	–	+	…	–	–	–
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
17	15	Ямпольский	–	+	–	–	–	+	–	+	–	…	–	–	–
18															

Указание по выполнению. Используйте функции ЕСЛИ, ИЛИ и НАИБОЛЬШИЙ.

5. Для условий задачи 1 подготовьте лист для определения в ячейке C19 лучшего результата среди всех спортсменов, а в столбце L — места, занятого каждым участником соревнований:

	A	B	C	D	E	…	J	K	L
1			**Высота, см**						
2	**№**	**Фамилия**	150	153	156	…	171		Место
3	1	Абрамов	+	+	–	…	–		
4	2	Бойко	+	–	–	…	–		
5	3	Варданян	–	–	–		–		
…	…	…	…	…	…	…	…		
17	15	Ямпольский	+	+	–	…	–		
18									
19		Лучший результат							

Указание по выполнению. Для определения занятого места используйте функции РАНГ.

6. На соревнованиях по прыжкам в высоту судейская коллегия решила применить электронную таблицу для учета хода соревнований и подведения их итогов:

	A	B	C	D	E	…	J
1			**Высота, см**				
2	**№**	**Фамилия**	150	153	156	…	171
3	1	Абрамов	2	1	0	…	0
4	2	Бойко	3	0		…	0
5	3	Варданян	1	2	2		0
…	…	…	…	…	…	…	…
17	15	Ямпольский	0	0	0	…	0
18							

Если какой-то спортсмен преодолел ту или иную высоту, то в соответствующей ячейке проставляется номер попытки, с которой он сделал это.

Подготовьте лист для определения в столбце K для каждого спортсмена номера попытки, с которой он преодолел наибольшую для него высоту.

Комментарий к выполнению. Найдите ячейку, соответствующую минимальной высоте, которую каждый спортсмен не преодолел. Условие для нахождения такой ячейки — сумма чисел в диапазоне от этой ячейки до столбца J должна быть равна нулю. Искомое значение будет в предыдущей (слева) ячейке.

7. На соревнованиях по прыжкам в высоту судейская коллегия решила применить электронную таблицу для учета хода соревнований и подведения их итогов:

	A	B	C	D	E	F	G	H	I	J	K	L
1			**Высота, см**									
2	**№**	**Фамилия**	150	153	156	159	162	165	168	171	**Результат**	**Место**
3	1	Абрамов	2	1	0	0	0	0	0	0	153	13
4	2	Бойко	1	1	2	2	1	2	0	0	165	1
5	3	Варданян	1	2	2	1	3	3	0	0	165	2
…	…	…	…	…	…	…	…	…	…	…	…	…
17	15	Ямпольский	3	0	0	…	0	0	0	0	150	15
18												

Подготовьте лист для определения в столбце L места, занятого каждым участником соревнований. В случае если несколько спортсменов показали одинаковый результат, то место каждого из них определяется

по лучшей попытке, с которой этот результат был показан.

Допускается, что одинаковый результат показали не более трех спортсменов и при этом они использовали разные попытки (первую, вторую или третью).

Комментарий к выполнению. Так как сравнивать результаты всех спортсменов между собой сложно, то придется при определении места, занятого *каждым* участником, кроме его результата, учитывать номер соответствующей попытки. Для этого каждой попытке следует сопоставить некоторое число (чем меньше номер попытки, тем больше число). Значения этих чисел должны быть такими, чтобы спортсмен, взявший меньшую высоту, не занял более высокое место, чем тот, кто преодолел большую высоту.

От редакции. Рабочие книги электронной таблицы с решениями присылайте в редакцию. В качестве имени книги укажите свою фамилию, а в качестве имен рабочих листов книги используйте номер задачи (переименовав листы и при необходимости добавив новые).

ЯПОНСКИЙ УГОЛОК

Два судоку

Решите, пожалуйста, две японские головоломки “судоку”:

1) простую:

2	4				5	3		8
			2			6		5
	6		8		7	9		
	1	4		8			6	2
	5	2	3		4	8	9	
8	9			5		4	3	
		9	5		1		8	
1		7			6			
6		5	9				4	3

2) сложную:

			9					6
1			7	2	5	3		8
9					6	5		7
2	9		6	7				
			2					
		8		5		4	1	
		4			1	6	8	
	6					9		
					7		5	

Ответы (можно не на все головоломки) присылайте в редакцию.

КРЕПКИЙ ОРЕШЕК

Как обычно, в этой рубрике мы проводим разбор задач, решение которых вызвало трудности.

Числовой ребус со звездочками

Напомним, что требовалось решить числовой ребус, в котором звездочками (“*”) могут быть любые цифры:

```
              5 * * 4
            × 5 * * 7
            -----------
            4 0 * * *
          1 * 4 * *
        1 7 * * *
    2   * * * 0
    -----------------
    *   0 * * 5 * 1 *
```

Благодаря читателей, приславших правильный ответ и которые будут награждены дипломами:

— Глазкову Екатерину, Республика Коми, г. Сыктывкар, МОУ “Лицей народной дипломатии”, учитель **Гранаткина О.М.**;

— Лопаткину Софию, г. Ярославль, школа № 33, учитель **Ярцева О.В.**,

приведем начало анализа.

Запишем в ребус очевидные значения (они выделены синим цветом и полужирным начертанием).

```
              5 * * 4
            × 5 3 * 7
            -----------
            4 0 * * 8
          1 * 4 * *
        1 7 * * *
    2   * * * 0
    -----------------
    3   0 * * 5 * 1 8
```

Кроме того, можно утверждать, что звездочка, оформленная красным цветом, может быть равна 2 или 3.

Далее следует исследовать слагаемые, образующие второй справа разряд результата:

```
    *   4
    *   7
    -------
    A   8
    B

    -------
    1   8
```

Возможные варианты сочетаний, при которых вторая справа цифра результата равна 1, приведены в таблице:

№	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
A	0	1	2	9	3	8	4	7	5	6
B	1	0	9	2	8	3	7	4	6	5

Вариант № 1 невозможен, так как произведение числа 4 ни на какое другое число не может оканчиваться на 1.

Предлагаем читателям исследовать все другие варианты, найти решение ребуса и прислать его в редакцию.

Задача “Работники магазина”

Напомним условие: “В магазине работают товаровед, кассир, бухгалтер, продавец и директор. Их фамилии — Соломенко, Боженко, Кондратенко, Демиденко и Василенко. Когда кассир и директор учились в торговом техникуме, они жили в одной комнате общежития. Товаровед пока еще семьей не обзавелся. Василенко и Соломенко недолюбливают друг друга. Жена Кондратенко очень обрадовалась, когда узнала, что директор разрешил ее мужу взять очередной отпуск в июне. Демиденко очень огорчился, узнав от директора, что бухгалтер и кассир недавно поженились. Соломенко и Боженко пока еще замуж не вышли. Кто кем работает?”.

Начало решения

Пронумеруем известные факты.

1. Когда кассир и директор учились в торговом техникуме, они жили в одной комнате общежития.
2. Товаровед пока еще семьей не обзавелся.
3. Василенко и Соломенко недолюбливают друг друга.
4. Жена Кондратенко очень обрадовалась, когда узнала, что директор разрешил ее мужу взять очередной отпуск в июне.
5. Демиденко очень огорчился, узнав от директора, что бухгалтер и кассир недавно поженились.
6. Соломенко и Боженко пока еще замуж не вышли. На их основании можно составить таблицу, проставив знак “–” для невозможных вариантов (в скобках указаны номера фактов, обосновывающих это):

	Товаровед	Кассир	Бухгалтер	Продавец	Директор
Соломенко Ж		– (5, 6)	– (5, 6)		
Боженко Ж		– (5, 6)	– (5, 6)		
Кондратенко М	– (2, 4)				– (4)
Демиденко М		– (5)	– (5)		– (5)
Василенко					

Примечание. М — мужчина, Ж — женщина.

Из нее следует, что бухгалтер и кассир — Кондратенко и Василенко и при этом Василенко женщина (факт 5).

Допустим, что Кондратенко — кассир. Тогда таблица примет вид:

	Товаровед	Кассир	Бухгалтер	Продавец	Директор
Соломенко Ж		– (5, 6)	– (5, 6)		
Боженко Ж		– (5, 6)	– (5, 6)		
Кондратенко М	– (2, 4)	+	–	–	– (4)
Демиденко М		– (5)	– (5)		– (5)
Василенко Ж	–	–	+	–	–

Далее, так как кассир — мужчина, то из факта 1 следует, что директор тоже должен быть мужчиной. Но Демиденко не может быть директором, то есть в этом случае имеет место противоречие.

Значит, Кондратенко — бухгалтер. В этом случае таблица станет следующей:

	Товаровед	Кассир	Бухгалтер	Продавец	Директор
Соломенко Ж		– (5, 6)	– (5, 6)		
Боженко Ж		– (5, 6)	– (5, 6)		
Кондратенко М	– (2, 4)	–	+	–	– (4)
Демиденко М		– (5)	– (5)		– (5)
Василенко Ж	–	+	–	–	–

Предлагаем читателям продолжить анализ и, получив ответ, прислать его в редакцию.

А здесь заметим, что Мусатов Максим и Мусатов Тимофей, ученики школы № 9 г. Златоуста Челябинской обл. (учитель **Мусатова И.Б.**), приславшие правильный ответ, будут награждены дипломами. Поздравляем!

Задача “Футбольная таблица”

Напомним, что требовалось восстановить результаты всех матчей турнира четырех футбольных команд по итоговой таблице:

Таблица 1

Команда	Игр	Выигрышей	Ничьих	Поражений	Голы		Всего очков
					Забито	Забито	
“Олимпия”	3	3	0	0	7	1	6
“Дина”	3	1	1	1	2	3	3
“Днепр”	3	1	1	1	3	3	3
“Волга”	3	0	0	3	1	6	0

Известно также, что “Олимпия” выиграла у “Дины” со счетом 3:0. За победу команде давалось 2 очка, за ничью — 1, за поражение — 0.

Благодаря Софию Лопаткину, г. Ярославль, школа № 33 (учитель **Ярцева О.В.**), правильно решившую задачу (София будет награждена дипломом), приведем начало решения.

Изменим заданную в условии таблицу, предусмотрев в ней столбцы для результатов игр (см. табл. 2).

Таблица 2

Команда	“Олимпия”	“Дина”	“Днепр”	“Волга”	Выигрышей	Ничьих	Поражений	Голы		Очки
								Забито	Пропущено	
“Олимпия”	X	3:0			3	0	0	7	1	6
“Дина”	0:3	X			1	1	1	2	3	3
“Днепр”			X		1	1	1	3	3	3
“Волга”				X	0	0	3	1	6	0

В нее можно вписать результаты игр без забитых и пропущенных мячей (В — выигрыш, Н — ничья, П — поражение):

Таблица 3

Команда	“Олимпия”	“Дина”	“Днепр”	“Волга”	Выигрышей	Ничьих	Поражений	Голы		Очки
								Забито	Пропущено	
“Олимпия”	X	3:0	В	В	3	0	0	7	1	6
“Дина”	0:3	X	Н	В	1	1	1	2	3	3
“Днепр”	П	Н	X	В	1	1	1	3	3	3
“Волга”	П	П	П	X	0	0	3	1	6	0

Рассмотрим голы, забитые и пропущенные “Диной”. Из последней таблицы следует, что в матчах с командами “Днепр” и “Волга” она голов не пропускала и два мяча забила. Значит, возможен только один вариант результатов игр этих команд: “Дина” : “Днепр” — 0:0 и “Дина” : “Волга” — 2:0. Дальнейший анализ проведите самостоятельно. Ответ, пожалуйста, присылайте в редакцию.

Бедный торговец (старинная задача)

Напомним условие: “В лавке бедного торговца вместо гирь было всего 4 камня. Однако с помощью этих камней он на рычажных чашечных весах совершенно правильно взвешивал предметы массой 1, 2, …, 40 кг. Какого веса были камни?”.

Для нахождения ответа целесообразно обсудить такую задачу.

“Есть пружинные весы, которые нужно отградуировать — сделать отметки на значения массы 1, 2, … кг. Каким должен быть набор гирь-разновесов, который можно использовать для градуировки весов на интервал масс 1, 2, …, 31 кг, чтобы при этом число гирь в наборе было минимальным?”.

Здесь рассуждения такие.

1. Десятичное число 31 в двоичной системе счисления выглядит так: 11111, а все числа, меньшие 31, в этой системе состоят (естественно) из единиц и нулей.

2. Любое двоичное число от 1 до 11111 можно получить, складывая двоичные числа 1, 10, 100, 1000 и 10 000 (убедитесь в этом!). Эти числа есть двойка в степени 0, 1, 2, 3 и 4, т.е. десятичные цифры 1, 2, 4, 8 и 16.

Значит, минимальный набор гирь-разновесов, который можно использовать для градуировки весов на интервал масс 1–31 кг, — это гири массой 1, 2, 4, 8 и 16 кг. Например, чтобы отградуировать весы на

3 кг ($3_{10} = 11_2$), можно использовать гири массой 1 и 2 кг (этим значениям соответствуют двоичные числа 1 и 10), на 13 кг ($13_{10} = 1101_2$) — гири массой 1, 4 и 8 кг (1_2, 100_2 и 1000_2).

На рычажных чашечных весах (как в "основной" задаче) при взвешивании груза камни можно размещать на обеих чашках — и на свободной, и вместе с грузом. Следовательно, в нашей задаче надо найти такой набор чисел, которые можно не только складывать, но и вычитать. Если здесь также рассматривать представление десятичных чисел 1, 2, …, 40 в двоичной системе счисления, то выяснится, что для взвешивания понадобится шесть разных камней — массой 1, 2, 4, 8, 16 и 32 кг (убедитесь, что с их помощью можно определить все нужные значения!). В условии же задачи говорится только о четырех камнях. Значит, надо попробовать использовать другие системы счисления. Чтобы найти, какую именно, рассмотрим более простую задачу: "С помощью какого минимального набора камней разной массы можно взвешивать предметы массой 1, 2, 3 и 4 кг?". Ответ здесь такой: с помощью двух камней — массой 1 и 3 кг. Это должно навести на мысль о том, что нужно использовать троичную систему счисления. Действительно, любое троичное число от 1 до 1111 ($40_{10} = 1111_3$) можно получить, складывая или вычитая числа 1_3, 10_3, 100_3 и 1000_3 (убедитесь в этом!). Эти числа есть тройка в степени 0, 1, 2 и 3, то есть десятичные числа 1, 3, 9 и 27, — именно такой массы и были камни в лавке бедного торговца.

Правильный ответ представили Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия, Тютин Кирилл и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.** Они же правильно решили задачи, предложенные для самостоятельной работы в статье "Об уравновешенной троичной системе". Все перечисленные читатели будут награждены дипломами. Поздравляем!

Дополнительное задание для самостоятельной работы

Подумайте, как определить, какие камни (или гири такого же веса) надо класть вместе с грузом, а какие — на свободную чашку, чтобы при грузе массой *A* кг весы были в равновесии.

Для решения еще раз ознакомьтесь с только что упомянутой статьей в сентябрьском выпуске прошлого года.

ЗАДАЧНИК

Четыре задачи

Е.А. Мирончик,
учитель информатики МБНОУ "Лицей № 111",
г. Новокузнецк Кемеровской обл.

1. Ира и Катя играют в игру — "угадайку", используя карточки:

Ира загадывает одну из карточек, а Катя должна отгадать ее. На вопрос можно отвечать только ответами "Да" или "Нет". Помогите Кате составить вопросы так, чтобы быстрее отгадать загаданную карточку.

2. В записи пароля использовали цифры, русские и латинские буквы. Пароль состоит из пяти символов. Известны все символы, кроме одного. Точно известно, что среди перечисленных ниже букв имеется необходимая.

Q А Ё I Ъ В Ю К Ц Я

Отвечая на какой из перечисленных вопросов "да" или "нет", сообщат один бит информации?

1) эта буква существует в русском алфавите?
2) эта буква есть среди перечисленных?
3) эта буква существует в латинском алфавите?
4) это цифра?
5) эта буква существует в том и другом алфавите?

3. Каким минимальным количеством бит можно закодировать состояние игры в "крестики-нолики" на поле 5 × 5 клеток?

4. Игральные карты выкладывают по принципу — после черной карты обязательно должна следовать красная, а после красной — черная. Цепочку составляют только из карт-картинок (валет, дама, король, туз) всех мастей. Сколько возможно вариантов цепочек из:

1) трех карт; 2) четырех карт?

Статистика

В классе 25 человек, каждого из которых в зависимости от роста можно отнести к высоким, средним или низким, из них три двоечника и пять отличников. Низких в классе восемь человек, высоких — девять, есть двоечники всех размеров, а среди "середняков" (не двоечников и не отличников) — три ученика среднего роста. Сколько отличников среднего и низкого роста, если самый высокий в классе тоже отличник?

2014 год и степени двойки

Как, конечно, известно читателям, число 2 и его степени играют в информатике большую роль. Найдите все даты 2014 года, для которых выражение ($день^{месяц} - год$) кратно наибольшей возможной степени двойки.

Цепочки цифр

Строки (цепочки цифр) создаются по следующему правилу.

Первая строка состоит из одной цифры 0. Каждая из последующих цепочек создается такими действиями: в очередную строку сначала записывается цифра, на единицу большая последней цифры в предыдущей строке, к ней слева дважды подряд приписывается предыдущая строка.

(1) 0
(2) 001
(3) 0010012
(4) 001001200100123.

Какие из приведенных ниже строчек соответствуют описанным правилам?

(1) 00100120010012300100120010012340010012001001230010120010012345
(2) 00100120010012300100120010012340010012001001230010012001012345 6
(3) 00100120010012300100120010012340010012001001230010012001001 2345
(4) 0001001200100123001001200100123400100120010012300100120010 12345
(5) 00100120010012340010012001001230010012001001230010012001001 2345
(6) 00100120010012340010012001001230010012001001230010012001000 1345

Ответ обоснуйте.

Число-исполин

Для измерений в пределах Солнечной системы астрономы считают единицей длины среднее расстояние от Земли до Солнца (149 500 000 км). Это так называемая “астрономическая единица”. Но для измерения расстояний до других солнц-звезд и туманностей эта мера слишком мала. Поэтому введены в употребление более крупные единицы:

1 астрономическая единица = 149 500 000 километров;

1 парсек = 206 265 астрономических единиц, или 30 800 000 000 000 километров;

1 килопарсек = 1000 парсеков;

1 мегапарсек = 1 000 000 парсеков.

Чтобы хоть отдаленно представить себе это число-исполин (мегапарсек), дадим одно сопоставление: тончайшая паутинная нить, протянутая от Москвы до Санкт-Петербурга, весила бы 10 граммов, от Земли до Луны — не более 8 килограммов, от Земли до Солнца — 3 тонны. Но протянутая на длину одного мегапарсека она весила бы...

Каков будет вес паутинной нити? Ответ получите, используя электронную таблицу.

Литература

1. На досуге. Сборник занимательных задач. Ростов-на-Дону: Ростовское книжное изд-во, 1959.

Перестановка двойки

Некоторое число оканчивается на 2. Если эту цифру перенести в начало числа, то оно удвоится. Найдите наименьшее такое число.

Точные квадраты

При каких основаниях систем счисления k являются точными квадратами числа:

1) 121_k; 2) $12\,321_k$?

Ответы, решения, разъяснения к заданиям, опубликованным в разделе “В мир информатики” ранее

Задача “Три фигуры”

Напомним, что требовалось определить, в каком из трех домиков — с высокой крышей, но маленьким окном, с высокой крышей и большим окном, с низкой крышей и большим окном — “живет” каждая из трех фигур — Треугольник, Круг и Квадрат, если Треугольник и Круг “живут” в домиках с большими окнами, а Круг и Квадрат — в домиках с высокой крышей.

Ответ

Круг “живет” в доме с высокой крышей и большим окном, Квадрат — в доме с высокой крышей и маленьким окном, Треугольник — с низкой крышей и большим окном.

Правильные ответы представили:

— Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.**;

— Березина Ксения, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Гарифуллин Игорь, Еличева Анастасия, Жих Данила, Костина Виктория, Логинов Никита, Труханова Яна и Федоров Дмитрий, Республика Карелия, г. Сегежа, школа № 5, учитель **Меньшиков В.В.**;

— Гируцкий Павел, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Гололобов Дмитрий, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Иванова Виолетта и Левченко Ирина, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Корольчук Сергей, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Богданова Л.М.**;

— Михайлова Светлана, Чувашская Республика, г. Канаш, Канашский педагогический колледж, преподаватель **Воеводина Р.В.**;

— Мусатов Максим и Мусатов Тимофей (ученик 2-го класса), Челябинская обл., г. Златоуст, школа № 9, учитель **Мусатова И.Б.**;

— Мухина Светлана, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Удалова Елизавета, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**

Отметим ответы Анастасии Еличевой и Яны Трухановой из школы № 5 г. Сегежа, разработавших презентации PowerPoint, иллюстрирующие решение задачи (остальные учащиеся этой школы оформили ответы в виде красочных рисунков).

Фальшивая банкнота

Напомним условие: “Один банкир нашел на улице банкноту в пять долларов. Он поднял ее, запомнил номер и пошел домой завтракать. За завтраком жена сообщила ему, что мясник прислал счет на пять долларов. Поскольку других денег у банкира при себе не оказалось, он отдал жене найденную банкноту, чтобы оплатить счет. Мясник отдал эту банкноту фермеру, когда покупал теленка, тот — торговцу, торговец, в свою очередь, дал ее прачке, а прачка, вспомнив, что задолжала банку пять долларов, отнесла ее туда и погасила свой долг. Банкир узнал банкноту, которой к тому времени было оплачено долгов на 25 долларов. При внимательном изучении банкнота оказалась фальшивой. Кто и сколько потерял на этих операциях? Как теперь можно компенсировать потери и вернуть долги?”.

Ответ

Все участники процесса (банкир, мясник, фермер, торговец и прачка) находятся в одинаковом положении: каждый из них должен кому-то пять долларов и ему должны ровно столько же.

Поэтому все пять участников могут собраться вместе и договориться считать долги взаимно погашенными.

Правильный ответ прислали:

— Барановская Татьяна и Жукова Ирина, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Березин Василий, Демьянова Елена и Хомякова Анна, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Кононенко Александра, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Лазарева Наталья, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Лопаткина София, г. Ярославль, школа № 33, учитель **Ярцева О.В.**;

— Назаров Алексей, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Новиков Сергей и Хромченкова Елизавета, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Федоренко Ирина, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**

Две задачи о числе 2013

Напомним условия задач.

1. Замените одинаковые буквы одинаковыми цифрами, а разные буквы разными так, чтобы следующее равенство было верным:

$$\mathbf{М} \times \mathbf{О} \times \mathbf{ДЕМ} = 2013$$

2. Замените одинаковые буквы одинаковыми цифрами, а разные буквы — разными так, чтобы число **ШАЛАШ** делилось на 2013 без остатка.

Решение задачи 1

Разложив число 2013 на простые множители, получим: $2013 = 1 \cdot 3 \cdot 11 \cdot 61$. Два числа в левой части равенства в условии — однозначные, третье — трехзначное. Значит, последнее (**ДЕМ**) равно 671, то есть **М** = 1, **О** = 3.

Решение задачи 2

Рассматривать все возможные числа, при умножении на которые числа 2013 получится пятизначное число **ШАЛАШ** (5, 6, …, 49), нецелесообразно. Лучше рассмотреть возможные значения цифры **Ш**.

Если **Ш** = 1, то пятизначное число, оканчивающееся на 1, получается в случае умножения 2013 на 7. Однако число $2013 \times 7 = 14\,091$ не может быть числом **ШАЛАШ**.

При **Ш** = 2 имеем допустимый вариант:

ШАЛАШ = 28 182.

Ответы представили:

— Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.**;

— Береговая Ирина, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Дикарева Елизавета и Калинина Ирина, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Донникова Анна, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Киришина Татьяна, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**;

— Кононенко Александра, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Корольчук Сергей, Республика Карелия, поселок Надвоицы, школа № 1, учитель **Богданова Л.М.**;

— Куликова Клавдия, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Лазарева Наталья, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Лопаткина София, г. Ярославль, школа № 33, учитель **Ярцева О.В.**;

— Назаров Алексей, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Чукова Мария, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**

Страшный сон

Вернувшийся из Китая английский офицер заснул в церкви во время службы. Ему приснилось, что к нему приближается палач, чтобы отрубить голову, и в тот самый момент, как сабля опускалась на шею несчастного офицера, его жена, желая разбудить заснувшего, слегка дотронулась до его шеи веером. Потрясение было столь велико, что офицер тут же упал замертво.

Что-то в рассказе неладно. Что именно?

Ответ

Так как никто не мог знать сон никогда не проснувшегося офицера, то история в рассказе придумана.

Правильный ответ прислали:

— Березин Василий, Демьянова Елена и Хомякова Анна, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Воронько Алена, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Дикарева Елизавета и Калинина Ирина, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Есипова Александра, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Казакова Мария, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Кононенко Александра, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Куликова Клавдия, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Лопаткина София, г. Ярославль, школа № 33, учитель **Ярцева О.В.**;

— Маркова Татьяна, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Соловьева Марина, Вадьковская средняя школа, Брянская обл., Погарский р-н, учитель **Цыганкова И.Ю.**;

— Чукова Мария, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Южин Николай, г. Пенза, школа № 512, учитель **Гаврилова М.И.**

Головоломка “Крест-накрест”

Ответ

З	А	П	И	С	Ь
П	А	Н	Е	Л	Ь
Р	Е	М	О	Н	Т
Д	Е	Р	Е	В	О
М	А	Л	И	Н	А
П	Р	А	В	К	А

Слова на диагоналях — “ЗАМЕНА” и “ПАРОЛЬ”.

Правильные ответы представили:

— Воронова Анжелика, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Газарова Елизавета, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Глазкова Екатерина, Республика Коми, г. Сыктывкар, МОУ “Лицей народной дипломатии”, учитель **Гранаткина О.М.**;

— Данько Алина, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Дикарева Елизавета и Калинина Ирина, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Елян Наталья, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Жаркова Екатерина, Лошак Антон и Турков Андрей, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Зорина Мария, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Лопаткина София, г. Ярославль, школа № 33, учитель **Ярцева О.В.**;

— Назаров Алексей, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**

Задача “Мартышка и бананы”

Условие: “Мартышка висит на хвосте. В каждой руке по 101 банану, а в каждой ноге — на один банан больше, чем в руке. Сколько всего бананов у мартышки? Расчеты проведите в двоичной системе, в которой записаны значения в условии”.

Ответ: $10110_2 = 22$ банана.

Задача “Мартышка и перчатки”

Условие: “Мартышка-мама связала 111 своим детишкам по перчатке на каждую руку и ногу. Но они порвали все свои перчатки, кроме младшего, который порвал только 11. Мартышка решила “починить” испорченные перчатки. Сколько перчаток попадет маме в работу?

Ответ представьте и в системе счисления, в которой записаны значения в условии, и в десятичной системе. При расчетах используйте только операцию вычитания”.

Решение

Так как число 11 в троичной системе счисления равно десятичному числу 4, а по условию кто-то порвал именно это число перчаток (“только 11”), то все значения в условии не могут быть записаны в этой системе. Значит, использована двоичная система счисления.

Общее число связанных перчаток равно $111_2 \times 4_{10}$. Для получения результата умножать нет необходимости — достаточно приписать к числу справа два нуля (11100_2). Так как непорванной осталась одна перчатка, то в ремонте побывает $11100_2 - 1 = 11011_2$, или 27 перчаток.

Правильные ответы на задачи о мартышке прислали:

— Антипов Анатолий, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Береговая Ирина, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Воронова Анжелика, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Донникова Анна, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Зорина Мария, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Лопаткина София, г. Ярославль, школа № 33, учитель **Ярцева О.В.**;

— Лазарева Наталья, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Мусатов Максим, Челябинская обл., г. Златоуст, школа № 9, учитель **Мусатова И.Б.**;

— Назаров Алексей, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Рябых Анна, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**

В одном из ответов был приведен следующий комментарий к тексту “101 + 1 = 110”: “(по-русски[3]: 5 + 1 = 6)” ☺.

Пять вопросов (октябрьский выпуск “В мир информатики”, рубрика “Поиск информации”)

Ответы

1. Американский вирусолог Джон Эндерс создал вакцину против кори.

2. Убийца на картине русского художника Ивана Петровича Аргунова “Умирающая Клеопатра” — аспид (ядовитая змея).

3. В чеховском рассказе “Петров день” в терновнике запела иволга.

4. В результате “Славной революции” (принятом в историографии названии государственного переворота в Англии в 1688 году) был свергнут король Яков II Стюарт.

5. Популярная в СССР поп-группа, название которой связано с рекой, — “Аракс”.

Ответы прислали:

— Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.**;

— Баранова Анна, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Братчиков Егор, Верхотурова Екатерина, Клепикова Анастасия, Левицкий Максим, Мельникова Виталия, Оботнина Анастасия, Попова Дарья и Черепанова Галина, Свердловская обл., г. Ревда, школа № 10, учитель **Игошева А.А.**;

— Буравцова Елизавета, Вышегородских Анастасия, Гнездилова Любовь, Карасева Екатерина и Серадская Дарья, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Гололобов Дмитрий, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Гусева Маргарита, Заляева Карина, Коптева Алена и Салимов Владислав, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Данько Алина, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Елян Наталья, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Заева Кристина, Республика Башкортостан, г. Уфа, школа № 54 (Центр дистанционного обучения), учитель **Искандарова А.Р.**;

— Иванова Виолетта и Левченко Ирина, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Маркова Татьяна, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Михайлова Светлана, Чувашская Республика, г. Канаш, Канашский педагогический колледж, преподаватель **Воеводина Р.В.**;

— Назаров Александр, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Соловьева Марина, Вадьковская средняя школа, Брянская обл., Погарский р-н, учитель **Цыганкова И.Ю.**

Ребусы по программе Microsoft PowerPoint

Ответы (согласно номерам ребусов): 1. Макет. 2. Дизайн. 3. Фотоальбом. 4. Звукозапись. 5. Обычный. 6. Сеть. 7. Макросы. 8. Архитектура. 9. Орфография.

Ответы представили:

— Андреева Елена и Михайлова Светлана, Чувашская Республика, г. Канаш, Канашский педагогический колледж, преподаватель **Воеводина Р.В.**;

— Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.**;

— Березина Ксения, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Бурмантова Юлия и Епанешникова Полина, Совхозная средняя школа, Московская обл., Серебряно-Прудский р-н, поселок Успенский, учитель **Жарикова Е.Н.**;

— Володин Вячеслав, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Гируцкий Павел, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Донникова Анна, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Ефанов Артем, г. Ярославль, школа № 33, учитель **Цикина Е.Н.**;

— Заляева Карина, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Истомина Полина, Челябинская обл., г. Златоуст, школа № 9, учитель **Мусатова И.Б.**;

— Левченко Ирина, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Нуретдинов Айдар, средняя школа села Сулеево им. Р.Г. Галеева, Республика Татарстан, Альметьевский р-н, учитель **Валиева Д.И.**;

— Омельченко Иван, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Пискунова Полина, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**

Судоку (октябрьский выпуск “В мир информатики”)

Ответы представили:

— Абрамова Екатерина, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**;

— Андреева Елена, Чувашская Республика, г. Канаш, Канашский педагогический колледж, преподаватель **Воеводина Р.В.**;

— Бадикова Ольга, Республика Башкортостан, г. Уфа, лицей № 60, учитель **Гильзер Н.В.**;

— Волчков Андрей, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Воронько Алена, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**;

— Байбуза Дарья и Репина Софья, средняя школа села Ириновка, Новобурасский р-н Саратовской обл., учитель **Брунов А.С.**;

[3] Выделено цветом редакцией.

— Гаязова Ильсеяр, Нуретдинов Айдар и Хазиев Линар, средняя школа села Сулеево им. Р.Г. Галеева, Республика Татарстан, Альметьевский р-н, учитель **Валиева Д.И.**;

— Гусева Маргарита, Заляева Карина, Копестинская Наталья и Салимов Владислав, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Карасева Ирина, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Лавренов Руслан, Вадьковская средняя школа, Брянская обл., Погарский р-н, учитель **Цыганкова И.Ю.**;

— Назаров Александр, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Пискунова Полина, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**;

— Шигаев Никита, Челябинская обл., г. Златоуст, школа № 9, учитель **Мусатова И.Б.**

Кроссворд (октябрьский выпуск “В мир информатики”)

Ответы

По горизонтали: 2. Модем. 5. Бит. 6. Код. 7. Школа. 12. Курсор. 14. Горнер. 17. Иск. 19. Ершов. 20. Аплет. 22. Розница. 25. Шрифт. 27. Фокал. 29. Плата. 32. Чип. 33. Дек. 34. Ответ.

По вертикали: 1. Винер. 3. Дно. 4. Домен. 8. Курсив. 9. Логика. 10. Икс. 11. Три. 13. Слэш. 15. Риал. 16. Сервер. 18. Строка. 21. Сноска. 23. Остап. 24. Цифра. 26. Истина. 28. Клиент. 30. Лист. 31. Тире.

Ответы прислали:

— Андреева Елена и Михайлова Светлана, Чувашская Республика, г. Канаш, Канашский педагогический колледж, преподаватель **Воеводина Р.В.**;

— Баранова Анна, средняя школа деревни Муравьево, Вологодская обл., учитель **Муравьева О.В.**;

— Волчков Андрей, средняя школа поселка Осиновка, Алтайский край, учитель **Евдокимова А.И.**;

— Горелкина Лариса и Михайлова Алена, средняя школа села Ириновка, Новобурасский р-н Саратовской обл., учитель **Брунов А.С.**;

— Долецкая Елена, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Карасева Ирина, средняя школа села Восточное Нижегородской обл., учитель **Долгова Г.А.**;

— Трептау Татьяна, Вадьковская средняя школа, Брянская обл., Погарский р-н, учитель **Цыганкова И.Ю.**;

— Тубу Наталия, г. Пенза, школа № 512, учитель **Гаврилова М.И.**;

— Туртаева Айзада, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Фатова Анастасия, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Хазиев Линар, средняя школа села Сулеево им. Р.Г. Галеева, Республика Татарстан, Альметьевский р-н, учитель **Валиева Д.И.**

Правильный ответ на задачу “На каникулах” представили:

— Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.**;

— Журавлева Анастасия, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Кононенко Александра, г. Челябинск, школа № 124, учитель **Юртаева Г.Ю.**;

— Лопаткина София, г. Ярославль, школа № 33, учитель **Ярцева О.В.**

Все они будут награждены дипломами. Поздравляем!

Методика решения задачи описана в статье “Круги Эйлера (диаграммы Эйлера — Венна)” в этом выпуске.

Ответы на задачу “Работники магазина” прислали:

— Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.**;

— Берибесова Анастасия, Дикарева Елизавета, Журавлева Анастасия и Калинина Ирина, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Мироненко Марина, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**;

— Лопаткина София, г. Ярославль, школа № 33, учитель **Ярцева О.В.**;

— Мусатов Максим и Мусатов Тимофей, г. Златоуст Челябинской обл., школа № 9, учитель **Мусатова И.Б.**;

— Назаров Алексей, средняя школа поселка Новопетровский Московской обл., учитель **Артамонова В.В.**

Правильные ответы на вопросы, предложенные в рубриках “Поиск информации” и “Для эрудитов” (сентябрьский выпуск “В мир информатики”), представили также Бельских Марина, Бойкова Юлия, Борникова Елена, Вышегородских Анастасия и Шамаев Евгений, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**

Проекты на основе статьи “Мелодия осеннего леса” (октябрьский выпуск “В мир информатики”) выполнили:

— Касимова Яна, Республика Башкортостан, г. Уфа, лицей № 60, учитель **Гильзер Н.В.**;

— Прохорова Александра, средняя школа села Горелово Тамбовской обл., учитель **Шитова Л.А.**;

— Парфененкова Анастасия, Смоленская обл., г. Демидов, школа № 1, учитель **Кордина Н.Е.** (рисунок из проекта Анастасии приведен ниже);

— Сумарокова Екатерина, г. Пенза, школа № 512, учитель **Гаврилова М.И.**

Яну, Александру, Анастасию и Екатерину редакция решила наградить дипломами.

Ответ на задачу “Восемь семейных пар” прислали также:

— Калинина Ирина, г. Воронеж, лицей № 2, учитель **Комбарова С.И.**;

— Удалова Елизавета, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**

Как и другие читатели, не побоявшиеся взяться за решение этой непростой задачи (их фамилии были опубликованы в предыдущих выпусках “В мир информатики”), Ирина и Елизавета будут награждены дипломами.

Решение кроссворда, опубликованного в сентябрьском выпуске, представили также Киришина Татьяна, Мироненко Мария и Надольный Артем, средняя школа поселка Ерофей Павлович, Амурская обл., Сковородинский р-н, учитель **Краснёнкова Л.А.**. Кроме того, Татьяна Киришина прислала ответы на вопросы, предложенные в рубриках “Поиск информации” и “Для эрудитов”, а также решение двух японских головоломок “судоку”. Редакция решила наградить Татьяну дипломом. Поздравляем всех награжденных!

ДЛЯ ЭРУДИТОВ

Да будет “свет”!

В перечисленных ниже утверждениях кое-где — чистая правда, кое-где — полная чушь. Распознаете ли, где что?

1. Имя “Люцифер” означает *несущий свет*.
2. Давление света было впервые экспериментально открыто в конце XVII века.
3. Греческий аналог славянского имени Светлана — Феодора.
4. Название “Новый Свет” дал Колумб.
5. Зодиакальный свет наблюдается вдоль эклиптики.
6. Демьян Бедный назвал своего сына Свет.
7. Главным произведением Светония является сборник жизнеописаний двенадцати римских императоров “О жизни цезарей”.
8. Благотворительный журнал “Утренний свет”, выходивший в Петербурге в 1777–1780 годах, издавался масонами.

Ответы присылайте в редакцию (можно комментировать не все утверждения).

По материалам “Российской газеты”, автор — Татьяна Луговская

ПОИСК ИНФОРМАЦИИ

Шесть вопросов

Найдите в Интернете или по другим источникам информации ответы на следующие вопросы.

1. Как называется хвойное дерево, плакучую разновидность которого вывели в Никитском ботаническом саду?

2. Каким было домашнее прозвище Владимира Маяковского?

3. Автомобилем какой марки торгует герой романа американского писателя Джона Апдайка “Кролик разбогател”?

4. Каков русский вариант испанского имени Диего?

5. Под каким псевдонимом Рудольф Абель добывал секреты американской атомной бомбы?

6. Как называется сухопутный гигант, близкими родственниками которого являются ламантин и дюгонь?

ВНИМАНИЕ! КОНКУРС!

Конкурс № 108

В качестве задания этого конкурса предлагается решить четыре задачи в статье с одноименным названием, опубликованной в рубрике “Задачник” (см. выше).

Ответы на конкурс отправьте в редакцию до 1 марта по адресу: 121165, Москва, ул. Киевская, д. 24, “Первое сентября”, “Информатика” или по электронной почте: *vmi@1september.ru*.

Конкурс № 109

Задание этого конкурса — решение задач в статье “Электронные таблицы и… прыжки в высоту” (см. рубрику “Microsoft Excel углубленно”). Можно решать не все задачи. Срок представления — 1 апреля.

Пожалуйста, четко укажите в ответах свои фамилию и имя, населенный пункт, номер и адрес школы, фамилию, имя и отчество учителя информатики. Читателей-выпускников просим учесть, что итоги конкурсов могут быть подведены в летних или осенних выпусках “В мир информатики”.

Победителями конкурса № 104 “Пушкин и кодирование слов” стали:

— Акчурин Наиль, Каширин Вячеслав, Лаврикова Дарья, Мостовой Давыд и Петухов Сергей, Московская обл., г. Наро-Фоминск, школа № 1, учитель **Павлова Н.Н.**;

— Андрющенко Александр, Остроухова Валерия, Пономаренко Анастасия и Уткина Ксения, Ставропольский край, Кочубеевский р-н, станица Барсуковская, школа № 6, учитель **Рябченко Н.Р.**;

— Васильева Мария, средняя школа села Сердар, Республика Марий Эл, учитель **Чернова Л.И.**;

— Луташева Екатерина, Ардатовский коммерческо-технический техникум, поселок Ардатов Нижегородской обл., преподаватель **Зудин В.П.**

Все они будут награждены дипломами. Поздравляем!

Решение задания конкурса будет описано в следующем выпуске.

Общероссийский проект

Школа цифрового века

Интернет-обеспечение проекта – Издательский дом «ПЕРВОЕ СЕНТЯБРЯ»

Общероссийский проект «Школа цифрового века» по комплексному обеспечению образовательных учреждений предметно-методическими материалами направлен на вовлечение педагогических работников в цифровое образовательное пространство.

В рамках проекта каждому работнику образовательного учреждения предоставляются **23 журнала** по всем учебным дисциплинам и направлениям школьной жизни с дополнительными материалами для практического использования (презентации, раздаточные материалы, образовательное видео), **методические брошюры**, **журнал для родителей**, **модульные 6–часовые курсы** из циклов «Навыки профессиональной и личной эффективности» и «Инклюзивный подход в образовании».

Открыт прием заявок от образовательных учреждений на 2014/15 учебный год

На новом этапе проекта

- Каждому педагогическому работнику образовательного учреждения будет предоставлена возможность выбрать и пройти один из десяти 36-часовых дистанционных курсов по общей педагогике с выдачей удостоверения установленного образца
- Расширится линейка модульных курсов

Стоимость участия в проекте для образовательного учреждения – 6 тысяч рублей за весь учебный год независимо от количества педагогических работников.

Участие образовательного учреждения и педагогических работников в проекте удостоверяется соответствующими документами. Для дошкольных учреждений предусмотрен свой набор удостоверяющих документов.

Срок действия проекта в 2014/15 учебном году: с 1 августа 2014 года по 30 июля 2015 года

Прием заявок и подробности

на сайте

digital.1september.ru