№ 44 (237) ноябрь 1999

еженедельное
ПРИЛОЖЕНИЕ К ГАЗЕТЕ
«ПЕРВОЕ СЕНТЯБРЯ»

ИНФОРМАТИК

Семинар: "Концепция курса информатики в 12-летней школе"


Е.К. Хеннер


Слева направо:
М.Н. Бородин, Ю.А. Первин,
А.Н. Смирнова, Н.В. Макарова.


А.В. Горячев


С.А. Христочевский


А.Г. Гейн

17 ноября в Министерстве образования состоялся семинар, посвященный обсуждению концепции курса информатики для 12-летней школы. Некоторые материалы этого семинара будут опубликованы позднее.

Читайте в номере

Секретная наука

И.Л. Миронов

Секретная наука

Криптография — это одна из самых интригующих и таинственных областей современной науки, которая не является служанкой политиков и военных, а сама меняет окружающий нас мир. Со времен Юлия Цезаря и древних греков способы сокрытия тайн охранялись строже, чем государственные секреты, так как ключ от любого сейфа представляет ценность заведомо большую, чем содержимое одного сундука с драгоценностями.

Начиная школьной запиской и кончая контрольной цифрой на любой купюре, выпущенной Центральным Банком РФ, мы окружены тайнами, секретами и способами их сохранения. Всем знакомы два классических произведения художественной литературы, в которых главной пружиной сюжета являются шифры. Конечно, это "Золотой жук" Эдгара По и "Пляшущие человечки" Конан Дойла. Недаром они относятся к детективному жанру, так как раскрытие даже непритязательного шифра есть задача, сложнее и запутаннее расследования весьма изощренного преступления.

С точки зрения профессиональных криптоаналитиков, шифры вроде того, что разгадал Шерлок Холмс или которому пират Кидд доверил тайну своих сокровищ, ненадежны и неприемлемы для серьезных приложений. Разумеется, нужно соизмерять сложность используемого шифра, ценность шифруемых данных и усилия, которые может приложить охотник до чужих тайн для его взлома. Если вам потребуется так зашифровать свои файлы, чтобы их не смогла прочесть младшая сестра, то можно воспользоваться любым подстановочным шифром. Если вам противостоит серьезная организация или секретная служба крупного государства, то требования к надежности шифра многократно возрастают.

Вот один пример, обнажающий те этические проблемы, которые ставит криптография. В годы второй мировой войны англичанам удалось взломать немецкий шифр Enigma. Между прочим, для этого была построена машина, с полным правом претендующая на звание первого компьютера в мире. Сведения, получаемые в результате радиоперехватов и расшифровки немецких переговоров, представляли исключительную важность. И даже получив информацию о готовящейся бомбежке Ковентри, британские военные власти не стали предпринимать никаких мер, которые могли бы навести противника на мысль, что его планы раскрыты...

В этой статье мы коснемся нескольких аспектов современной криптографии, которые помогут вам создать представление о том, чем занимаются самые секретные отделы ведущих спецслужб мира. Хочется лишь предостеречь тех, кто, не встретив здесь ни одной формулы, сделает неправильный вывод о том, что криптография — это удел программистов и офицеров связи. На самом деле целые разделы теории чисел развиваются, чтобы подкрепить математический аппарат криптографии, а многие статьи невозможно понять без солидной подготовки в алгебре, анализе и статистике.

DES, NSA etc.

Истории создания и взлома шифров могли бы стать основой для увлекательных романов, если бы они не хранились в архивах спецслужб под грифами наивысших уровней секретности. Например, сам факт существования агентства по национальной безопасности — *National Security Agency* (NSA) в США являлся тайной в течение многих лет. Бюджет агентства засекречен и по сей день. Считается, что NSA является крупнейшим в мире работодателем для математиков, так же, как и самым крупным покупателем компьютерного оборудования.

Поэтому, когда под контролем NSA был разработан и предложен в качестве стандарта алгоритм шифрования DES (*Data Encryption Standard*) в 1976 году, возникло необычайно много спекуляций и слухов относительно его надежности. Ведь главная проблема анализа стойкости алгоритмов шифрования заключается в том, что не существует никаких реальных методов формального доказательства того, что данный конкретный алгоритм надежен. С другой стороны, доказать слабость алгоритма весьма просто — для этого достаточно предъявить способ его взлома. Поэтому общепринятым методом подтверждения надежности алгоритма является его публикация и обсуждение научным сообществом. Если в течение достаточно долгого времени анализ авторитетных экспертов или организаций не обнаружил "дырок" в его защите, то такой алгоритм начинает пользоваться доверием.

Данная статья была опубликована в журнале "Компьютерные инструменты в образовании" № 3, 4/99. Информацию о журнале читайте на с. 10.

DES является типичным алгоритмом блочного шифрования; он шифрует блоки по 64 бита. Блок из 64 бит поступает на вход алгоритма, блок той же длины зашифрованных данных является результатом работы алгоритма. DES — это симметричный алгоритм: один и тот же алгоритм и ключ используются для шифровки и расшифровки (за исключением незначительных различий в управлении ключом). Фактическая длина ключа составляет 56 бит (хотя ключ представляется в виде 64-битного числа, каждый восьмой бит используется для контроля четности и при шифровании игнорируется). Ключом может быть любое 56-битное число. В секретности ключа заключается вся защита, предоставляемая алгоритмом, так как, не зная ключа, расшифровать текст невозможно.

На элементарном уровне алгоритм есть всего лишь комбинация двух стандартных методов шифрования: смешивания и распространения. Работа основных блоков, из которых состоит DES, заключается в однократном применении этих методов (подстановки и перестановки) с использованием ключа. Эта операция называется раундом. DES предусматривает 16 раундов, заключающихся в применении одних и тех же методов, использующих разные части ключа.

Стандарт DES в том виде, в котором он был опубликован, вызвал массу вопросов, касающихся его внутренней структуры. Почему были выбраны именно такие логические функции? Почему он совершает ровно 16 раундов? Почему длина ключа лишь 56 бит? Почему, наконец, DES рекомендован к применению только для защиты правительственной информации без грифа "секретно"?

Как оказалось, на многие вопросы у NSA были свои ответы, которые оно предпочло не раскрывать в свое время научной общественности. Например, DES оказался специально защищен от применения дифференциального криптоанализа, предложенного в 1990 году и показавшего себя очень эффективным против других известных алгоритмов. Оказалось, еще в середине 70-х NSA было знакомо с этим методом, но оно не стало вводить его в научный обиход, желая использовать свое преимущество для взлома других способов шифрования. Это дало повод предположить, что NSA, возможно, знало методы взлома DES, которые были встроены в него еще на этапе проектирования. Все эти сомнения, связанные с алгоритмом и позицией его создателей, спровоцировали беспрецедентную атаку со стороны виднейших специалистов в области криптографии на DES. Привлеченное этой проблемой внимание к криптографии и стало фактически одним из катализаторов ее бурного развития в течение последних 20 лет. Тем не менее DES является все еще очень надежным и превосходно зарекомендовавшим себя алгоритмом, для которого существуют очень эффективные аппаратные методы реализации.

Существует ли надежный шифр?

А существуют ли вообще шифры, которые было бы невозможно раскрыть? Оказывается, да. Более того, они уже давно применяются шпионами. Для таких шифров необходимо наличие идентичных копий одноразового блокнота у получателя и отправителя секретного сообщения. Представим себе блокнот, сплошь заполненный абсолютно случайными буквами. Отправитель использует одну букву из блокнота для шифрования одной буквы исходного сообщения. Шифрование состоит в сложении по модулю 32 номеров букв из блокнота и сообщения.

Например, если сообщение представляет собой строку

ПАКЕТВЫКРАЛ,

а соответствующие буквы из блокнота — это

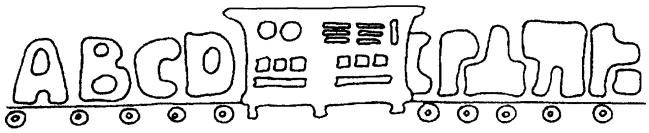
ЕИАКПАФКАЬЯ,

то зашифрованный текст будет

ФИКПБВПФРЬК,

так как

(П + Е) mod 32 = Ф,
(А + И) mod 32 = И
...



К сожалению, у этого абсолютно надежного метода есть несколько существенных недостатков. Во-первых, длина ключа оказывается равной длине шифруемого сообщения. Во-вторых, ключ должен использоваться не больше одного раза (шпионы обязаны уничтожать использованные страницы своих блокнотов), так как в случае повторного использования взлом кода становится весьма простой задачей. В-третьих, у получателя и отправителя должны быть совершенно одинаковые копии этих блокнотов, а если они смогли секретным образом передать такие блокноты, то почему бы им не воспользоваться тем же каналом для передачи самого сообщения? Тем не менее этот метод реально применяется, и в литературе можно встретить утверждение, что в "горячей линии" связи Кремль — Вашингтон используется похожий алгоритм.

Как передать информацию без кодирования

А нужны ли шифры вообще? Можно ли передавать информацию, не используя никакого кодирования? Криптография занимается изучением и таких вопросов, предлагая алгоритмы, с помощью которых можно обмениваться информацией, не раскрывая никаких тайн.

Представим, что в школе провели тестирование IQ. Всем интересно сравнить свой результат с интеллектуальным коэффициентом одноклассников, но никто не хочет раскрывать свой IQ. Оказывается, школьники могут узнать средний IQ в своем классе, при этом ни-

чей результат не раскрывается. Конечно, для этого им достаточно бросить бумажки в шляпу, а потом огласить результаты. А если они уже разъехались на каникулы и поддерживают связь только по электронной почте?

Для простоты будем считать, что среднее своих результатов хотят узнать четыре школьника: Аня, Боря, Вася и Гена. Пусть Аня выберет любое случайное число R, прибавит его к своему результату и сообщит его по секрету Боре. Боря прибавит к полученному числу свой результат и пошлет его Васе втайне от остальных участников. Вася прибавит свой результат и сообщит его Гене. Гена, добавив к числу свой IQ, пошлет его Ане. Аня вычтет из полученного числа R, поделит его на четыре и объявит результат.

Можно заметить, что если все честно следуют правилам, то полученный результат и будет средним IQ в классе. Конечно, Аня и Вася, сговорившись, могут узнать результат Бори. Немного модифицированная версия этого алгоритма (придумайте ее!) уберегает Борю от сговора любых двух участников. В таком алгоритме для выяснения точного результата одного участника требуется сговор всех остальных.

Как честно сыграть в орлянку

Пришло время рассказать одну историю.

Алиса и Боб хотят сыграть в орлянку, но у них нет монетки, которую можно было бы подкинуть. Алиса придумала простой способ подкидывания монетки в уме.

— Давай ты задумаешь случайный бит (0 или 1), я тоже задумываю случайный бит. Если наши биты равны, то выпал орел, если нет, то выпала решка, — предложила Алиса.

— А если кто-то из нас выберет свой бит не случайно? — поинтересовался Боб.

— Не имеет значения. Пока какой-нибудь из наших битов выбирается случайно, результат их сравнения тоже будет случаен, — ответила Алиса после недолгого раздумья. Боб признал правоту Алисы.

Вскоре после этого Алиса и Боб нашли книжку по искусственному интеллекту, валяющуюся на дороге. Хорошая девочка Алиса сказала: "Один из нас должен подобрать книгу и отнести ее в подходящее мусорное ведро". Боб согласился и предложил, чтобы они подкинули монетку в уме, чтобы определить, кто должен понести книгу.

— Если орел, то книгу понесешь ты, если решка, то я, — сказала Алиса, — какой твой бит?

— 1, — ответил Боб.

— У меня тоже, — лукаво сказала Алиса, — мне кажется, тебе сегодня не повезло.

Не нужно объяснять, что в алгоритме подкидывания монетки в уме есть серьезная проблема. В действительности алгоритм, предложенный Алисой, не гарантирует того, что биты, выбираемые игроками, независимы.

По счастью, Алиса и Боб получили письмо от студента, интересующегося криптографией. Содержание письма было слишком сложным, чтобы его кто-то смог понять, но конверт, в котором пришло письмо, оказался очень полезен.

В следующий раз, когда Алиса и Боб захотели подкинуть монетку, они сыграли в несколько модифицированную версию исходного алгоритма. Когда Боб задумал бит, он не стал его сообщать Алисе, а вместо этого записал его на клочке бумаги и положил в конверт. Затем Алиса объявила свой бит. После этого Алиса и Боб открыли конверт и сравнили бит Алисы с записанным битом Боба.

Результат сравнения битов оказывается случайным, если хотя бы один игрок играет честно. Алиса и Боб стали пользоваться таким чудесным алгоритмом и прожили долгую и счастливую жизнь.

Как же реализовать такой алгоритм на компьютере, не пользуясь никакими конвертами и записками? Для этого нам потребуется понятие односторонней функции. Односторонняя функция — это такая функция f, что по значению $f(x)$ вычислить x можно только перебором всех вариантов. Такую функцию построить сравнительно легко. Для этого необходимо предложить алгоритм, который бы полностью перемешивал биты x, применяя к ним достаточно много раз нелинейные преобразования, а множество значений функции состояло бы из нескольких сотен бит. Кроме того, требуется, чтобы было вычислительно очень сложно подобрать такие значения x и y, чтобы выполнялось равенство $f(x) = f(y)$.

Итак, пусть Алиса и Боб согласовали функцию f. Боб загадывает случайное число x, вычисляет $y = f(x)$ и посылает y Алисе. Алиса пытается угадать, четное ли число x, и посылает свою догадку Бобу. Если Алиса угадала, то результатом игры считается орел, если нет, то решка. Боб сообщает результат игры и посылает x Алисе. Алиса проверяет, действительно ли $y = f(x)$.

Видно, что игра оказывается абсолютно честной: Алиса не имеет представления о четности x в силу односторонности функции f, а Боб не может подменить x после того, как Алиса высказала свою догадку, так как подобрать число с тем же значением функции f очень сложно.

Как сравнить записки, не раскрывая их содержимого

Давайте ознакомимся еще с одной историей, герои которой мучаются подозрениями, а искусное использование криптографических протоколов позволяет им совершенно честно узнать то, что они хотят, не скомпрометировав ничье имя.

Соседи по парте Андрей и Борис получили записки, в которых они приглашались на свидание. Зная смешливость девочек своего класса, они заподозрили, что эти записки одинакового содержания и, придя на свидание, они встретят там друг друга.

— Боря, покажи мне свою записку; если там то же, что и у меня, то нам нужно будет кое-кого здорово проучить, — предложил Андрей.

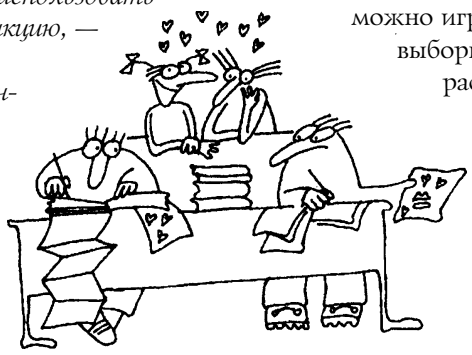
— Нет, лучше ты мне свою покажи, — спрятав получше свою записку, сказал Боря.

— А если там не то же, что у тебя? Не буду я тебе свою записку показывать, — отказался Андрей, надеясь, что у него действительно состоится романтическое свидание.

— А то давай так. Ты вычислишь одностороннюю функцию f от своей записки, а я от своей. После этого мы их сравним и, если они равны, то никуда не пойдем, — предложил выход Боря.

— А если они разные, ты ничего не сможешь узнать про мою записку? — встревожился Андрей.

*— Конечно, нет, ведь мы будем использовать действительно **одностороннюю** функцию, — убежденно сказал Борис.*



Помогая себе громким сопением, Андрей и Борис вычислили значения одной и той же односторонней функции f от текста своих записок, закодированного в двоичной системе счисления. У них получились числа длиной 128 бит.

— А как мы будем сравнивать наши числа? Ведь если кто-то из нас покажет другому свое число, то тот может запросто сказать, что у него число другое, хотя на самом деле они одинаковые, — задумался Андрей.

— И правда, что же делать? Хотя можно было бы вычислить односторонние функции от наших чисел, а потом ими обменяться, но уж больно не хочется еще раз вычислять эти длиннющие функции, — согласился Борис.

— Я придумал! Давай, каждый из нас будет по очереди называть биты своего числа. Я — первый бит, ты — второй, потом я — третий бит и так далее. У каждого из нас будет возможность проверить, совпадают ли наши числа! — воскликнул Андрей.

— Но ведь так мы узнаем только каждый второй бит другого числа. Если они отличаются, то это, конечно, означает, что числа разные, ну а если они равны?

— Вероятность того, что у двух почти случайных чисел совпадут 64 бита, настолько мала, что на нее можно не обращать никакого внимания. А односторонняя функция формирует почти случайные числа, не имеющие ничего общего с исходным текстом, — заверил Борю Андрей.

Так они и сделали, разрешив разом все свои сомнения.

И это все?

Прочитав эту статью о криптографии, вы познакомились с несколькими элементарными примерами, в которых как в капле воды отражаются многие понятия, появившиеся лишь в последние два десятка лет. Конечно, можно было бы также рассказать о том, как можно играть в покер по телефону, проводить выборы по радио (когда все слышат всех), расплачиваться цифровыми деньгами, которые невозможно подделать или скопировать. Но ведь где-то нужно остановиться.

Часть информации для этой статьи была почерпнута из книги Брюса Шнайера "Прикладная криптография", которая пока еще не переведена на русский язык. В этой книге описываются поразительные протоколы и алгоритмы, которые уже начинают применяться в высокотехнологичных устройствах, таких, как сотовые телефоны или "умные" кредитные карточки.

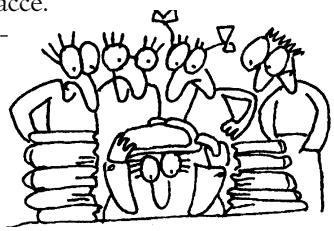
Надеюсь, что вам стала несколько ближе и понятнее криптография — наука о секретах и об алгоритмах, их стерегущих. Хотя конца спирали "алгоритм шифрования — способ взлома" не видно, каждый следующий виток становится еще круче и увлекательнее. Захотите — и вы тоже сможете принять участие в этой гонке.

ЗАДАЧИ

Задачи подготовлены И.Л. Мироновым и П.Г. Черкасовой

ЗАДАЧА 1

Уровень 1. Вспомните школьников, которые выясняли среднее значение IQ в своем классе. У приведенного алгоритма есть один существенный недостаток: если два соседа одного школьника сговорятся, то они смогут определить его IQ. Как следует модифицировать этот алгоритм, чтобы для определения IQ одного школьника был необходим сговор всех остальных?



Именно это число участников (все без одного) и является, как можно заметить, наилучшим возможным значением количества школьников, необходимых для раскрытия тайны своего одноклассника. Действительно, если все школьники без одного выяснят свой средний IQ, то по этому числу и по среднему IQ класса они смогут выяснить интеллектуальный коэффициент выставленного за дверь одноклассника. Таким образом, алгоритм, который вам предла-

гается придумать, предоставляет своим участникам самую высокую возможную защиту.

ЗАДАЧА 2

Уровень 1. Пусть в группе из нескольких человек одному известна некая тайна. Он хочет ее сообщить всем остальным, но хочет, чтобы его личность не была раскрыта. Предложите такой алгоритм, при котором все узнают это анонимное сообщение, но пославший его ничем не будет рисковать.

ЗАДАЧА 3

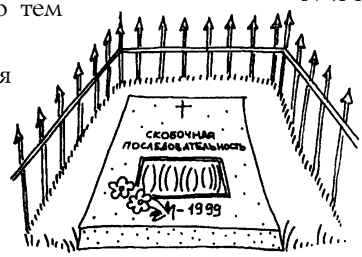
Уровень 1. Считается, что секрет приготовления кока-колы хранится в сейфе, который могут открыть члены совета директоров компании, только собравшись вместе. Руководители компании ведут активный образ жизни, часто летают на самолетах и отдыхают на лыжных курортах. Поэтому следует учесть вероятность того, что в промежутке между заседаниями совета директоров с кем-нибудь из них может произойти несчастный случай. Чтобы обезопасить компанию от безвозвратной утраты секрета, было принято решение, что отныне сейф можно открыть на собрании совета директоров, на котором присутствуют все, кроме, быть может, одного (любого!) его члена. Предложите конструкцию такого сейфа с N замками, в котором было бы как можно меньше движущихся деталей, а открывался бы он любым набором из $N-1$ ключа. Постарайтесь не использовать сложные механизмы с рычагами и блоками, ведь, как известно, чем проще конструкция, тем она надежнее.

ЗАДАЧА 4

Жила-была правильная скобочная последовательность из круглых скобок. Некто взял и написал под каждой открывающейся скобкой число скобок, которые содержатся между ней и соответствующей закрывающейся. После этого некто стер скобочную последовательность. Сможете ли вы ее восстановить по тем числам, которые написал некто?

Уровень 1. Изложите алгоритм для восстановления скобочной последовательности.

Уровень 2. Напишите программу, которая находит скобочную последовательность.



Формат ввода:
Количество чисел M
Первое число
...
M-е число

Формат вывода:
Скобочная последовательность.

Пример

Ввод:	*Вывод:*
6 0 8 0 4 0 0	( ) (( ) (( ) ( )))

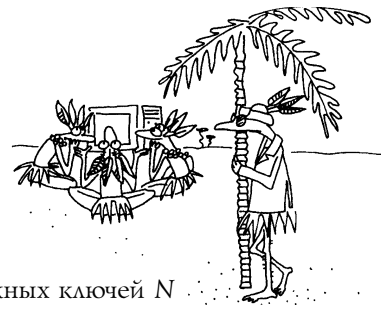
ЗАДАЧА 5

Шпионам из племени мумба-юмба требуется переслать секретный шифр, состоящий из чисел от 1 до 100. Сначала они хотели прибавлять ко всем числам одно и то же задуманное секретное число — ключ (тоже от 1 до 100). Но потом решили, что сложение — это слишком простое действие, а умножение гораздо сложнее и таинственнее. Поэтому они стали умножать все числа на ключевое число.

Агенты из племени юмба-мумба перехватили донесение и пронюхали, что шпионы кодируют донесение с помощью умножения на одно и то же число. Помогите им найти все возможные ключи.

Уровень 1. Опишите алгоритм, с помощью которого, зная донесение, можно найти все варианты ключей, которыми могут пользоваться шпионы.

Уровень 2. Напишите программу, которая решает эту задачу.



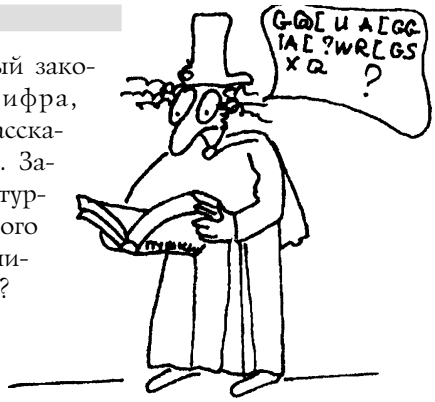
Формат ввода:
Длина донесения M
Первое число
...
M-е число

Формат вывода:
Количество возможных ключей N
Первый ключ
...
N-й ключ

Пример

Ввод:	*Вывод:*
3 1 3 4	1 1

ЗАДАЧА 6



Расшифруйте текст, который закодирован при помощи шифра, аналогичного описанному в рассказе Эдгара По "Золотой жук". Зашифрован отрывок из литературного произведения, из которого были удалены все знаки препинания. Что это за произведение?

```
X Q[G@[U DCY R G[EF YH
YGV[@AWR[U XPUYHW
YPH[ODHHZD EZB[HWDV
YGDHW W G IYVY\WJ G@[AYCY G[EYRHWL[ PDADOHY
WB FLF@ZR[U @DIUYM GYUYVYM UWKY DCY
WQYPA[O[UY GIYLYMG@RWD QEYAYRSD W
EYPAYEF?WD YH VHD YPA[EYR[UGX W G@[U
A[GGIA[?WR[@S YP FO[GHZB IAYWG?DG@RWXB
LYWV X PZU GRWED@DUS X A[GGL[Q[U DVF RGD
G@[AWL GUF?[U VDHX GY RHWV[HWDV W VDOEF
@DV Y@ADQZR[U GFBWD RD@RW PDEHZM VWAYHYR
GL[Q[U YH LYCE[ X LYHNWU GRYJ IDN[USHFJ
IYRDG@S O[US DCY BYAY?WM PZU YTWKDA
```

ЗАДАЧА 7

Муха ползет по квадратной проволочной сетке размером $N \times N$ ($N < 30$) из одного угла (точка (0, 0)) в противоположный (точка (N, N)). Движется муха только в двух направлениях: вверх и вправо. Два паука сидят по краям сетки (в точках (0, N) и (N, 0)) и наблюдают за мухой. Каждый из них записывает, сколько клеток проползла муха, прежде чем повернуться спиной к пауку (при каждом повороте муха поворачивается спиной к одному из пауков).

После этого пауки встречаются и по полученным данным вычисляют путь, по которому ползла муха. (Ведь обратно она поползет тем же путем, и пауки смогут устроить засаду.)

Комарик хочет спасти муху. Но для этого ему тоже нужно знать путь мухи. Он подсмотрел записи пауков. Помогите ему.

Уровень 1. Опишите, как нарисовать путь мухи, если известны записи пауков.

Уровень 2. Напишите программу, которая это делает. Программа должна выводить координаты всех узлов (включая первый, последний и все промежуточные) сетки, которые проползла муха.

Формат ввода:
Сторона квадрата N
Количество чисел в записи первого паука $M1$
Первое число
...
$M1$-е число
Количество чисел в записи второго паука $M2$
Первое число
...
$M2$-е число

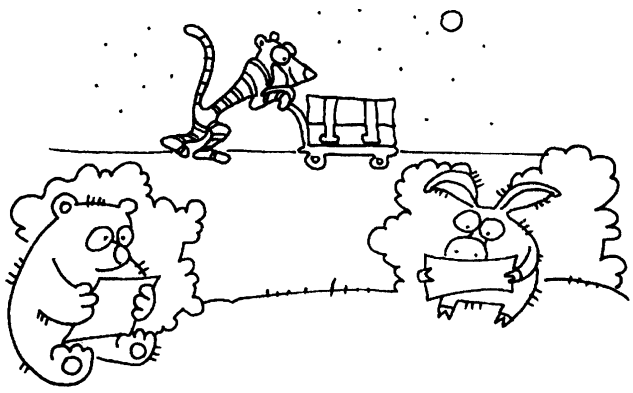
Формат вывода:
Число узлов, которые проползла муха
Координаты первого узла (через пробел)
...
Координаты последнего узла

Пример

Ввод:	*Вывод:*
3	7
2	0 0
2	1 0
1	2 0
2	2 1
2	2 2
1	3 2
	3 3

ЗАДАЧА 8

Винни-Пух и Пятачок занимаются поисками древнего клада. И, чтобы Тигра не раскрыл их планов, они обмениваются зашифрованными записками.



В их шифре используются числа Фибоначчи. Они устроены следующим образом: первые два числа равны единице, а каждое следующее число равняется сумме двух предыдущих: 1, 1, 2, 3, 5, 8, 13, 21, 34, ... Обозначим их f_1, f_2, f_3, ...

Шифр следующий: вместо первой буквы сообщения пишем следующую по алфавиту. Вместо второй — тоже следующую. А дальше вместо каждой буквы пишем букву, увеличенную на очередное число Фибоначчи. То есть вместо четвертой буквы "q" мы пишем "q" + f_4 = "q" + 3 = "t". Если мы уперлись в конец алфавита, то нужно перейти в начало и отсчитывать дальше (напоминаю, что английский алфавит состоит из 26 букв).

Пробелы и знаки препинания так и пишутся. Ввиду важности события сообщения пишутся только заглавными буквами.

Но записки у Винни-Пуха и Пятачка были очень длинные. А числа Фибоначчи растут очень быстро. Поэтому все время у друзей уходило на вычисления и отсчитывание букв, а на поиски клада совсем ничего не оставалось. Заменить шифр они не могли, потому что Тигра усилил бдительность и у них не было возможности обговорить новый шифр.

Уровень 1. Попробуйте придумать способ упростить расчеты так, чтобы времени на зашифровку текста тратилось как можно меньше. Изложите алгоритм такой оптимизации. Зашифруйте текст:

"ISN'T IT FUNNY HOW A BEAR LIKES HONEY?"

Уровень 2. Напишите программу, которая зашифровывает таким образом текст. Программа должна брать текст из файла IN.TXT и записывать зашифрованный текст в файл OUT.TXT.

Формат ввода:	*Формат вывода:*
Исходный текст	Зашифрованный текст

Пример
Ввод:
WINNIE-THE-POOH

Вывод:
XJPQNM-GCM-SZCG

ЗАДАЧА 9

Уровень 1. Злой Карабас-Барабас посадил в два темных подвала Буратино и Мальвину. Он разрешает им обмениваться письмами, но читает их, поскольку опасается, что они договорятся о побеге. К несчастью для Карабаса, пленники заранее договорились о способе передачи секретных сообщений. Чтобы зашифровать такое сообщение (определенную последовательность из нулей и единиц), пленники составляют письмо на правильном русском языке, в котором нулям из секретного сообщения соответствуют слова четной длины, а единицам — нечетной. Знаки препинания (точки, запятые, тире и т.п.) при дешифровке не учитываются. В зашифрованном тексте запрещается:

1) повторять предложения;
2) дважды использовать слово в одном предложении (использовать одно слово в разных предложениях не запрещается).

Напишите программу, которая вводит секретное сообщение — последовательность не более чем из 100 нулей и единиц и выводит его в зашифрованном виде. Зашифрованный текст должен быть составлен по правилам русского языка (т.е. не содержать орфографических, пунктуационных и синтаксических ошибок).

Исходные данные:

Файл содержит одну или несколько секретных последовательностей. Каждая последовательность состоит не более чем из 100 нулей и единиц и записывается на отдельной строке. Файл исходных данных не содержит пробелов и пустых строк.

Выходные данные:

Для каждого секретного сообщения вывести в выходной файл его зашифрованный вариант. Сообщения в выходном файле должны разделяться пустой строкой. Сообщение не должно содержать пустых строк. Каждое зашифрованное сообщение может располагаться на нескольких строках, переносить слова запрещается.

Пример
Файл исходных данных INPUT.TXT:
11000100000100011100101110
0

Выходной файл OUTPUT.TXT:

Мороз и солнце: день чудесный!
Еще ты дремлешь, друг прелестный —
Пора, красавица, проснись:
Открой сомкнуты негой взоры
Навстречу северной Авроры,
Звездою севера явись!
А.С. Пушкин
Вечереет.

Примечание. Различные наборы исходных данных задают различные секретные сообщения, поэтому одно предложение может встречаться в разных зашифрованных текстах.

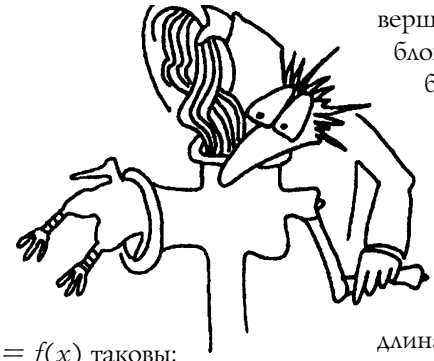
ЗАДАЧА 10

Уровень 1. Придумайте симметричный шифр, то есть такой, чтобы один и тот же алгоритм зашифровывал и расшифровывал текст. То есть если на вход алгоритму дать им же зашифрованный текст, то он его расшифрует.

Уровень 2. Напишите программу, которая реализует этот алгоритм. Программа должна брать текст из файла IN.TXT и записывать зашифрованный текст в файл OUT.TXT.

Формат ввода:
Исходный текст

Формат вывода:
Зашифрованный текст



ЗАДАЧА 11

Уровень 1. Устройте конкурс односторонних функций.

Требования к функции $y = f(x)$ таковы:

— область определения и область значений функции (то есть диапазон изменения x и y) от 0 до 4 294 967 295 ($2^{32} - 1$);

— функция не должна быть слишком сложной и громоздкой. По значению аргумента значение функции должно определяться быстро;

— функция должна быть односторонней — то есть для того, чтобы по значению функции найти значение аргумента, нужно просто перебрать все аргументы.

Уровень 2. Будет хорошо, если, кроме письменного описания функции, вы напишете программу, которая по значению x находит значение $f(x)$.

Формат ввода:
Целочисленное значение аргумента

Формат вывода:
Целочисленное значение функции

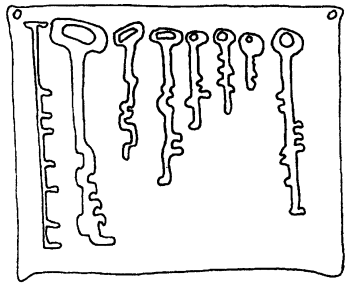
Пример

Ввод:	*Вывод:*
999990	12

ЗАДАЧА 12

Уровень 1. Очень важным этапом проектирования любой криптостойкой системы является обоснованный и тщательно продуманный выбор длины ключа.

Для этого необходимо оценить стоимость и продолжительность атаки методом грубой силы, то есть взлома кода путем полного перебора всех возможных ключей. Представим себе, что в открытую продажу поступил чип (микросхема) стоимостью 10 долларов, совершающий миллион операций шифрования одного блока в секунду. Оцените, сколько времени потребуется начинающему шпиону, чтобы расшифровать ваше сообщение, если вы пользуетесь ключом длины 40 бит, а он готов потратить на это 100 долларов. А если длина ключа 56 бит? 168 бит? (40 бит — это максимальная длина ключа, которая допускается в программных и аппаратных средствах, разрешенных к импорту из США, 56 — длина ключа в обычном DES, 168 бит — длина ключа в троированном DES, то есть один текст шифруется трижды с разными ключами). А если вам противостоит крупная частная организация, которая в состоянии потратить 100 тысяч долларов, лишь бы взломать код? Теперь поставьте себя на место Джеймса Бонда, за которым охотится зловещая организация СПЕКТР. Сколько времени есть у него в запасе, прежде чем его код будет раскрыт, если на это может быть потрачено десять миллионов долларов?



Заполните таблицу, вписав в нее время, требуемое для атаки методом грубой силы с данной длиной ключа и в рамках заданного бюджета.

Как изменятся вычисленные значения через 5 и 15 лет? Для этого воспользуйтесь законом Мура, гласящим, что производительность процессоров удваивается каждые 18 месяцев.

Уровень 2. Выбрав длину ключа (см. предыдущую задачу), следует решить следующую проблему: откуда эти ключи брать?

Алгоритм	Импортированный из США алгоритм	DES	Троированный DES
Длина ключа	40 бит	56 бит	168 бит
Бюджет начинающего шпиона ($100)			
Бюджет банка ($100 тыс.)			
Бюджет спецслужбы ($10 млн)			

Если предлагать пользователю ввести пароль, то количество возможных ключей сильно сокращается. Если

воспользоваться датчиком псевдослучайных чисел используемого языка программирования, то становится возможным поиск ключа с помощью перебора начальных значений этого датчика. Чтобы обойти эти трудности, попробуем извлечь из компьютера как можно больше "почти" случайных чисел, а потом используем их для построения ключа. Например, можно взять количество микросекунд, прошедших с начала секунды, последнюю цифру количества свободных байтов на диске и т.п. Найдите такие числа, из которых можно сформировать ключ длиной хотя бы 40 бит. Может быть, для этого вам потребуется помощь пользователя компьютера.

ЗАДАЧА 13

Уровень 1. Передать одноразовый блокнот для шифрования может оказаться нелегко. Еще труднее его спрятать, а уж обнаружение такого блокнота демаскирует его владельца полностью. Тем не менее можно договориться использовать в качестве такого блокнота какую-нибудь книгу, экземпляры одного издания которой есть у разведчика и в центре. Непосредственное использование букв в книге является не очень удачной идеей, так как буквы в естественных текстах встречаются крайне неравномерно. Придумайте другой способ извлечь из книги абсолютно случайную последовательность.

ЗАДАЧА 14

В занятии 2 (см. № 37/99) было рассказано, что растровую двухцветную картинку можно закодировать с помощью нулей и единиц. В этой задаче предлагается закодировать 256-цветную картинку размером $M \times N$. Причем это нужно сделать так, чтобы итоговый файл с картинкой оказался как можно меньше.

Уровень 1. Опишите алгоритм записи графического изображения.

Уровень 2.

1. Напишите программу, которая записывала бы картинку в придуманном вами формате. Исходные данные находятся в файле IN.TXT, а закодированную картинку следует записывать в файл PICTURE.PIC. Этот файл не обязательно должен иметь текстовый формат (ведь он занимает слишком много места). Будет хорошо (но не обязательно), если ваша программа сумеет при этом показать картинку на экране.

Формат ввода:
Высота рисунка *N*
Ширина рисунка *M*
Цвет первой точки в первом ряду
Цвет второй точки в первом ряду
...
Цвет *M*-й точки в первом ряду
...
...
Цвет первой точки в *N*-м ряду
Цвет второй точки в *N*-м ряду
...
Цвет *M*-й точки в *N*-м ряду

Формат вывода:
Запись картинки в вашем формате.

2. Напишите программу, которая показывала бы на экране картинку, записанную в придуманном вами формате в файл PICTURE.PIC.

Рисунки художника А.В. Васильковой.

Транслятор?.. Это очень просто!

А.А. Дуванов

И все доступно уж, эхма!
Теперь для нашего ума!
Николай Носов
(стихи Цветика)

Продолжение. Начало в № 41, 42/99

Дорогой читатель! Вы еще со мной? Если это так, то усаживайтесь поудобнее и пристегните ремни: виражи сегодня будут очень крутые!

Для начала мы построим примитивный безыдейный транслятор для перевода записи арифметического примера в число — результат вычислений. Это для разминки. Никаких открытий, только здравый смысл.

Затем, разогревшись, мы вступим в сферы перевода на "польский" и вычислений на стеке.

Если вы ограничите свое чтение только "примитивным транслятором", я не обижусь. Ну а если пойдете за мной до конца, буду очень рад!

ПРИМИТИВНЫЙ ТРАНСЛЯТОР

Вторая часть этих публикаций (№ 42/99) содержала конкурсную задачу Корректора под номером 4. Рассмотрим ее решение.

ЗАДАЧА 4 (9 БАЛЛОВ)

Введем следующее определение:

Определение

```
<выражение>::=<число>|<выражение>+<число>
<число>::=0|1|2
```

Задание. Напишите программу, которая вычисляет выражение, если оно правильное, или записывает на ленту сообщение "ош", если запись содержит ошибку. Если запись ошибочная, то дополнительно к сообщению "ош" окошко должно указывать на первый неверный символ.

Замечание 1. Значение выражения должно быть записано на ленте в виде обычного целого десятичного числа.

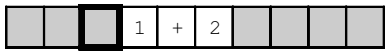
Замечание 2. Предполагается, что результат вычисления выражения не превышает длины алфавита Корректора.

В начальный момент окошко расположено перед записью.

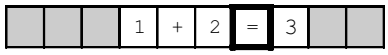
Примеры работы программы:

1

Начальное состояние



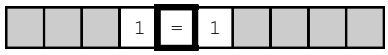
После работы программы



2

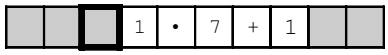
Начальное состояние

После работы программы

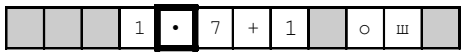


3

Начальное состояние

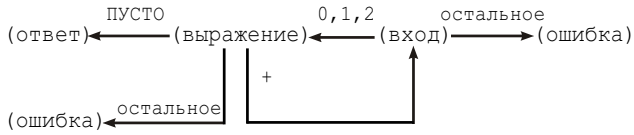


После работы программы



Решение

Диаграмма переходов здесь очень проста:



Однако нам нужно не просто проверить выражение, но и вычислить его. Поэтому в программе появляются дополнительные процедуры, связанные не с переходом в другие состояния алгоритма, а с вычислениями.

ПРОГРАММА ВЫЧИСЛЕНИЯ ВЫРАЖЕНИЯ

```
ЭТО вход0                       Эта заглавная процедура подготавливает ящик для
  ОБМЕН                         вычислений перед входом в состояние (вход) алгоритма.
  ПИШИ ПУСТО
  ОБМЕН                         Подготовка состоит в записи символа ПУСТО в ящик.
  вход                          Эту ячейку памяти Корректора будем использовать как
КОНЕЦ                           сумматор.
```

```
ЭТО вход                          Состояние (вход)
  ВПРАВО                          Смотрим следующий символ.
  ЕСЛИ 0 ТО выражение             Если это 0, переходим в состояние (выражение)
  ИНАЧЕ ЕСЛИ 1 ТО добавить1       без вычислений. Если 1 или 2, перед переходом
  ИНАЧЕ ЕСЛИ 2 ТО добавить2       в состояние (выражение) "добавим" число в ящик.
  ИНАЧЕ ошибка                    Любой другой символ в состоянии (вход) — ошибка.
КОНЕЦ

ЭТО выражение                     Состояние (выражение)
  ВПРАВО                          Смотрим следующий символ.
  ЕСЛИ  + ТО вход                 Если это "+", переходим в состояние (вход) —
  ИНАЧЕ ЕСЛИ ПУСТО ТО ответ       выражение продолжается.
  ИНАЧЕ ошибка                    Если это ПУСТО, выражение закончилось.
КОНЕЦ                             Любой другой символ — ошибка.

ЭТО добавить1                     Вспомогательная переходная процедура
  ОБМЕН                           между состояниями (вход) и (выражение).
  ПЛЮС                            В ней к ящику "добавляется" 1.
  ОБМЕН                           Конечно, команда ПЛЮС не арифметическая, она
  выражение                       просто заменяет символ на следующий по алфавиту
КОНЕЦ                             Корректора. Но, как увидим дальше, это действие
                                  совершенно эквивалентно операции "+1".

ЭТО добавить2                     Вспомогательная переходная процедура
  ОБМЕН                           между состояниями (вход) и (выражение).
  ПЛЮС ПЛЮС                       В ней к ящику "добавляется" 2.
  ОБМЕН
  выражение
КОНЕЦ

ЭТО ошибка                        Идем от ошибочного символа до конца записи, сжимая
  ВПРАВО                          рекурсивную пружину.
  ЕСЛИ ПУСТО                      Записываем в конце записи сообщение об ошибке
  ТО ответ_ошибка                 и под воздействием "энергии" закрученных
  ИНАЧЕ ошибка                    витков возвращаем окно на место ошибки.
  ВЛЕВО
КОНЕЦ

ЭТО ответ_ошибка                  Сообщение об ошибке.
  ВПРАВО ПИШИ о
  ВПРАВО ПИШИ ш
  ПОВТОРИ 2 ВЛЕВО                 Переместиться назад на 2 клетки необходимо для того,
КОНЕЦ                             чтобы точно попасть "под пружину".

ЭТО ответ                         Запись ответа в виде числа на ленте.
  ВПРАВО ВПРАВО                   Алгоритм превращения символа из ящика в десятичное
  ПИШИ 0                          число на ленте:
  ОБМЕН                           — записать на ленту символ 0
  ПОКА НЕ ПУСТО                   — ПОКА в ящике НЕ ПУСТО, делать:
  {                                 — символ в ящике заменить на предыдущий
    МИНУС ОБМЕН                     — прибавить к числу на ленте 1
    плюс1
    ПОКА НЕ ПУСТО ВПРАВО          Процедура "плюс1" добавляет к числу
    ВЛЕВО ОБМЕН                   на ленте 1.
  }
  ОБМЕН                           Последний дизайнерский штрих:
  ПОКА НЕ ПУСТО ВЛЕВО             знаком "=" отделим результат от примера.
  ПИШИ =
КОНЕЦ

ЭТО плюс1                         Процедура добавления единицы к числу на ленте.
  ЕСЛИ 9 ТО девять                В начальный момент окошко должно быть
  ИНАЧЕ ПЛЮС                      установлено на последний символ числа.
КОНЕЦ
```

```
ЭТО девять
  ПОКА 9 { ПИШИ 0 ВЛЕВО }
  ЕСЛИ НЕ ЦИФРА
  ТО ПИШИ 1
  ИНАЧЕ ПЛЮС
КОНЕЦ
```

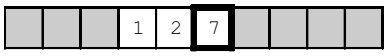
Разбирая работу процедуры, не забывайте, что команда ПЛЮС преобразует цифру "8" в цифру "9", но цифру "8" в букву "а"!
(Смотрите алфавит Корректора и его систему команд.)

Пожалуй, самое интересное место в этой программе — процедура плюс1. Она добавляет к целому числу на ленте единицу.

Вот примеры ее работы:

1

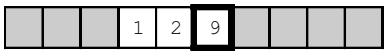
Начальное состояние



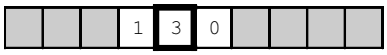
После работы процедуры

2

Начальное состояние

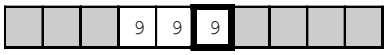


После работы процедуры

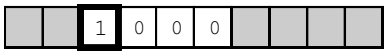


3

Начальное состояние



После работы процедуры



Заметьте, что процедура плюс1 работает на основе цикла ПОКА, хотя можно было бы предложить и рекурсивный вариант.

Однажды эта задача решалась в семье Куков, знакомых читателям "Информатики" по публикации "Введение в Интернет" ("Информатика", № 21/99).

Давайте послушаем (из любопытства), как Куки строят свое рекурсивное решение.

Папа. Научим Корректора выполнять арифметические операции над числами!

Вася. В его СКИ нет нужных команд.

Папа. Сделаем это при помощи хитрости.

Петя. А какие числа мы будем рассматривать?

Папа. Давайте займемся арифметикой целых неотрицательных чисел и, чтобы не говорить каждый раз так длинно, будем называть их просто числами. Вот первая задача.

> **Задача**. К числу на ленте прибавить единицу. В начальный момент в окно видна младшая цифра.

Вася. Ну, такую задачу мы уже решали. Решением является единственная команда ПЛЮС.

Папа. В той задаче, которую мы решали, говорилось о сложении единицы с четным числом, и это было не случайно.

Петя. Проблема заключается в возможном окончании числа на девять, ведь в этом случае команда ПЛЮС запишет на место девятки букву "а" (она идет следом за символом "9" в алфавите исполнителя).

Вася. Да, я поспешил... А что же делать, если младшая цифра у числа —девять?

Петя. Ну, понятно, вместо нее надо записать ноль, затем необходимо прибавить единицу к следующему разряду или, что то же самое, прибавить единицу к исходному числу без последней цифры.

Папа. Посмотри, решая исходную задачу, ты снова пришел к ней же самой, но с меньшим количеством цифр в числе. Как называется такой прием?

Петя. Конечно, это рекурсия!

Вася. Пусть число на ленте равно 239. Вместо цифры "9" записываем ноль и прибавляем 1 к числу 23. Получается 240.

Папа. А если на ленте число 299?

Вася. Самую младшую девятку заменим нулем и прибавим единицу к числу 29. Здесь та же проблема: девятку в числе 29 заменим нулем и прибавим один к числу 2. В итоге получается 300!

Петя. Я придумал красивую схему. Давайте обозначим нашу задачу так:

```
плюс1(299)
```

Эта запись показывает, что нужно прибавить один к числу 299. Теперь решение можно записать так:

```
плюс1(299)=плюс1(29)0=плюс1(2)0 0=300
```

Ведь известно, что

```
плюс1(...0)=...1
плюс1(...1)=...2
плюс1(...2)=...3
....
плюс1(...8)=...9
```

Папа. Схема мне понравилась, в ней очень хорошо видна рекурсивность. А как решить задачу для числа 99?

Петя. Запишем так:

```
плюс1(99)=плюс1(9)0=плюс1()0 0
```

Понятно! Нужно предусмотреть в решении "плюс один от пустого места", и результат должен равняться числу один. Пустое место получается как бы равнозначно нулю:

```
плюс1()=1, и тогда
плюс1(99)=плюс1(9)0=плюс1()0 0=100
```

Папа. Что-то я не слышу ничего от Васи!

Вася. Я успеваю только с трудом следить за вашими схемами... Но, кажется, я все понял. В качестве доказательства записываю программу для Корректора:

```
ЭТО плюс1
 ЕСЛИ НЕ 9 ТО пЛЮС// Если последняя цифра не 9,
 ИНАЧЕ             // задача решается командой ПЛЮС.
 {                 // Когда число заканчивается
   ПИШИ 0          // на 9, заменяем девятку
   ВЛЕВО           // нулем и добавляем 1 к
   плюс1           // "сокращенному" числу.
 }
КОНЕЦ
```

Петя. В твоей программе упущен случай, когда число состоит из одних девяток.

Вася. Да, действительно. Такое число приводит к необходимости обработать пустую клетку:

```
ЭТО плюс1
  ЕСЛИ НЕ 9      // Если последняя цифра
  ТО             // не 9, то ...
  {
    ЕСЛИ ПУСТО   // Если пусто, то
    ТО ПИШИ 1    // записываем 1,
    ИНАЧЕ ПЛЮС   // иначе заменяем на
  }              // следующий по алфавиту символ.
  ИНАЧЕ
  {              // Когда число заканчивается
    ПИШИ 0       // на 9, заменяем девятку
    ВЛЕВО        // нулем и добавляем один к
    плюс1        // числу слева.
  }
КОНЕЦ
```

Папа. По последнему решению нет никаких замечаний.

Петя. Вася, как ты думаешь, сколько раз надо запустить нашу программу, чтобы убедиться в правильности ее работы?

Вася. Думаю, минимум 3 раза. Первый раз для числа, которое не оканчивается девяткой, например, для числа 458; второй раз для числа, у которого первая цифра не девять, а последние — девятки, например, для числа 57 999; и третий раз для числа, состоящего из одних девяток, например, для числа 9999. Тем самым мы проверим все ветвления нашей программы.

Петя. Молодец! Проверка программ называется тестированием. Тестирование должно быть спланировано по возможности так, чтобы в программе проработали все ее части во всех возможных комбинациях. Исходные данные, которые обрабатывает проверяемая программа, называются тестами. Для каждого теста нужно подготовить контрольный результат, который затем сравнивается с результатом работы программы. Обычно результаты тестирования оформляют в виде таблицы. Для твоих тестов таблица будет иметь вид:

Тест	Что проверяется	Контрольный результат	Результат по программе
458	Младший разряд не 9	459	
57 999	Число заканчивается девятками	59 000	
9999	Число состоит из одних девяток	10 000	

Вася занялся тестированием программы, и все тесты прошли хорошо: результаты программы совпали с контрольными результатами.

Петя. Тестирование — это очень важный элемент труда программиста. Ведь в программе очень легко допустить ошибку и не заметить этого. Ошибка может затаиться в какой-нибудь очень редко работающей ветви программы, предусмотренной для обработки особого случая. Представь себе, что твоя программа является элементом какой-нибудь очень важной системы, управляющей космическим полетом, и ветвь, обрабатывающая одни девятки, содержит ошибку. На вход программы поступают числа, которые она должна увеличивать на один. Вероятность того, что появится число из одних девяток, очень мала. Программа долгое время будет работать прекрасно, но, когда на вход поступит 99, аппарат собьется с курса и упадет на Солнце.

Вася. Ну а если программу протестировали, то теперь-то уж все в порядке и можно спать спокойно?

Петя. Увы! Программист никогда не спит спокойно. Представь себе, что обработку одних девяток ты не предусмотрел в алгоритме. Тогда в программе эта ветвь будет отсутствовать и, конечно, "девяточный" тест ты заготавливать просто не будешь. Остальные тесты пройдут нормально, ты уснешь спокойно, а космический корабль упадет на Солнце.

Вася. Вот это да! А как же проверять программы и избавляться от ошибок?

Папа. Теме тестирования программ был посвящен один из моих репортажей. Обычно это происходит так. Сначала программу проверяет автор. Когда он исправляет в ней "последнюю" ошибку, то передает ее эксперту — специалисту по тестированию. В коллективе, занимающемся разработкой программ, такие специалисты просто необходимы. Они, подобно музыкальным или литературным критикам, имеют особый склад мышления, направленный на выявление самых незаметных, глубоко спрятанных изъянов и погрешностей. Когда специалист по тестированию выдыхается (все его самые изощренные тесты проходят нормально), программа поступает в опытную эксплуатацию, где длительное время обрабатывает данные, приближенные к реальным. Только потом она передается потребителю, и все равно в ней, как правило, остается некоторое количество ошибок.

Вася. Вот это да... Все так безнадежно?

Папа. Увы, все большие программы именитых солидных фирм, с которыми я постоянно работаю, — и компьютерная оболочка Windows, и текстовый процессор Word, и табличный процессор Excel — время от времени работают неправильно или попросту "зависают", то есть перестают реагировать на нажатие клавиш и перемещение мыши. В этом случае помогает только перезагрузка компьютера.

Вася. Иными словами, как ни тестируй, а космический аппарат все равно упадет на Солнце?

Папа. В очень ответственных случаях вычисления выполняются одновременно по нескольким разным программам, написанным по разным алгоритмам, и в качестве результата принимается тот, который выдали большинство программ.

Извините, дорогой читатель, что не прервал моих увлеченных Куков и они незаметно перешли от "плюс один" к вопросам тестирования программ! Это потому, что третьего дня мой Windows не просто завис, а умер! Пришлось его устанавливать заново. Справедливости ради хочу отметить, что Windows погиб не сам по себе, а из возникшего "плохого" блока на винчестере. Два дня из моей жизни выпали на реанимацию компьютера. Кто-то сказал, что вопрос не в том, "полетит" винт или нет, а вопрос: когда? А вы хорошо подготовились к полету? Копии ваших данных хранятся в надежном месте?

Материалы сборника

Оценка качества подготовки выпускников основной школы по информатике

Сборник подготовили:
А.А. Кузнецов, Л.Е. Самовольнова, Н.Д. Угринович

Продолжение. Начало в № 38, 39, 41, 42, 43/99

ВАРИАНТ 5

1. В настоящее время во всем мире количество серверов Интернета насчитывает около...

1) *400 тыс.* 3) *40 млн*
2) *4 млн* 4) *400 млн*

2. Чему равен 1 Кбайт?

1) 2^{10} *байт* 3) *1000 бит*
2) 10^3 *байт* 4) *1000 байт*

3. Количество информации, которое требуется для двоичного кодирования 256 символов, равно...

1) *1 бит* 3) *1 Кбайт*
2) *1 байт* 4) *1 бод*

4. Как записывается десятичное число 3 в двоичной системе счисления?

1) *00* 3) *01*
2) *10* 4) *11*

5. Какое устройство обладает наименьшей скоростью обмена информацией?

1) *CD-ROM-дисковод*
2) *жесткий диск*
3) *дисковод для гибких дисков*
4) *микросхемы оперативной памяти*

6. Заражение компьютерными вирусами может произойти в процессе...

1) *печати на принтере*
2) *работы с файлами*
3) *форматирования дискеты*
4) *выключения компьютера*

7. Задан полный путь к файлу

C:\DOC\PROBA.TXT

Каково имя каталога, в котором находится файл PROBA.TXT?

1) *DOC* 3) *C:\DOC\PROBA.TXT*
2) *PROBA.TXT* 4) *TXT*

8. Генеалогическое древо семьи является...

1) *табличной информационной моделью*
2) *иерархической информационной моделью*
3) *сетевой информационной моделью*
4) *предметной информационной моделью*

9. Какое из слов является командой исполнителя Черепашка?

1) *линейный* 3) *алгоритм*
2) *программа* 4) *вперед*

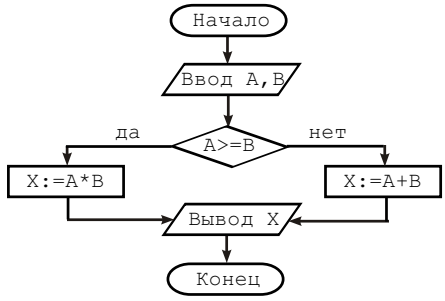
10. Алгоритм какого типа записан на алгоритмическом языке?

1) *циклический*
2) *разветвляющийся*
3) *вспомогательный*
4) *линейный*

```
алг сумма (вещ A, B, S)
  арг A,B
  рез S
нач
 | S:=A + B
кон
```

11. При исходных данных A:=5, B:=4 определите результат выполнения алгоритма, изображенного в виде блок-схемы.

1) X = 20
2) X = 9
3) X = 5
4) X = 4



12. Каково будет значение переменной X после выполнения операций присваивания:

X:=5;
B:=10;
X:=X + B

1) *5* 3) *15*
2) *10* 4) *20*

Таблица номеров правильных ответов на вопросы теста для 5-го варианта

№ вопроса	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
№ прав. ответа	3	1	2	4	3	2	1	2	4	4	1	3	2	3	1	2	4	3	3	2	2	4	1	1

13. В текстовом редакторе основными параметрами при задании параметров абзаца являются...

1) *гарнитура, размер, начертание*
2) *отступ, интервал*
3) *поля, ориентация*
4) *стиль, шаблон*

14. Каково наиболее распространенное расширение в имени текстовых файлов?

1) *EXE* 3) *TXT*
2) *BMP* 4) *COM*

15. Минимальным объектом, используемым в растровом графическом редакторе, является...

1) *точка экрана (пиксель)*
2) *объект (прямоугольник, круг и т.д.)*
3) *палитра цветов*
4) *знакоместо (символ)*

16. В процессе преобразования растрового графического файла количество цветов уменьшилось с 65 536 до 256. Во сколько раз уменьшится информационный объем файла?

1) *в 2 раза* 3) *в 8 раз*
2) *в 4 раза* 4) *в 16 раз*

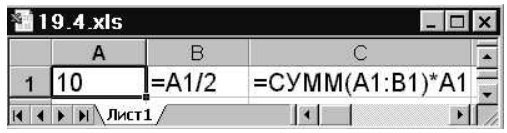
17. Наибольший информационный объем будет иметь файл, содержащий...

1) *1 страницу текста*
2) *черно-белый рисунок 100×100*
3) *аудиоклип длительностью 1 мин.*
4) *видеоклип длительностью 1 мин.*

18. В электронных таблицах формула не может включать в себя...

1) *числа*
2) *имена ячеек*
3) *текст*
4) *знаки арифметических операций*

19. Результатом вычислений в ячейке `C1` будет:



1) *50* 3) *150*
2) *100* 4) *200*

20. Сколько в предъявленной базе данных полей?

Компьютер	Опер. память	Винчестер
Pentium	16	2Гб
386DX	4	300Мб
486DX	8	800Мб
Pentium II	32	4Гб

1) *4* 2) *3* 3) *2* 4) *1*

21. Какую строку будет занимать запись “Pentium” после проведения сортировки по возрастанию в поле **Компьютер**?

Компьютер	Опер. память	Винчестер
Pentium	16	2Гб
386DX	4	300Мб
486DX	8	800Мб
Pentium II	32	4Гб

1) *4* 2) *3* 3) *2* 4) *1*

22. Скорость передачи информации по магистральной оптоволоконной линии обычно составляет не меньше, чем...

1) *56,6 Кбит/с* 3) *28,8 бит/с*
2) *100 Кбит/с* 4) *1 Мбит/с*

23. Серверы Интернета, содержащие файловые архивы, позволяют...

1) *скачивать необходимые файлы*
2) *получать электронную почту*
3) *участвовать в телеконференциях*
4) *проводить видеоконференции*

24. Компьютер, подключенный к Интернету, обязательно имеет...

1) *IP-адрес* 3) *домашнюю web-страницу*
2) *web-сервер* 4) *доменное имя*

ВАРИАНТ 6

1. В современном мире быстрее всего растет протяженность…

1) *автомобильных дорог*
2) *авиарейсов*
3) *железных дорог*
4) *линий связи*

2. Чему равен 1 Гбайт?

1) 2^{10} *Мбайт*
2) 10^3 *Мбайт*
3) *1000 Мбит*
4) *1 000 000 Кбайт*

3. Для двоичного кодирования цветного рисунка (256 цветов) размером 10×10 точек требуется…

1) *100 бит*
2) *100 байт*
3) *400 бит*
4) *800 байт*

4. Как записывается десятичное число 2 в двоичной системе счисления?

1) *00*
2) *10*
3) *01*
4) *11*

5. Процессор обрабатывает информацию…

1) *в десятичной системе счисления*
2) *в двоичном коде*
3) *на языке Бейсик*
4) *в текстовом виде*

6. Заражению компьютерными вирусами могут подвергнуться…

1) *только программы*
2) *графические файлы*
3) *программы и документы*
4) *звуковые файлы*

7. Задан полный путь к файлу

C:\DOC\PROBA.TXT

Каково расширение файла, определяющее его тип?

1) *C:\DOC\PROBA.TXT*
2) *DOC\PROBA.TXT*
3) *PROBA.TXT*
4) *TXT*

8. Информационной (знаковой) моделью является…

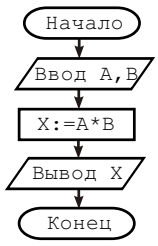
1) *анатомический муляж*
2) *макет здания*
3) *модель корабля*
4) *диаграмма*

9. Алгоритм является…

1) *предметной информационной моделью*
2) *статической информационной моделью*
3) *динамической информационной моделью*
4) *табличной информационной моделью*

10. Алгоритм какого типа изображен на блок-схеме?

1) *Циклический*
2) *Разветвляющийся*
3) *Вспомогательный*
4) *Линейный*



11. По записанному на алгоритмическом языке алгоритму подсчитать сумму квадратов последовательности натуральных чисел.

1) *S=15*
2) *S=18*
3) *S=36*
4) *S=29*

```
алг сумма квадратов (цел S)
    рез S
нач нат n
|   S := 0
|   для n от 2 до 4
|   нц
|   |   S := S + n*n
|   кц
кон
```

12. Значением логической переменной может являться:

1) *любое число*
2) *любой текст*
3) *истина или ложь*
4) *таблица*

Таблица номеров правильных ответов на вопросы теста для 6-го варианта

№ вопроса	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
№ прав. ответа	4	1	3	2	2	1	4	4	3	4	4	3	1	3	4	2	1	3	3	2	4	4	1	2

13. В текстовом редакторе основными параметрами при задании шрифта являются...

1) *гарнитура, размер, начертание*
2) *отступ, интервал*
3) *поля, ориентация*
4) *стиль, шаблон*

14. В процессе форматирования текста изменяется...

1) *размер шрифта*
2) *параметры абзаца*
3) *последовательность символов, слов, абзацев*
4) *параметры страницы*

15. Растровый графический редактор предназначен для...

1) *создания чертежей*
2) *построения графиков*
3) *построения диаграмм*
4) *создания и редактирования рисунков*

16. В процессе сжатия растровых графических файлов по алгоритму JPEG его информационный объем обычно уменьшается в...

1) *2 — 3 раза*
2) *10 — 15 раз*
3) *100 раз*
4) *не изменяется*

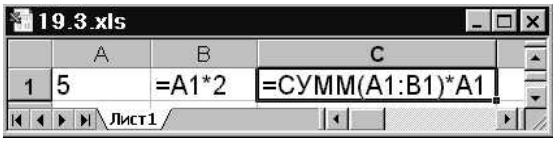
17. Информационная емкость стандартных CD-ROM-дисков может достигать...

1) *650 Мбайт*
2) *1 Мбайт*
3) *1 Гбайт*
4) *650 Кбайт*

18. В электронных таблицах имя ячейки образуется...

1) *из имени столбца*
2) *из имени строки*
3) *из имени столбца и строки*
4) *произвольно*

19. Результатом вычислений в ячейке `C1` будет:



1) *25* 2) *50* 3) *75* 4) *100*

20. Сколько в предъявленной базе данных текстовых полей?

Компьютер	Опер. память	Винчестер
Pentium	16	2Гб
386DX	4	300Мб
486DX	8	800Мб
Pentium II	32	4Гб

1) *1* 3) *3*
2) *2* 4) *4*

21. Какие записи будут найдены после проведения поиска в текстовом поле **Компьютер** с условием **содержит** DX?

Компьютер	Опер. память	Винчестер
Pentium	16	2Гб
386DX	4	300Мб
486DX	8	800Мб
Pentium II	32	4Гб

1) *2* 3) *1,4*
2) *3* 4) *2,3*

22. Модем — это...

1) *почтовая программа*
2) *сетевой протокол*
3) *сервер Интернета*
4) *техническое устройство*

23. Задан адрес электронной почты в сети Интернет: user_name@mtu-net.ru
Каково имя домена верхнего уровня?

1) *ru*
2) *mtu-net*
3) *user_name*
4) *mtu-net.ru*

24. Web-страницы имеют формат (расширение)...

1) *TXT* 3) *DOC*
2) *HTM* 4) *EXE*

Тесты по информатике. Послесловие

От редакции

В № 42, 43, 44/99 мы опубликовали варианты тестов, входящих в сборник "Оценка качества подготовки выпускников основной школы по информатике". Редакция считает необходимым явно обозначить свою позицию по отношению к этой публикации.

Во-первых, мы стремимся познакомить с данными тестами наших читателей, поскольку они входят в указанный сборник, являющийся официальным изданием Министерства образования (и не исключено, что именно эти материалы будут использовать региональные органы образования для проведения различных контрольных мероприятий).

Во-вторых, мы считаем, что уровень данных тестов столь низок, что использовать их в существующем виде категорически недопустимо (маловероятно, что изменения, которые, возможно, будут внесены в тесты при подготовке книжного варианта сборника, существенно изменят ситуацию). Более всего огорчает то, что тесты эти существуют в различных редакциях уже не первый год (см., например, № 1, 41/98). За это время к ним было высказано множество критических замечаний, но внесенные поправки носили исключительно косметический характер. Общий уровень тестов, к сожалению, не изменился.

Мы попросили кратко прокомментировать данные тесты ведущего российского специалиста в области тестирования, доктора психологических наук, научного руководителя центра тестирования МГУ "Гуманитарные технологии" Александра Георгиевича Шмелева.

Замечания к тестам по информатике, вошедшим в сборник "Оценка качества подготовки выпускников основной школы по информатике"

А.Г. Шмелев

Крайне неудачные тесты по информатике, подвергавшиеся уже не раз справедливой критической оценке, внесли свой вклад в то, что информатика "вылетела" из состава вступительных экзаменов в большинстве вузов. А жаль, ведь при грамотном подходе именно тесты по информатике максимально помогли бы раскрыть уровень развития способности учащихся к выполнению таких операций **содержательно-логического** мышления, которые не могут раскрыть тесты по математике (в силу сложившегося крайне консервативного формалистического подхода к преподаванию этой дисциплины в школе).

Не стали исключением в ряду неудачных тестов, увы, и данные. Конечно, было бы неправильно сказать, что в этих тестах нет "ни одного живого места" и их нужно выкинуть целиком. Нет, немало тестовых заданий можно считать вполне добротными, но наличие большого количества неудачных заданий и общий крен в сторону формализма (оценки внешних признаков знания, но не реального операционального мышления) делают эти тесты в целом малопригодными для тех целей, на какие они претендуют.

Оставим в стороне содержательно-тематическую концепцию, ибо она задана, очевидно, не авторами тестов, а нынешними стандартами в области информатики, сконцентрируем замечания на ошибках в тестовой технологии.

Мало заданий

24 задания — это примерно в 2 раза меньше, чем требуется. Основания — известные и измеренные многократно за рубежом и в нашей стране коэффициенты надежности для тестов достижений. Эти коэффициенты не достигают приемлемых значений (0,9 и выше) при тестах с таким малым количеством заданий. Особенно если эти тесты содержат задания по такой новой, "неустоявшейся" и стремительно развивающейся дисциплине, как информатика, где еще не накоплена даже минимальная культура тестирования, нет банка хороших заданий и т.п.

Некорректные формулировки заданий и ответов, не дающие явного преимущества какому-либо одному ответу

Например, строго говоря, и биты, и байты, и килобайты являются "единицами измерения количества информации", а вопрос номер 2 в первом варианте следовало бы задавать про "единицу минимального объема", а не единицу вообще. Другой пример: в задании 9 (вариант 2) совсем не очевидно, что менее правильным является ответ 1, ибо в правилах техники безопасности элементы алгоритма (предписания, сопряженные с условиями) также могут встречаться, как и в инструкции по пользованию банкоматом. Хотя

вообще-то здесь правильный ответ отыскивается совершенно формально: ученик может никогда не видеть ни одного банкомата, но знать, что слово "инструкция" ближе всего к слову "алгоритм" (формально-ассоциативный поиск правильного ответа). Еще один "впечатляющий" пример — вопрос 24 в варианте 3. Приведем его:

Гиперссылки на web-странице могут обеспечить переход...

1) на любую web-страницу любого сервера Интернета
2) на любую web-страницу в пределах данного домена
3) на любую web-страницу в пределах данного сервера
4) в пределах данной web-страницы

Совершенно очевидно, что авторы теста правильным вариантом считают первый. Но не менее очевидно, что три оставшихся также верны. Таким образом, *все четыре ответа являются верными.*

Неоптимальный уровень трудности

Есть задания, которые (по нашему опыту) должны выполнять 95% школьников, изучавших информатику. Следовательно, эти задания — просто балласт, они не несут с собой никакой информации, и это уже смахивает на обычную учительскую "выводиловку" (когда надо повысить процент успеваемости, в данном случае — средний процент правильных ответов по тесту). Например, и ежу должно быть понятно, что магнитные диски надо прежде всего защищать от магнитных полей, а не от "атмосферного давления" (задание 6, вариант 3).

Формальные признаки правильного ответа

Когда правильный ответ является просто более развернутой формулировкой, то школьнику не надо быть семи пядей, чтобы выбрать именно его. Например, правильное определение гипертекста оказывается просто более развернутым, длинным, чем дистракторы (*отвлекающие ответы*) в задании 24 (вариант 1).

Явно неправдоподобные дистракторы, не соответствующие элементарному здравому смыслу, никак не связанному со знаниями в области информатики

Таких примеров на основании данных тестов можно привести великое множество. Например, в задании 22 (вариант 1) и так понятно, что модем не должен передавать 2 странички текста за 1 час или за 1 день, ибо это делает практически бессмысленным его использование. Выбор при этом сужается в лучшем случае до двух альтернатив. И тогда вероятность случайного угадывания правильного ответа доходит до 50%, что уже недопустимо. Ну кто даст исполнителю Черепашка такие *команды* — "линейный", "программа", "алгоритм"? (Таким образом, *все* дистракторы в этом задании являются формально неправдоподобными, и не остается просто ничего другого, как выбрать правильный ответ — "вперед".)

Бессодержательные задания

Согласитесь, если мы будем спрашивать, какого цвета обложка у школьного учебника информатики, то правильный ответ не гарантирует нам, что школьник открывал этот учебник. Пример подобного задания: "Количество кодировок русского алфавита..." (задание 14, вариант 1). Мало-мальский здравый смысл подсказывает: "называй больше, не ошибешься". Кстати, кто их вообще смог подсчитать? И кто установит, какую из них считать "общепринятой"? И вообще: что из того, если школьник знает, что кодировок существует много? Ведь знание различных марок автомобиля ничего не говорит о глубине познания в автоделе, а только "слегка коррелирует" с этими познаниями. Другой "блестящий" пример абсолютно бессодержательного задания — про 24-скоростной CD-ROM: элементарное чувство языка должно помочь человеку, ничего не знающему про то, что такое CD-ROM, найти правильный ответ, который выглядит просто откровенной подсказкой ("*имеет в 24 раза большую скорость вращения диска, чем односкоростной CD-ROM*"). Такое задание можно было бы сформулировать про 24-скоростную мясорубку, автомобиль, синхрофазотрон, "шпагоглотатель" и т.п.

Наш опыт проведения в прошлом году конкурса авторов тестовых заданий (в котором приняли участие 212 авторов и авторских коллективов из 127 городов страны, в том числе свыше 30 по информатике) показал: если по всем предметам наши учителя и методисты составляют задания просто плохо, то по информатике — из рук вон плохо. Недаром конвергентная валидность (*согласованная достоверность*) между данными организуемого нами "Телетестинга" и централизованного тестирования самая низкая именно по информатике (коэффициент линейной корреляции стандартизованных баллов — только 0,47; для сравнения: по физике — 0,77). Хотя авторы тестов по информатике из МПГУ в последние годы и подтянулись.

Объективная причина нынешних проблем с информатикой понятна и вызвана, очевидно, тем, что низкая компьютерная вооруженность наших школ, а также отсутствие общепринятой и общедоступной для всех методики и программы заставляет авторов тестов составлять псевдотесты, искусственно поднимающие балл школьной оценки и не дающие никакой реальной картины уровня обучения предмету.

Автоматы и жизнь

А.Н. Колмогоров

Мы продолжаем публиковать материалы из сборника "Очерки истории информатики в России" (см. также № 19, 29/99). Доклад А.Н. Колмогорова "Автоматы и жизнь" оказал значительное влияние на многие поколения математиков и информатиков нашей страны. Надеемся, он и теперь будет интересен нашим читателям.

Мой доклад "Автоматы и жизнь", подготовленный для семинара научных работников и аспирантов механико-математического факультета Московского государственного университета, вызвал интерес у самых широких кругов слушателей. Популярное изложение доклада подготовлено моей сотрудницей по лаборатории вероятностных и статистических методов МГУ Н.Г. Рычковой. Изложение это во всех существенных чертах правильно, хотя иногда словесное оформление мысли, а следовательно, и некоторые ее оттенки принадлежат Н.Г. Рычковой.

Подчеркну основные идеи доклада, имеющие наиболее широкий интерес.

I. Определение жизни как "особой формы существования белковых тел" (Энгельс) было прогрессивно и правильно, пока мы имели дело только с конкретными формами жизни, развившимися на Земле. В век космонавтики возникает реальная возможность встречи с "формами движения материи" (см. статью "Жизнь" в Большой советской энциклопедии), обладающими основными практически важными для нас свойствами живых и даже мыслящих существ, устроенных иначе. Поэтому приобретает вполне реальное значение задача более общего определения понятия жизни.

II. Современная электронная техника открывает весьма широкие возможности моделирования жизни и мышления. Дискретный (арифметический) характер современных вычислительных машин и автоматов не создает в этом отношении существенных ограничений. Системы из очень большого числа элементов, каждый из которых действует чисто "арифметически", могут приобретать качественно новые свойства.

III. Если свойство той или иной материальной системы "быть живой" или обладать способностью "мыслить" будет определено чисто функциональным образом (например, любая материальная система, с которой можно разумно обсуждать проблемы современной науки или литературы, будет признаваться мыслящей), то придется признать в принципе вполне осуществимым искусственное создание живых и мыслящих существ.

IV. При этом, однако, следует помнить, что реальные успехи кибернетики и автоматики на этом пути значительно более скромны, чем иногда изображается в популярных книгах и статьях. Например, при описании "самообучающихся" автоматов или автоматов, способных "сочинять" музыку или писать стихи, иногда исходят из крайне упрощенного представления о действительном характере высшей нервной деятельности человека и, в частности, творческой деятельности.

V. Реальное продвижение в направлении понимания механизма высшей нервной деятельности, включая и высшие проявления человеческого творчества, естественно, не может ничего убавить в ценности и красоте творческих достижений человека. Я думаю, что это и хотела сказать редакция журнала "Техника — молодежи", сделав лозунг "Материализм — это прекрасно!" одним из подзаголовков в изложении моего доклада.

25 августа 1961 г.

* * *

Я принадлежу к тем крайне отчаянным кибернетикам, которые не видят никаких принципиальных ограничений в кибернетическом подходе к проблеме жизни и полагают, что можно анализировать жизнь во всей ее полноте, в том числе и человеческое сознание со всей его сложностью, методами кибернетики.

Очень часто задают такие вопросы:

— Могут ли машины воспроизводить себе подобных и может ли в процессе самовоспроизведения происходить прогрессивная эволюция, приводящая к созданию машин, существенно более совершенных, чем исходные?

— Могут ли машины испытывать эмоции: радоваться, грустить, быть недовольными чем-нибудь, чего-нибудь хотеть?

— Могут ли, наконец, машины сами ставить перед собой задачи, не поставленные перед ними их конструкторами?

Иногда пытаются отделаться от этих вопросов или обосновать отрицательные ответы на них, предлагая, например, определить понятие "машина" как нечто, каждый раз искусственно создаваемое человеком. При таком определении часть вопросов, скажем первый, автоматически отпадает. Но вряд ли можно считать разумным упорное нежелание разобраться в вопросах, действительно интересных и сложных, прикрываясь насильственно ограниченным пониманием терминов.

Вопрос о том, можно ли на пути кибернетического подхода к анализу жизненных явлений создать подлинную, настоящую жизнь, которая будет самостоятельно продолжаться и развиваться, остается насущной проблемой современности. Уже сейчас он актуален, годен для серьезного обсуждения, ибо изучение аналогий между искусственными автоматами и настоящей живой системой уже сейчас служит принципом исследования самих явлений жизни, с одной стороны, и способом, помогающим изыскивать пути создания новых автоматов — с другой.

Есть и другой способ сразу ответить на все эти вопросы. Он заключается в ссылке на математическую теорию алгоритмов. Математикам хорошо известно, что в пределах каждой формальной системы, достаточно богатой математически, можно сформулировать вопросы, которые кажутся содержательными, осмысленными и должны предполагать наличие определенного ответа, хотя в пределах данной системы такого ответа найти нельзя. Вот поэтому-то и провозглашается, что развитие самой формальной системы есть задача машины, а обдумывание правильного ответа на вопрос — это уже дело человека, преимущественное свойство человеческого мышления.

Такая аргументация, однако, использует идеализированное толкование понятия "мышление", с помощью которого можно легко доказать, что не только машина, но и сам человек мыслить не могут. Здесь предполагается, что человек может давать правильные ответы на любые вопросы, в том числе и на поставленные неформально, а мозг человека способен производить неограниченно сложные формальные выкладки. Между тем нет никаких оснований представлять себе человека столь идеализированным образом — как бесконечной сложности организм, в котором умещается бесконечное количество истин. Чтобы достичь такого положения, заметим в шутку, пришлось бы расселить человечество по звездным мирам, чтобы, пользуясь бесконечностью мира, организовать формальные логические выкладки в бесконечном пространстве и даже передавать их по наследству. Тогда можно было бы считать, что любой математический алгоритм человечество может развить до бесконечности.

Но вряд ли эта аргументация имеет отношение к реальному вопросу. И уж во всяком случае это не возражение против постановки вопроса о том, возможно ли создание искусственных живых существ, способных к размножению и прогрессивной эволюции, в высших формах обладающих эмоцией, волей и мышлением.

Этот же вопрос поставлен изящно, но формально математиком Тьюрингом в его книге "Может ли машина мыслить?". Можно ли построить машину, которую нельзя было бы отличить от человека? Такая постановка как будто ничуть не хуже нашей и к тому же проще и короче. На самом же деле она не вполне отражает суть дела. Ведь, по существу, интересен не вопрос о том, можно ли создать автоматы, воспроизводящие известные нам свойства человека; хочется знать, можно ли создать новую жизнь, столь же высокоорганизованную, хотя, может быть, очень своеобразную и совсем не похожую на нашу. В современной научной фантастике сейчас появляются произведения, затрагивающие эти темы. Интересен и остроумен рассказ "Друг" в сборнике Станислава Лема "Вторжение с Альдебарана" о машине, пожелавшей управлять человечеством. Однако фантазия романистов не отличается особой изобретательностью. И.А. Ефремов, например, выдвигает концепцию, что "все совершенное похоже друг на друга". Стало быть, у высокоорганизованного существа должны быть, по его мнению, два глаза и нос, хотя, может быть, и несколько измененной формы. В век космонавтики не праздно предположение, что нам, возможно, придется столкнуться с другими живыми существами, весьма высокоорганизованными и в то же время совершенно на нас не похожими. Сможем ли мы установить, каков внутренний мир этих существ, способны ли они к мышлению, присущи ли им эстетические переживания, идеалы красоты или чужды и т.п. Почему бы, например, высокоорганизованному существу не иметь вид тонкой пленки — плесени, распластанной на камнях?

1. Что такое жизнь?

Возможно ли искусственное разумное существо?

Поставленный нами вопрос тесно связан с другими: а что такое жизнь, что такое мышление, что такое эмоциональная жизнь, эстетические переживания? В чем, скажем, состоит отличие последних от простых элементарных удовольствий — от пирога, например, или еще чего-нибудь в этом роде? Если говорить в более серьезном тоне, то можно сказать следующее: точное определение таких понятий, как *воля, мышление, эмоции,* еще не удалось сформулировать. Но на естественно-научном уровне строгости такое определение возможно. Если мы не признаем эту возможность, мы окажемся безоружными против аргументов солипсизма.

Хотелось бы научиться на основании фактов поведения, например, делать выводы о внутреннем состоянии живого высокоорганизованного существа.

Как изучать высшую нервную деятельность, используя кибернетический подход? Здесь открываются следующие пути: во-первых, можно детально изучать само

поведение животных или человека; во-вторых, изучать устройство их мозга; можно, наконец, иногда довольствоваться и так называемым симпатическим пониманием. Если, скажем, просто внимательно наблюдать кошку или собаку, то, и не зная науки о поведении и условных рефлексах, можно прекрасно понять, что они думают и чего хотят. Несколько труднее достигнуть такого понимания с птицами или, например, с рыбами, но вряд ли и это невозможно. Это вопрос не новый, частично он уже решен, частично легко решаем, частично — трудно. Опыт индуктивного развития науки говорит нам, что все вопросы, долго не находившие решения, постепенно разрешаются, и вряд ли нужно думать, что именно здесь существуют заранее установленные пределы, дальше которых продвинуться нельзя.

Если считать, что анализ любой высокоорганизованной системы естественно входит в состав кибернетики, придется отказаться от распространенного мнения, что основы кибернетики включают в себя лишь изучение систем, имеющих заранее назначенные цели. Часто кибернетику определяют как науку, занимающуюся изучением управляющих систем. Считается, что все такие системы обладают общими свойствами и свойство номер один у них — наличие цели. Это верно лишь до тех пор, пока все, что мы выделяем в качестве организованных систем, управляющих собственной деятельностью, похоже на нас самих. Однако если мы хотим методами кибернетики изучать происхождение таких систем, их естественную эволюцию, то такое определение становится узким. Вряд ли кибернетика поручит какой-либо другой науке выяснять, каким образом обычная причинная связь в сложных системах путем естественного развития приводит к возможности рассматривать всю систему как действующую целесообразно.

Обычно понятие "действовать целесообразно" включает умение охранять себя от разрушающих внешних воздействий или, скажем, способность содействовать своему размножению. Спрашивается: кристаллы действуют целесообразно или нет? Если "зародыш" кристалла поместить в некристаллическую среду, будет ли он развиваться? Ведь никаких отдельных органов у кристалла различить невозможно, стало быть, это есть некая промежуточная форма. И существование таковых неизбежно.

По-видимому, частные задачи, подобные этой, будут решать науки, непосредственно с ними связанные. Опытом частных наук никак нельзя пренебрегать. Но исключить из содержания кибернетики общие представления о причинных связях в целесообразно действующих системах, ставящих себе цели, так же нельзя, как нельзя, например, уже при имитации жизни автоматами не считаться, скажем, с тем, что и сами эти цели меняются в процессе эволюции, а вместе с этим изменяется и представление о них.

Когда говорят, что организация механизма наследственности, позволяющего живым организмам передавать свое целесообразное устройство потомкам, имеет целью воссоздать данный вид, придать ему определенные свойства, а также возможности изменчивости, прогрессивной эволюции, то кто же ставит эту цель? Или если рассматривать систему в целом, то кто же, как не она сама, ставит перед собой цель развития путем отсеивания негодных экземпляров и размножения совершенных?

Подводя итоги, можно сказать, что изучение в общей форме возникновения систем, в которых применимо понятие целесообразности, есть одна из главных задач кибернетики. При этом изучение в общей форме естественно предполагает знание, отвлеченное от деталей физического осуществления, от энергетики, химии, возможностей техники и т.п. Нас здесь интересует только, как возникает возможность сохранять и накапливать информацию.

Такая широкая постановка задачи содержит в себе много трудностей, но отказаться от нее на современном этапе развития науки уже невозможно.

Если признавать важность задачи определения в объективных обобщенных терминах существенных свойств внутренней жизни (высшей нервной деятельности) какой-то незнакомой нам и непохожей на нас высокоорганизованной системы, то нельзя ли тот же путь предложить и в применении к нашей системе — человеческому обществу? Хотелось бы на общем языке, одном и том же для всех высокоорганизованных систем, уметь описывать и все явления жизни человеческого общества. Представим себе воображаемого постороннего наблюдателя нашей жизни, который совершенно не обладает ни симпатиями к нам, ни умением понять, что мы думаем и переживаем. Он просто наблюдает большое скопление организованных существ и желает понять, как оно устроено. Совершенно так же, как, скажем, мы наблюдаем муравейник. Через некоторое время он, пожалуй, без особого труда сможет понять, какую роль играет информация, содержащаяся, например, в железнодорожных справочниках (человек теряет такой справочник и не может попасть на нужный поезд). Правда, наблюдателю пришлось бы столкнуться с большими трудностями. Как, например, понять ему следующую картину: множество людей приходит вечером в большое помещение, несколько человек поднимаются на возвышение и начинают делать беспорядочные движения, а остальные сидят при этом спокойно; по окончании люди расходятся без всякого обсуждения. Один из молодых математиков, может быть в шутку, приводит и другой пример необъяснимого поведения: люди заходят в помещение, там получают бутылки с некоей жидкостью, после чего начинают бессмысленно жестикулировать. Постороннему наблюдателю будет трудно установить, что же это такое — просто разлад в машине, какая-то

пауза в ее непрерывной осмысленной работе, или же можно описать, что происходит в этих двух случаях, и установить разницу между ними.

Оставив шутливый тон, сформулируем серьезно возникающую здесь проблему: нужно научиться в терминах поведения осуществлять объективное описание самого механизма, это поведение обусловливающего, уметь различать отдельные виды деятельности высокоорганизованной системы. Впервые в нашей стране И.П. Павлов установил возможность объективного изучения поведения животных и человека, а также регулирующих это поведение мозговых процессов без всяких субъективных гипотез, выраженных в психологических терминах. Глубокое изучение предложенной проблемы есть не что иное, как павловская программа анализа высшей нервной деятельности в ее дальнейшем развитии.

Создание высокоорганизованных живых существ превосходит возможности техники наших дней. Но всякие ограничительные тенденции, всякое неверие или даже утверждение невозможности на рациональных путях достичь объективного описания человеческого сознания во всей его полноте сейчас явились бы тормозом в развитии науки. Разрешение этой проблемы необходимо, ибо уже истолкование разных видов деятельности может служить толчком для развития машинной техники и автоматики. С другой стороны, возможности объективного анализа нервной системы сейчас столь велики, что не хочется заранее останавливаться перед задачами любой трудности.

Если технические трудности будут преодолены, то вопрос о практической целесообразности осуществления соответствующей программы работ останется по меньшей мере спорным.

Однако в рамках материалистического мировоззрения не существует никаких состоятельных принципиальных аргументов против положительного ответа на наш вопрос. Более того, этот положительный ответ является сейчас современной формой убеждений о естественном возникновении жизни и материальной основе сознания.

2. Дискретна или непрерывна мысль?

В кибернетике и теории автоматов сейчас наиболее разработана теория работы дискретных устройств, т.е. таких устройств, которые состоят из большого числа отдельных элементов и работают отдельными тактами. Каждый элемент может находиться в небольшом числе состояний, и изменение состояния отдельного элемента зависит от предыдущих состояний сравнительно небольшого числа элементов. Так устроены электронные машины, так, предположительно, устроен и человеческий мозг. Считается, что мозг имеет таких отдельных элементов — нервных клеток — 10^{10}, а может быть, и еще больше. Несколько проще, но еще более грандиозно в смысле объема устроен аппарат наследственности.

Иногда делают вывод, что кибернетика должна заниматься лишь дискретными устройствами. Против такого подхода есть два возражения. Во-первых, реальные сложные системы — как многие машины, так и все живые существа — действительно имеют определенные устройства, основанные на принципе непрерывного действия. Что касается машин, то таким примером может служить, скажем, руль автомобиля и т.п. Если мы обратимся к человеческой деятельности — сознательной, но не подчиненной законам формальной логики, т.е. деятельности интуитивной или полуинтуитивной, например, к двигательным реакциям, — то мы обнаружим, что большое совершенство и отточенность механизма непрерывного движения построены на движениях непрерывно-геометрического характера. Если человек совершает тройной прыжок или прыжок с шестом или, например, готовится к дистанции слалома, его движение должно быть заранее намечено как непрерывное (для математиков: путь слаломиста оказывается даже аналитической кривой). Можно полагать, однако, что это не есть радикальное возражение против дискретных механизмов. Скорее всего интуиция непрерывной линии в мозге осуществляется на базе дискретного механизма.

Второе возражение против дискретного подхода заключается в следующем: заведомо человеческий мозг и даже, к сожалению, часто вычислительные машины отнюдь не всегда действуют детерминированно — полностью закономерным образом. Результат их действия в некоторый момент (в данной ячейке) нередко зависит от случая. Желая обойти эти возражения, можно сказать, что и в автоматы можно "ввести случайность". Вряд ли имитирование случайности (т.е. замена случая какими-то закономерностями, не имеющими отношения к делу) может принести сколько-нибудь серьезный вред при моделировании жизни. Правда, вмешательство случайности часто рассматривается несколько примитивно: заготавливается достаточно длинная лента случайных чисел, которая затем используется для имитаций случая в различных задачах. Но при частом употреблении эта заготовленная "случайность" в конце концов перестает быть случайностью. Исходя из этих соображений, к вопросу имитации случая на автоматах следует подходить с большой осторожностью. Однако принципиально это вещь, во всяком случае, возможная.

Только что изложенная аргументация приводит нас к следующему основному выводу.

Несомненно, что переработка информации и процессы управления в живых организмах построены на сложном переплетении дискретных (цифровых) и непрерывных механизмов, с одной стороны, детерминированного и вероятностного принципов действия — с другой.

Однако дискретные механизмы являются ведущими в процессах переработки информации и управления в живых организмах. Не существует состоятельных аргументов в пользу принципиальной ограниченности возможностей дискретных механизмов по сравнению с непрерывными.

3. Что такое "очень много"?

Часто, сомневаясь в возможности моделировать человеческое сознание на автоматах, говорят, что количество функций высшей нервной деятельности человека необъятно велико и никакая машина не может стать моделью сознательной человеческой деятельности в полном ее объеме. Одних только нервных клеток в коре головного мозга 10^{10}. Каково же должно быть число элементов в машине, имитирующей всю сложную высшую нервную деятельность человека?

Эта деятельность, однако, связана не с разрозненными нервными клетками, а с довольно большими агрегатами их. Невозможно представить себе, чтобы, скажем, какая-нибудь математическая теорема "сидела" в одной-единственной, специально для нее заготовленной нервной клетке или даже в каком-то определенном числе их. По-видимому, дело обстоит совершенно иначе. Наше сознание оперирует небольшими количествами информации. Количество единиц информации, которое человек воспринимает и перерабатывает в секунду, совсем невелико. Вот один несколько парадоксальный пример: слаломист, преодолевая дистанцию, в течение десяти секунд воспринимает и перерабатывает значительно большую информацию, чем при других, казалось бы, более интеллектуальных видах деятельности, во всяком случае, больше, чем математик пропускает через свою голову за сорок секунд напряженной работы мысли. Вообще вся сознательная жизнь человека устроена как-то очень своеобразно и сложно, но, когда закономерности ее будут изучены, для моделирования ее потребуется гораздо меньше элементарных ячеек, чем для моделирования всего мозга, как это ни удивительно.

Какие же объемы информации могут создавать уже качественное своеобразие сложных явлений, подобных жизни, сознанию и т.п.?

Можно разделить все числа на малые, средние, большие и сверхбольшие. Эта классификация нестрога, в рамках ее нельзя будет сказать, что такое-то число, например, среднее, а следующее за ним — уже большое. Здесь числа делятся на категории с точностью до порядка величин. Но большая строгость нам здесь и не нужна. Каковы же эти категории? Начнем с определений, понятных лишь математикам.

I. *Число* А *назовем малым, если практически возможно перебрать все схемы из* А *элементов с двумя входами и выходами или выписать для них все функции алгебры логики с* А *аргументами.*

II. *Число* Б *называется средним, если мы оказываемся не в состоянии перебрать практически все схемы из* Б *элементов, а можем перебрать лишь сами эти элементы или (что чуть-чуть сложнее) выработать систему обозначений для любой системы из* Б *элементов.*

III. *И, наконец, число* В *— большое, если мы не в состоянии практически перебрать такое число элементов, а можем лишь установить систему обозначений для этих элементов.*

IV. *Числа будут сверхбольшими, если практически и этого нельзя сделать; они нам, как мы увидим дальше, и не понадобятся.*

Поясним теперь эти определения на доступных примерах.

Пусть к одной электрической лампочке подсоединены три выключателя, каждый из которых может находиться в левом (Л) или правом (П) положении. Тогда, очевидно, возможных совместных положений трех выключателей будет $2^3 = 8$. Перечислим их для наглядности:

1)ЛЛЛ 3)ЛПП 5)ПЛЛ 7)ПЛП
2)ЛПЛ 4)ЛЛП 6)ППЛ 8)ППП.

Проводку к нашим выключателям можно сделать таким образом, что в каждом из выписанных положений лампочка может как гореть, так и не гореть. Если произвести подсчет, то окажется, что различных положений выключателей, сопровожденных такими отметками, будет 2^{2^3}, т.е. $2^8 = 256$. Справедливость этого последнего утверждения читатель без труда может проверить самостоятельно, дополняя выписанные положения выключателей знаками "горит", "не горит".

Тот факт, что такое упражнение под силу читателю и не займет у него слишком много времени, и убеждает нас в том, что число 3 (число выключателей) относится к малым. Если бы выключателей было не 3, а, скажем, 5, то пришлось бы выписать $2^{2^5} = 4\,294\,967\,296$ различных совместных положений выключателей, сопровожденных отметками "горит", "не горит". Вряд ли можно за какое-нибудь разумное время практически проделать все это, не сбившись. Поэтому число 5 уже нельзя считать малым.

Чтобы стал понятен термин "среднее число", приведем другой пример. Представьте себе, что вас ввели в помещение, где находится 1000 человек, и предложили с каждым из них поздороваться за руку. Правда, ваша рука после таких упражнений будет чувствовать себя неважно, но практически (по времени) проделать такое упражнение вполне возможно. Вы вполне сумеете, не сбившись, подойти к каждому из тысячи и протянуть ему руку. А если бы последовало предложение всей тысяче присутствующих обменяться друг с другом рукопожатиями, да еще каждой компании из трех человек внутри своего кружка дополнительно обменяться рукопожатия-

ми и т.д., то это оказалось бы немыслимым. Число 1000 и есть среднее. Можно сказать, что мы "перебрали" тысячу элементов, отметив при этом каждого (рукопожатием).

Совсем простым примером большого числа является число видимых звезд на небосклоне. Каждый знает, что невозможно пересчитать звезды пальцем, а тем не менее существует каталог звездного неба (т.е. выработана система обозначений), пользуясь которым, мы в любой момент можем получить справку о нужной нам звезде.

Естественно, что вычислительная машина может, во-первых, дольше работать не сбиваясь, а во-вторых, она составляет различные схемы во много раз быстрее, чем человек. Поэтому в каждой категории соответствующие числа для машины будут больше, чем для человека.

Числа	Человек	Машина
Малые	3	10
Средние	1000	10^{10}
Большие	10^{100}	$10^{10^{10}}$

Что поучительного в этой таблице? Из нее видно, что хотя соответственные числа для машины гораздо больше, чем для человека, но остаются близкого порядка с ними. Между же числами разных категорий существует непроходимая грань: числа, средние для человека, не становятся малыми для машины, так же как числа, большие для человека, не становятся средними для машины. 10^3 несравненно больше, чем 10, а 10^{100} безнадежно больше, чем 10^{10}. Заметим, что объем памяти живого существа и даже машины характеризуется средними числами, а многие проблемы, решающиеся путем так называемого простого перебора,— большими.

Здесь мы сразу выходим за пределы возможностей сравнения путем простого перебора. Проблемы, которые не могут быть решены без большого перебора, останутся за пределами возможностей машины на сколь угодно высокой ступени развития техники и культуры.

К этому выводу мы пришли, не обращаясь к понятию бесконечности. Оно нам не понадобилось и вряд ли понадобится при решении реальных проблем, возникающих на пути кибернетического анализа жизни.

Зато важным становится другой вопрос: существуют ли проблемы, которые ставятся и решаются без необходимости большого перебора? Такие проблемы должны прежде всего интересовать кибернетиков, ибо они реально разрешимы.

Принципиальная возможность создания полноценных живых существ, построенных полностью на дискретных (цифровых) механизмах переработки информации и управления, не противоречит принципам материалистической диалектики. Противоположное мнение может возникнуть лишь потому, что некоторые привыкли видеть диалектику лишь там, где появляется бесконечность. При анализе явлений жизни существенна, однако, не диалектика бесконечного, а диалектика большого числа.

4. Осторожно, увлекаемся!

В настоящее время для кибернетики, пожалуй, больше, чем для всякой другой науки, важно, что о ней пишут. Я не принадлежу к большим энтузиастам всей той литературы по кибернетике, которая сейчас так широко издается, и вижу в ней большое количество, с одной стороны, преувеличений, а с другой — упрощенчества.

Нельзя, конечно, сказать, что в этой литературе утверждается то, что на самом деле недостижимо, но в ней часто встречаются восторженные статьи, сами заглавия которых уже кричат об успехах в моделировании различных сложных видов человеческой деятельности, которые в действительности моделируются пока совсем плохо. Например, в американской кибернетической литературе и у нас порой даже в совсем серьезных научных журналах можно встретить работы о так называемом машинном сочинении музыки (это не относится к работам Р.Х. Зарипова). Под этим обычно подразумевается следующее: в память машины "закладывается" нотная запись большого числа (скажем, 70) ковбойских песен или, например, церковных гимнов; затем машина по первым четырем нотам одной из этих песен отыскивает все те песни, где эти четыре ноты встречаются в том же порядке, и, случайным образом выбрав одну из них, берет из нее следующую, пятую ноту. Теперь перед машиной вновь четыре ноты (2, 3, 4 и 5), и она снова таким же способом осуществляет поиски и выбор. Так машина как бы на ощупь "создает" некую новую мелодию. При этом утверждается, что если в памяти машины были ковбойские песни, то и в ее творении слышится нечто "ковбойское", а если это были церковные гимны — то нечто "божественное". Спрашивается: а что произойдет, если машина будет производить поиск не по четырем, а по семи идущим подряд нотам? Поскольку в действительности двух произведений, содержащих семь одинаковых нот подряд, почти не встретишь, то, очевидно, "запев" семь нот из какой-нибудь песни, машина вынуждена будет пропеть ее до конца. Если же, наоборот, машине для собственного творчества достаточно знать только две ноты (а произведений с двумя одинаковыми нотами сколько угодно), то здесь ей представился бы такой широкий выбор, что вместо мелодии из машины послышалась бы какофония звуков.

Вся эта несложная схема преподносится в литературе как "машинное сочинение музыки", причем всерьез заявляется, что с увеличением числа нот, нужных "для затравки", машина начинает создавать музыку более серьезного, классического характера, а с уменьшением этого числа переходит на современную, джазовую.

На сегодня мы еще очень далеки от осуществления анализа и описания высших форм человеческой деятельности, мы даже еще не научились в объективных терминах давать определения многих встречающихся здесь категорий и понятий, а не только моделировать такие сложные виды этой деятельности, к каким относится создание музыки. Если мы не умеем понять, чем отличаются живые существа, нуждающиеся в музыке, от существ, в ней не нуждающихся, то, приступая сразу к машинному сочинению музыки, мы окажемся в состоянии моделировать лишь чисто внешние факторы.

"Машинное сочинение музыки" — это только пример упрощенного подхода к проблемам кибернетики. Другой распространенный недостаток заключается в том, что сторонники кибернетики настолько увлеклись возможностями кибернетического подхода к решению любых самых сложных задач, что позволяют себе пренебрегать опытом, накопленным другими науками за долгие века их существования. Часто забывают о том, что анализ высших форм человеческой деятельности был начат давно и продвинулся довольно далеко. И хотя он и ведется в других, не кибернетических, терминах, но по существу объективен, и его необходимо изучать и использовать. А то, что сумели сделать кибернетики "голыми руками" и вокруг чего поднимают такую шумиху, зачастую не выходит за рамки исследования самых примитивных явлений. Однажды на вечере в московском Доме литераторов один из участников вел с трибуны разговор о том, что наше время должно было создать и уже создало новую медицину. Эта новая медицина есть достояние и предмет изучения не медиков, а специалистов по теории автоматического регулирования! Самое главное в медицине, по мнению выступавшего, — это циклические процессы, происходящие в человеческом организме. А такие процессы как раз и описываются дифференциальными уравнениями, изучаемыми в теории автоматического регулирования. Так что изучать медицину в медицинских институтах теперь вроде как устарело — ее надо передать в ведение втузов и математических факультетов. Может быть, и верно, что специалисты по теории автоматического регулирования могут сказать свое слово в разрешении отдельных проблем, стоящих перед медициной. Но если они захотят принять участие в этой работе, то прежде всего им потребуется колоссальная доквалификация, ибо опыт, накопленный медициной, этой старейшей из наук, огромен и, для того чтобы сделать в ней что-то серьезное, надо сначала овладеть им.

5. Почему только крайности?

Вообще анализ высшей нервной деятельности в кибернетике сосредоточен пока на двух крайних полюсах. С одной стороны, кибернетики активно занимаются изучением условных рефлексов, т.е. простейшего типа высшей нервной деятельности. Всем, вероятно, известно, что такое условный рефлекс. Если два каких-нибудь раздражителя многократно осуществляются одновременно друг с другом (например, одновременно с подачей пищи включается звонок), то через некоторое время уже один из этих раздражителей (звонок) вызывает ответную реакцию организма (слюноотделение) на другой раздражитель (подачу пищи). Это сцепление является временным и, если его не подкреплять, постепенно исчезает. Значительная часть кибернетических проблем, которые известны сейчас под названием математической теории обучения, охватывает такие очень простые схемы, которые не исчерпывают и малой доли всей сложной высшей нервной деятельности человека и в анализе самой условно-рефлекторной деятельности представляют собой лишь начальную ее ступень.

Другой полюс — это теория формально-логических решений. Эта сторона высшей нервной деятельности человека хорошо поддается изучению математическими методами, и с созданием вычислительной техники и вычислительной математики исследования такого рода быстро двинулись вперед. И здесь кибернетики во многом преуспели.

А все огромное пространство между этими двумя полюсами — самыми примитивными и самыми сложными психическими актами (даже такие простые формы синтетической деятельности, как, скажем, механизм точно рассчитанного геометрического движения, о котором говорилось выше, пока плохо поддаются кибернетическому анализу) — изучается крайне мало, чтобы не сказать, вовсе не изучается.

6. Кибернетика и язык

Особое положение сейчас занимает математическая лингвистика. Эта наука только еще создается и развивается по мере накопления кибернетических проблем, связанных с языком. Она имеет дело с анализом высших форм человеческой деятельности скорее интуитивного, нежели формально-логического характера, ибо эта деятельность плохо поддается точному описанию. Каждый знает, что такое грамотно построенная фраза, правильное согласование слов и т.п., но никто пока не может адекватно передать это знание машине. Точный, логически и грамматически безукоризненный машинный перевод сейчас возможен был бы, пожалуй, только с латинского и на латинский язык, грамматические правила которого достаточно полны и однозначны. Грамматические же правила новых, живых языков, по-видимому, еще недостаточны для осуществления с их помощью машинного перевода. Необходимым здесь анализом занимаются уже давно, и в настоящее время машинный перевод стал предметом широко и серьезно поставленной деятельности. Можно, пожалуй, сказать, что именно на нем сосредоточено сейчас основное внимание математических лингвистов. Однако в теоретических работах по математической лингвистике мало учитывается одно обстоятельство, а

именно тот факт, что язык возник значительно раньше формально-логического мышления. Быть может, для теоретической науки одно из самых интересных исследований (в котором могут естественно сочетаться идеи кибернетики, новый математический аппарат и современная логика) есть исследование процесса образования слов как второй сигнальной системы. Первоначально, при полном еще отсутствии понятий, слова выступают в роли сигналов, вызывающих определенную реализацию. Возникновение логики обычно относят к сравнительно недавнему времени: по-видимому, только в Древней Греции было ясно понято и сформулировано, что слова не просто являются обозначениями неких непосредственных представлений и образов, но что от слова можно отделить понятие. До настоящего, формально-логического, мышления мысли возникали не формализованные в понятии, а как комбинирование слов, которые ведут за собой другие слова, как попытки непосредственно зафиксировать проходящий перед нашим сознанием поток образов и т.д. Проследить этот механизм выкристаллизовывания слов как сигналов, несущих в себе комплекс образов, и создания на этой базе ранней логики — крайне благодарная область исследования, для математика в частности, что, впрочем, неоднократно отмечалось в кибернетической литературе.

Интересным может показаться и следующий вопрос: как формулируется логическая мысль у человека? Попробуем проследить этапы этого процесса на примере работы математика над какой-нибудь проблемой. Сначала, по-видимому, возникает желание исследовать тот или иной вопрос, затем какое-то приблизительное, неведомо откуда возникшее представление о том, что мы надеемся получить в результате наших поисков и какими путями нам, может быть, удастся этого достичь, и уже на следующем этапе мы пускаем в ход свой внутренний "арифмометр" формально-логического рассуждения. Таков, по-видимому, путь формирования логической мысли, такова схема процесса творчества. Может, вероятно, представиться интересным не только исследовать первую, интуитивную стадию этого процесса, но и задаться целью создать машину, способную помочь человеку в процессе творчества на стадии оформления мысли (математику, например, на стадии оформления вычислений), поручить, скажем, такой машине понимать и фиксировать в полном виде какие-то неясные, вспомогательные наброски чертежей и формул, которые всякий математик рисует на бумаге в процессе творческих поисков, или, например, воссоздавать по наброскам изображения фигур в многомерных пространствах и т.п. Иными словами, интересно подумать о создании машин, которые, не подменяя человека, уже сейчас помогали бы ему в сложных процессах творчества. Пока еще трудно даже представить себе, каким образом и на каких путях такую машину можно было бы осуществить. Но хотя пока еще эта задача и далека от своего разрешения, разговор обо всех таких вопросах уже возник в кибернетической литературе, что, по-видимому, можно только приветствовать.

Как можно уже увидеть из нескольких приведенных здесь примеров, различных проблем, связанных с пониманием объективного устройства самых тонких разделов высшей нервной деятельности человека, очень много. И все они заслуживают должного внимания кибернетиков.

7. Материализм — это прекрасно!

В заключение следует остановиться на вопросах, касающихся, если можно так сказать, этической стороны идей кибернетики. Встречающиеся часто отрицание и неприятие этих идей проистекают из нежелания признать, что человек является действительно сложной материальной системой, но системой конечной сложности и весьма ограниченного совершенства и поэтому доступной имитации. Это обстоятельство многим кажется унизительным и страшным. Даже воспринимая эту идею, люди не хотят мириться с ней: такая картина всеобъемлющего проникновения в тайны человека, вплоть до возможности, так сказать, "закодировать" его и "передать по телеграфу" в другое место, кажется им отталкивающей и пугающей. Встречаются опасения и другого рода: а допускает ли вообще наше внутреннее устройство исчерпывающее объективное описание? Предлагалось, например, поставить перед кибернетикой задачу научиться отличать по объективным признакам существа, нуждающиеся в сюжетной музыке, от существ, в ней не нуждающихся. А вдруг поанализируем, поанализируем — и окажется, что и в самом деле нет никакого разумного основания выделять такую музыку как благородную по сравнению с другими созвучиями.

Мне представляется важным понимание того, что ничего унизительного и страшного нет в этом стремлении постичь себя до конца. Такие настроения могут возникать лишь из полузнания: реальное понимание всей грандиозности наших возможностей, ощущение присутствия вековой человеческой культуры, которая придет нам на помощь, должно производить огромное впечатление, должно вызывать восхищение! Все наше устройство в самих себе понятно, но понятно и то, что это устройство содержит в себе колоссальные, ничем не ограниченные возможности.

На самом деле нужно стремиться этот глупый и бессмысленный страх перед имитирующими нас автоматами заменить огромным удовлетворением тем фактом, что такие сложные и прекрасные вещи могут быть созданы человеком, который еще совсем недавно находил простую арифметику чем-то непонятным и возвышенным.

Лучший образовательный сервер!

Мы очень редко пишем о собственных успехах. Но в этот раз просто не можем удержаться! По результатам анкетирования участников выставки "Информационные технологии в образовании", прошедшей в Москве 9—12 ноября, сервер Объединения педагогических изданий "Первое сентября" **www.1september.ru** признан лучшим в номинации "Образовательные серверы".

Сервер "Первого сентября", который начинался в 1997 году с маленькой странички "Информатики", сегодня превращается в мощный информационный центр, в котором может найти много интересного преподаватель практически любого предмета. Само за себя говорит и постоянно растущее количество посетителей: сегодня в "лучшие дни" на сервере бывает более тысячи человек.

Мы прекрасно понимаем, что еще очень не скоро придет время, когда Интернет станет повседневным рабочим инструментом для каждого учителя, но твердо уверены, что это обязательно произойдет.

Гл. редактор "Информатики"
и сервера **www.1september.ru**
С.Л. ОСТРОВСКИЙ


ИТО'99

Институт ЮНЕСКО по информационным технологиям в образовании
Министерство образования России
Институт проблем информатики Российской академии наук
Московский комитет образования
Научно-производственное предприятие «БИТ про»

Диплом

Награждается
газета «Первое сентября»
за лучший образовательный сайт
www.1september.ru
по мнению посетителей IX выставки
«Информационные технологии в образовании»

Сопредседатели оргкомитета — Л.Г. Гудков, М.А. Зайцева, И.А. Смольянинова
Председатель программного комитета конференции — С.А. Христочевский

Москва
9 – 12 ноября 1999 года

"Информатика" в Нижнем Тагиле

На встрече с читателями в Нижнем Тагиле. Справа — региональный представитель "Информатики" Е.Д. Ширинкин.

Организовав сеть региональных представителей газеты, редакция "Информатики" проводит встречи с читателями в различных городах России. Осенью уже прошли встречи с учителями Челябинска (региональный представитель в Челябинске — А.Л. Королев), Воронежа (О.А. Богданова), Красноярска (А.В. Алексеев), Калуги (Л.М. Матвеева), Кирова (С.М. Окулов). 29 октября состоялась встреча с учителями информатики Нижнего Тагила, организованная региональным представителем газеты Е.Д. Ширинкиным.

Встречи эти приносят огромную пользу. И хотя пожелания, которые высказывают читатели, зачастую диаметрально противоположны, мы прилагаем максимум усилий, чтобы как можно полнее учитывать их при работе над газетой. Нашей с вами профессиональной газетой. Так, по результатам обсуждения в Нижнем Тагиле мы постараемся в ближайшее время усилить информационный блок газеты, станем чаще публиковать обзоры программных средств и репортажи с различных выставок, конференций и встреч. (Первую статью о прошедшей конференции-выставке "ИТО-99" с кратким обзором новинок программного обеспечения, представленных на выставке, вы сможете прочитать в № 46.)

Мы благодарим всех, кто своими советами, замечаниями, пожеланиями, статьями помогает нам делать эту газету. Пишите нам. Мы всегда готовы рассмотреть предложения по организации встреч с учителями информатики в различных регионах.

Первое сентября

Индексы 32024 и 32586

Газета для учителей и родителей. Создатель — Симон Соловейчик. Выходит 2 раза в неделю.

20 ПРЕДМЕТНО-МЕТОДИЧЕСКИХ ПРИЛОЖЕНИЙ

1. **Английский язык** (индексы 32025 и 32587)
2. **Биология** (32026 и 32588)
3. **Воскресная школа** (32742 и 32743)
4. **География** (32027 и 32589)
5. **Дошкольное образование** (33373 и 33374)
6. **Здоровье детей** (32033 и 32590)
7. **Информатика** (32291 и 32591)
8. **Искусство** (32584 и 32585)
9. **История** (32028 и 32592)
10. **Литература** (32029 и 32593)
11. **Математика** (32030 и 32594)
12. **Начальная школа** (32031 и 32598)
13. **Немецкий язык** (32292 и 32599)
14. **Русский язык** (32383 и 32601)
15. **Управление школой** (32652 и 32653)
16. **Физика** (32032 и 32596)
17. **Французский язык** (33371 и 33372)
18. **Химия** (32034 и 32597)
19. **Школьный психолог** (32898 и 32899)
20. **Спорт в школе** (32384 и 32595)

Газеты распространяются только по подписке во всех регионах России и странах СНГ.

Первое сентября
Объединение педагогических изданий
www.1september.ru
Адрес: 121165, Москва, ул. Киевская, д. 24
Телефон/факс: (095) 249-31-38, 249-31-84
Internet: www.1september.ru
E-mail: gazeta@1september.ru

Новые газеты — по старым ценам

Помощь учителю в его каждодневной работе – при подготовке к проведению занятий и непосредственно на уроках – главная задача предметно-методических приложений к газете «Первое сентября». Читатели приложений еженедельно получают практический материал для работы в классе: поурочные планы, опорные конспекты, подборки вопросов и заданий, разработки учебных тем, варианты контрольных работ и многое другое.

На каждое издание можно подписаться отдельно.

Подписаться на газеты можно в любом почтовом отделении по каталогу «Газеты, журналы» агентства «Роспечать» (первое полугодие 2000 года, стр. 14)

НАШИ НОВИНКИ

С 1 января 2000 года

новые приложения

индексы 33373 и 33374 **индексы 33371 и 33372**

новый журнал

«Я иду на урок. Начальная школа. Прописи. Тесты»

Сборник методических пособий для работы с детьми в школе и дома. Индекс подписки — 79384 (каталог агентства «Роспечать», с. 251).

Универсальный язык для бизнеса

40 лет назад (в декабре 1959 года) были завершены работы над первой версией языка Кобол

В конце 1950-х годов, когда язык программирования Фортран приобретал приверженцев в традиционной области применения ЭВМ — науке и технике, в мире бизнеса все сильнее стала ощущаться необходимость автоматизации процессов обработки информации. В больших корпорациях, руководствующихся принципом "время — деньги", стали понимать, какую выгоду можно извлечь, быстро обрабатывая огромные объемы данных с помощью компьютеров [1].

Из языков, которые предполагалось использовать в сфере бизнеса, наибольшую популярность приобрели Флоуматик (FLOW-MATIC), Комтран (COMTRAN, Коммерческий транслятор) фирмы IBM и Аймако (AIMACO), разработанный в ВВС США. Однако ни один из указанных языков не годился для широкого применения, а кроме того, каждый из них мог использоваться лишь на компьютерах одного семейства — недостаток почти всех языков программирования того времени. Не подходил в качестве универсального языка для обработки коммерческой информации и Факт (FACT, от Fully Automatic Compiling Technique — полностью автоматический метод компиляции).

Таким образом, требовался "стандартизованный" машинно-независимый язык для деловых приложений, причем было ясно, что Фортран здесь не подойдет: для бизнеса нужно в первую очередь формировать документы, а не решать уравнения.

В конце мая 1959 года в США состоялась специальная конференция по языкам программирования, на которой был принят ряд решений, связанных с созданием языка для бизнеса. Позже это собрание получило название КОДАСИЛ (CODASYL, от Conference on Data System Languages — конференция по языкам систем обработки данных). В декабре того же года рабочая группа, действовавшая под эгидой исполнительного комитета КОДАСИЛ, завершила подготовку первой версии нового языка программирования, получившего название Кобол (COBOL, от COmmon Business-Oriented Language — универсальный язык, предназначенный для бизнеса).

Новый язык стал быстро развиваться и приобретать популярность. В течение нескольких лет появились еще три его версии, последняя из которых в 1968 году была утверждена в качестве первого американского стандарта языка Кобол [2]. (У нас в стране в 1968 году появились первые компиляторы для Кобола, а отечественный стандарт был введен в 1977 году.) Тем не менее изменение и совершенствование Кобола не прекращалось, причем популярность его росла. К середине 1980-х годов (к 25-летию создания языка) общая стоимость программ, написанных на Коболе, составляла, по оценкам зарубежных специалистов, 50 млрд долларов [1].

Кобол особенно эффективен при описании простых операций (таких, как сложение, вычитание, вычисление процентов), применяемых к большим массивам данных, — подобные расчеты часто выполняются в сфере бизнеса. При этом программа на Коболе напоминает обычный английский текст, что делает ее легко читаемой и облегчает освоение языка (в нашей стране принят русский вариант языка Кобол).

И хотя Кобол не так сильно повлиял на появившиеся позже языки, как, скажем, тот же Фортран, он оставил заметный след в истории развития языков программирования [3].

Литература

1. Язык компьютера: Пер. с англ. М.: Мир, 1989.

2. *Бабенко Л.П.* Кобол // Энциклопедия кибернетики. Т. 1. Киев: Гл. редакция Украинской Советской Энциклопедии, 1975.

3. *Малыхина М.П., Частиков А.П.* Языки программирования: Кобол / Вычислительная техника и ее применение, № 10/88.

Редакция "Информатики" просит откликнуться Мильтову И.В. из г. Усть-Кута Иркутской области. Нам необходимо уточнить Ваш почтовый адрес, чтобы отправить причитающийся Вам приз.

Гл. редактор
С.Л. Островский
Зам. гл. редактора
Е.Б. Докшицкая
Редакция:
И.Н. Фалина,
Н.Л. Беленькая,
Н.П. Медведева
Дизайн и компьютерная верстка:
Н.И. Пронская
Корректоры:
Е.Л. Володина,
С.М. Подберезина

выходит четыре раза в месяц
При перепечатке ссылка на ИНФОРМАТИКУ обязательна, рукописи не возвращаются

121165, Киевская, 24
тел. 249 4896
Отдел рекламы
тел. 249 9870

Учредитель: ООО "Чистые пруды"
Регистрационный номер 012868
Отпечатано в типографии ОАО ПО "Пресса-1".
125865, ГСП, Москва, ул. "Правды", 24.
Тираж 5000 экз.
Заказ №

ИНДЕКС ПОДПИСКИ
для индивидуальных подписчиков 32291
комплекта приложений 32744

Тел. (095)249 3138, 249 3386. Факс (095)249 3184

Internet: inf@1september.ru
Fidonet: 2:5020/69.32
WWW: http://www.1september.ru