Миронов Илья Лазаревич

КОМАНДНЫЕ СОРЕВНОВАНИЯ ШКОЛЬНИКОВ ПО ПРОГРАММИРОВАНИЮ

Среди разнообразия школьных олимпиад, проводимых по многим предметам и по разным формулам, выделяется командная олимпиада школьников по программированию. Ее коренное отличие от иных соревнований состоит в том, что она предполагает не личное соперничество, а командную борьбу. Кроме того, для многих школьников она является не финальным этапом их участия в олимпиадном движении, а важной ступенькой на пути к студенческому чемпионату, проходящему по сходным правилам и являющемуся частью целой системы соревнований, организуемых ACM. В этом году командная олимпиада школьников прошла в нашем городе в шестой раз, во второй раз изменив свои правила и в первый – свое название.

Название олимпиады пришлось сменить в связи с тем, что сейчас она рассматривается как городской этап всероссийского чемпионата школьников. Его финал планируется провести в следующем году в нашем городе, а пока городская олимпиада стала называться “городским чемпионатом”. Свидетельством интереса школьников и преподавателей других регионов к этому событию является то, что в последние годы в городской олимпиаде традиционно принимает участие несколько приезжих команд. Несколько печальным завершением этой традиции оказалась безоговорочная победа иногородних команд, увезших в этом году три первых приза из Санкт-Петербурга. Будем надеяться, что на финал будущего российского чемпионата наш город сумеет выставить команду, готовую на равных соревноваться с командами других регионов.

Второе изменение правил за время существования этих соревнований в Санкт-Петербурге объясняется необходимостью расширить круг участников и адаптировать требования олимпиады к их возможностям. Дело в том, что в олимпиаде готовы принимать участие команды с чрезвычайно различающимся опытом и уровнем подготовки. Хотя технически это возможно (в Городском Дворце творчества юных достаточно компьютеров), составить такой набор задач, который не был бы слишком легок для сильнейших или слишком труден для слабейших участников, оказывается крайне трудным.

Если на студенческих олимпиадах команде засчитывается задача как решенная только тогда, когда она выдает правильные ответы на всех тестах, то опыт прошлых лет показал, что для школьников подобное требование является подчас непреодолимым препятствием. Неоднократно мы были свидетелями того, что сильная команда раз за разом посылает программу, которая проходит все тесты, кроме одного, самого первого, проверяющего элементарный частный случай. Так как тесты во время олимпиады участникам неизвестны, то команда продолжает биться над этой задачей, получая в ответ сообщения “Неправильный ответ”, падает духом и выходит из борьбы. Поскольку ни опыт школьников, ни их психологический настрой не помогают справиться с такими проблемами, особенно в условиях неполной информации, было принято решение допустить “неполную” сда-

чу задачи, когда команда набирает баллы за пройденные тесты даже за частично правильное решение.

Таким образом, формула олимпиады'98 выглядела так:

1) в любой момент команда может послать решение (исходный текст на Паскале или Си) любой задачи на проверку в жюри;

2) жюри проверяет задачу, пропуская ее на всех тестах. Тестовый комплект для всех участников одинаковый;

3) решение считается прошедшим тест, если оно уложилось в отведенное время (заранее известное участникам), программа вышла с нулевым кодом возврата (то есть не возникло ошибки времени исполнения) и результат работы программы, обычно записываемый в выходной файл, оказался правильным;

4) *оценка* команды за задачу определяется как максимальное количество тестов, пройденных одним из посланных решений. Если пройдены все тесты, то есть задача решена полностью, оценка команды удваивается;

5) *штрафное время* за задачу подсчитывается как время в минутах, прошедшее с начала тура до самого раннего подхода с максимальным пройденным количеством тестов (не меньше одного теста), плюс двадцать минут за каждый оставшийся подход;

6) выигрывает команда, набравшая максимальное суммарное количество пройденных тестов (с учетом премии за полностью решенные задачи). Если у двух команд количество пройденных тестов совпадает, то сравнивается штрафное время, набранное ими по всем задачам, по которым у них есть хотя бы один зачтенный тест.

Результаты прошедшей олимпиады сведены в таблицу (см. табл. 1). Из нее видно, что участники, особенно из второй половины списка, активно использовали нововведение, набирая по нескольку тестов на разных задачах. В то же время победители предпочитали решать задачи целиком, не размениваясь на отдельные тесты.

Примечание: *запись X/Y означает, что на данной задаче команда набрала X тестов, совершив Y подходов. Каждая задача тестировалась на 10 тестах, таким образом, максимальное количество тестов, которое можно было получить за задачу, равнялось 20 (с учетом премии за полностью сделанную задачу).*

Решая задачи олимпиады и пользуясь этой табличкой, вы можете сравнить свои результаты с результатами других участников. Имейте только в виду, что решать задачи дома за компьютером гораздо легче, чем в аудитории с дюжиной команд, в условиях ограниченного времени и крайнего волнения!

Ни одна олимпиада не могла бы пройти без помощи и содействия множества людей: жюри, волонтеров и сотрудников ГДТЮ. Особенно хотелось бы отметить вклад авторов большей части задач Олега Етеревского и Марка Сандлера, а также Николая Дурова, составившего тестовые комплекты.

Таблица 1. Результаты командной олимпиады школьников по программированию. Санкт-Петербург. 1998 г.

Место	Команда	A	B	C	D	E	F	G	Тестов	Время
1	Москва	20/1	20/1	20/1	20/1	20/2	20/1	20/4	140	886
2	Киров	20/1	20/1	20/3	20/1	20/1	20/3	7/10	127	1372
3	Саратов	20/1	20/1	20/1	20/1	20/1	20/1	1/3	121	987
4	Центр. сб.-3	20/1	20/1	20/1	20/1	20/1	20/1	1/1	121	171
5	Школа № 239-1	20/2	20/1	20/1	20/1	20/3	7/1	2/6	109	1299
6	Гомель-1	20/1	20/1	20/5		20/1	20/1	6/3	106	997
7	Школа № 30	20/1	20/1	20/1	20/1	20/2	0/1		100	635
8	Рыбинск	20/1	20/2	20/2	9/3	20/2			89	770
9	Сб. Кировск. р-на	20/3	20/4	20/8	3/2	20/2			83	1247
10	Екатеринбург	20/1	20/1	20/1	20/1				80	562
11	Школа № 489	20/2	20/1		20/1	20/1			80	664
12	Акад. гимназия-1	20/1	20/1	20/1	5/4	8/1	5/3	1/1	79	1252
13	ЦИК Кировск. р-на	20/1	20/1	20/1		7/4	7/4		74	588
14	ФТШ-2	20/1	20/2	20/1		8/2			68	631
15	Аничков лицей-2	20/1	20/1	8/1	6/1	8/7		2/2	64	1058
16	ФТШ-1	20/1	20/1	7/1	4/1	6/1	7/3		64	1123

17	Аничков лицей-1	20/2	20/2	3/1	6/2	8/5		4/1	61	1147
18	Пушкин	20/6	20/1			0/1	8/2	0/1	48	666
19	Калининск. р-н	7/3	20/1			20/4		0/2	47	621
20	Школа № 470-2	7/4	20/1		20/2				47	634
21	Школа № 470-1	7/2	20/1	7/2	9/3				43	813
22	Школа № 64	7/2	20/1		2/3	4/2	7/1	0/1	40	985
23	Школа № 139-1	7/1	8/2	2/2		20/2			37	741
24	Гомель-2	7/2	9/3	20/4					36	889
25	Школа № 521	9/6	20/5			4/5	3/6		36	1077
26	Школа № 344	7/1	20/1	1/1			7/2		35	787
27	Школа № 260	7/6	20/10			8/7			35	1004
28	"Охта"-1	8/1	20/1			5/2			33	686
29	Василеостр. р-н	8/2	20/1			4/3			32	780
30	Школа № 261-2		20/1		6/4	5/3			31	562
31	Школа № 239-2	9/3	20/1	1/2					30	553
32	Школа № 446	7/5	20/2			3/2			30	619
33	Школа № 426	1/1	20/1	2/1	1/1	6/2	0/1	0/1	30	1166
34	Школа № 261-1		20/1	9/1					29	346
35	ДТЮ Фрунз. р-на	0/1	20/3		1/2	1/1	7/1	0/1	29	976
36	Школа № 570		20/2		4/2	2/1			26	800
37	Школа № 292		20/1	0/1			5/1		25	318
38	Школа № 163		20/1		3/2				23	309
39	Великий Новгород	7/3	8/3	2/3		6/6			23	867
40	Школа № 330		20/1	0/2		1/1			21	482
41	Акад. гимназия-2		20/2						20	201
42	ЦСТТ Кировск. р-на	0/2	20/1						20	250
43	Школа № 402	7/1	8/7	2/1		2/3	0/1		19	1014
44	"Охта"-2	7/3	3/2			2/5			12	728
45	Центр подг. сбор.-1	8/4	0/2	1/3	1/2	1/1		1/1	12	1470
46	Центральный р-н, 1	4/1	5/1						9	380
47	Школа № 366	7/2	0/2						7	228
48	Красносельский р-н	2/4	0/3			4/1			6	374
49	Школа № 450		2/7	4/6					6	577
50	Школа № 139-2	4/2	0/7						4	290
51	Школа № 519	0/1			0/5				0	0
51	ЦСТТ Кировск. р-на		0/2						0	0
51	Центр подг. сбор.-2				0/5	0/1			0	0
51	Центр подг. сбор.-3		0/1						0	0
51	Компьютерный центр-2		0/6			0/9			0	0
51	Сб. Центрального р-на		0/1		0/2	0/1			0	0
51	ДДТ "У Вознесенского моста"								0	0

УСЛОВИЯ ЗАДАЧ

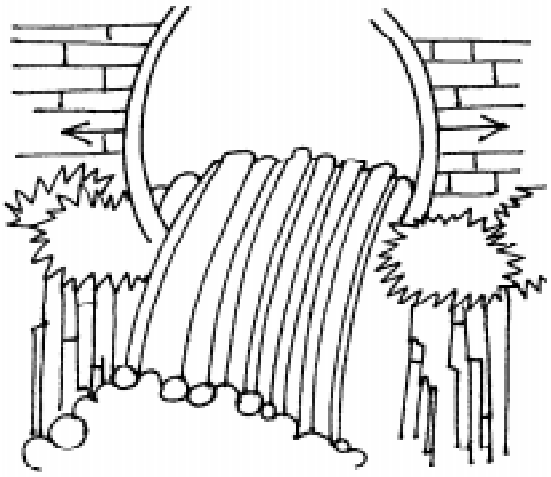
Задача A. "Раскрытие скобок"

Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение времени: 10 с. на каждый тест

Во входном файле задано алгебраическое выражение, являющееся произведением двух сумм нескольких переменных. Вам следует упростить это выражение, раскрыв скобки и приведя подобные слагаемые.

Формат входных данных.

В первой строчке входного файла input.txt записано выражение вида $(x_1 + x_2 + \ldots + x_k) \times (y_1 + y_2 + \ldots + y_l)$. Все переменные обозначаются маленькими буквами латинского алфавита. В каждой из



сумм переменные не повторяются. Пробелы и символы табуляции во входном файле не встречаются.

Формат выходных данных.

В выходной файл следует выдать сумму одночленов вида XY, 2XY или X^2. Порядок, в котором записываются сомножители и слагаемые, значения не имеет.

Пример входного файла.

```
(a+b)*(a+b)
```

Пример выходного файла.

```
a^2+2ab+b^2
```

Задача B. "Восстановление треугольника"

Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение времени: 10 с. на каждый тест

Во входном файле даны координаты середин сторон треугольника. Требуется по ним восстановить координаты вершин треугольника и вывести их в выходной файл.

Формат входных данных.

В первых трех строках входного файла записаны координаты середин сторон треугольника (x_1,y_1), (x_2,y_2), (x_3,y_3), при этом $0 \le x_i, y_i \le 100$ для $1 \le i \le 3$. Все координаты вещественные.

Формат выходных данных.

Распечатайте в выходной файл координаты вершин треугольника по одной на строке. Треугольник может оказаться вырожденным, проверять и отдельно обрабатывать этот случай не нужно. Ответ должен быть указан с точностью до 10^{-4}.

Пример входного файла.

```
2.5 1
1.5 2
3 2
```

Пример выходного файла.

```
1.0 1.0
2.0 3.0
4.0 1.0
```

Задача C. "Код с исправлением ошибок"

Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение времени: 10 с. на каждый тест

Для передачи данных по ненадежным каналам связи (например, по шумящей телефонной линии) применяются коды с исправлением или обнаружением ошибок. В этой задаче описывается один такой код и предлагается промоделировать его использование со стороны приемного устройства.

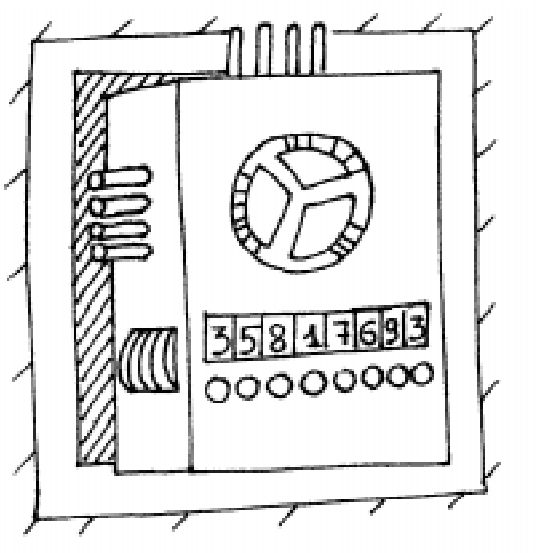
Будем кодировать четверки битов a_1, a_2, a_3, a_4. Вычислим дополнительно четыре бита $b_1=a_1 \text{xor} a_2$, $b_2=a_3 \text{xor} a_4$, $b_3=a_1 \text{xor} a_3$, $b_4=a_2 \text{xor} a_4$. Положим бит c равным $b_1 \text{xor} b_2$. Набор из девяти битов a_1, a_2, a_3, a_4, b_1, b_2, b_3, b_4, c будет кодировать исходную последовательность из четырех битов. Утверждается, что, если в этой последовательности из девяти битов в процессе пересылки была допущена ошибка (один из битов изменил свое значение), то эту ошибку можно исправить. Если же случилось две ошибки, то, хотя указать их позиции нельзя, можно об этом сигнализировать.

Формат входных данных.

В первой строчке записано количество N полученных кодовых последовательностей $1 \le N \le 100$. В N последующих строчках перечислены наборы из девяти нулей и единиц.

Формат выходных данных.

Для каждого набора из входного файла выведите на отдельной строке четверку битов, которые она кодировала, может быть,



исправив при этом одну ошибку. Если ошибок более одной, то выдайте сообщение “multiple error”.

Пример входного файла.

```
3
111100000
000111000
110110110
```

Пример выходного файла.

```
1111
0101
multiple error
```

Подсказка. Легче всего представить процесс кодирования, записав исходные и вычисляемые биты в такую табличку:

a_1	a_2	b_1
a_3	a_4	b_2
b_3	b_4	c

Тогда вычисление битов b и c будет выглядеть как сложение битов в соответствующих строках или столбцах.

Задача D. "Псевдоримская система счисления"

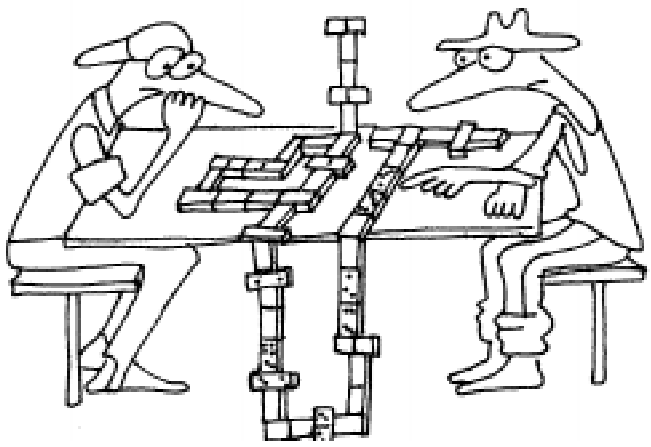
Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение времени: 10 с. на каждый тест

Рассмотрим систему счисления, которую мы назовем *псевдоримской*. В качестве цифр в ней, как и в обычной римской системе счисления, используются буквы I, V, X, L, C, D, M, обозначающие, соответственно, числа 1, 5, 10, 50, 100, 500, 1000. В этой системе счисления любая последовательность цифр является правильной записью некоторого числа. Пустая последовательность имеет значение ноль. В противном случае ищется самая большая цифра, участвующая в записи. Если таких цифр несколько, то выбирается самая левая из них. Тогда данная последовательность цифр является записью числа, равного сумме значения этой цифры и значения последовательности цифр, стоящих справа он него, за вычетом значения последовательности цифр, располагающихся левее. Например, запись XILIX соответствует числу 50+(10-1)-(10+1)=48.

Можно заметить, что в псевдоримской системе счисления одно число может иметь несколько различных записей. В этой задаче вы должны найти кратчайшую запись в псевдоримской системе счисления для данного числа.

Формат входных данных.

Во входном файле записано натуральное число N, $1 \le N \le 10000$.



Формат выходных данных.

Выведите в выходной файл самую короткую запись числа *N* в псевдоримской системе счисления. Если таких записей несколько, то выведите любую из них.

Пример входного файла.
41

Пример выходного файла.
IXL

Задача E. "Домино"

Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение времени: 10 с. на каждый тест

Некто в течение одной партии в домино вел запись последовательности, в которой доминошки выкладывались на стол. Постройте по этой записи конечную конфигурацию или сообщите, что это сделать невозможно.

Формат входных данных.

В первой строчке записано количество доминошек *N* (1≤*N*≤28), выложенных на стол. В следующих *N* строчках записаны сами эти доминошки в виде пар чисел от 0 до 6, разделенных одним или несколькими пробелами.

Формат выходных данных.

Если искомой конфигурации не существует, то выдайте в выходной файл строку “no solutions”. Иначе выведите в файл последовательность из доминошек, которая могла быть выложена в соответствии с правилами игры из доминошек, появляющихся в указанном порядке.

Пример входного файла.
3
1 2
2 3
1 3

Пример выходного файла.
1 2 2 3 3 1

Задача F. "Длинная арифметика"

Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение времени: 10 с. на каждый тест

Рассмотрим следующее преобразование целого числа *N*: находятся максимальное и минимальное число, получающиеся из него перестановкой цифр, и берется разность этих чисел. Например, если мы возьмем число 471, то такими двумя числами для него будут 741 и 147. Их разность, равная 594, и будет результатом применения преобразования к числу 471.

Так как всего чисел, не превосходящих некоторой степени десяти, конечное число, то эта последовательность с какого-то момента обязательно зациклится. К примеру, для уже рассмотренного числа получится такая последовательность:

471, 594, 495, 495, 495... Вам требуется определить для введенного числа длину такого цикла.

Формат входных данных.

Во входном файле записано натуральное число N, меньшее 10^{100}.

Формат выходных данных.

Выведите в выходной файл длину цикла, образуемого последовательными применениями данного преобразования к числу N.

Пример входного файла.

471

Пример выходного файла.

1

Задача G. "Записка Буратино"

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение времени:	10 с. на каждый тест

Закоренелые преступники лиса Алиса и кот Базилио похитили Буратино с целью получения выкупа от папы Карло. К счастью, Буратино смог передать через черепаху Тортилу записку, где он описал путь, по которому его везли от школы до места заточения. Так как Буратино затолкали в багажник машины, то он мог замечать только повороты и остановки на перекрестках. Помогите Мальвине, Пьеро и пуделю Артемону разыскать при помощи записки тот подвал, где страдает от сырости и пауков их деревянный друг.

Город, по которому везли Буратино, представляет собой прямоугольник $M \times N$ кварталов. Школа, где Буратино собирался учиться, находится в левом верхнем углу карты. Координаты перекрестка, на котором напали на Буратино, (0, M). По свидетельству очевидцев, машина похитителей начала двигаться от школы на восток.

В своей записке Буратино помечал то, куда повернула машина и сколько остановок перед светофорами она сделала. Разумеется, машине не нужно было останавливаться на зеленый свет, поэтому некоторые перекрестки Буратино мог проехать, их не сосчитав.

Формат входных данных.

В первой строке входного файла указаны числа M и N ($1 \le M,N \le 50$). Далее в файле приведена записка, доставленная Тортилой. В первой ее строке написано: “Меня запихнули в машину!”. В следующих строках находятся записи о том, как ехала машина (таких записей не более ста). Они бывают одного из двух форматов:

n раз остановились

или

повернули направо (налево)

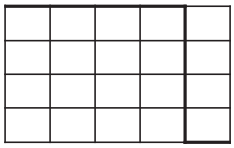
Последняя строка записки гласит: “Кто-то открывает багажник!”

Формат выходных данных.

Выведите в выходной файл координаты тех перекрестков, перед которыми могла остановиться машина.

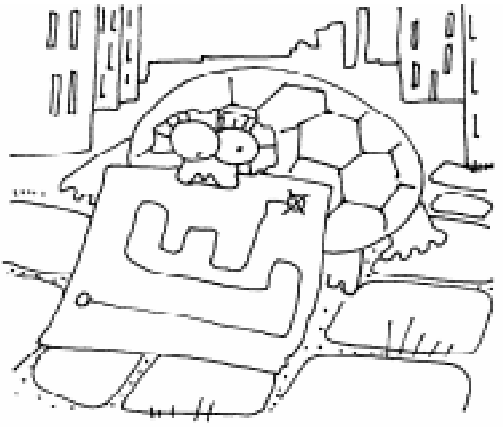
Пример входного файла.

4 5
Меня запихнули в машину!
3 раза остановились
повернули направо
4 раза остановились
повернули налево
Кто-то открывает багажник!



Пример выходного файла.

4 0
5 0



РАЗБОР ЗАДАЧ

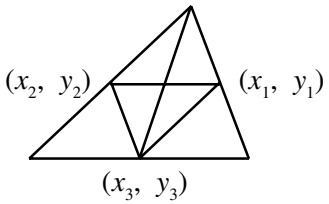
Задача A

В этой задаче основную трудность представляет чтение входных данных из файла. После этого рекомендуется заполнить квадратную табличку значениями коэффициентов при соответствующих одночленах. См. такую табличку для примера из условия (для экономии места опущены столбцы и строки, соответствующие буквам алфавита с c по z).

	a	*b*
a	1	1
b	1	1

Затем необходимо вывести эту табличку, складывая коэффициенты при равных одночленах.

Задача B



Решив элементарную систему уравнений или при помощи геометрического построения, изображенного на рисунке, можно убедиться, что координатами вершин треугольника будут следующие значения:

$$\left(\frac{x_1 + x_2 - x_3}{2}, \frac{y_1 + y_2 - y_3}{2}\right);$$
$$\left(\frac{x_1 + x_3 - x_2}{2}, \frac{y_1 + y_3 - y_2}{2}\right);$$
$$\left(\frac{x_2 + x_3 - x_1}{2}, \frac{y_2 + y_3 - y_1}{2}\right).$$

Задача C

Существует ровно 2^4=16 возможных четверок битов, а следовательно, и столько же их дополнений до девяти бит контрольными битами по приведенному в условии алгоритму. Если полученную приемником кодовую последовательность сравнить с каждой из этих девяток и найти ту, которая отличается от нее в минимальном числе разрядов, то можно определить, была ли передача произведена правильно. Если расхождений нет, то можно предположить, что ошибок не было. Если есть отличие в одном разряде, то исходная четверка восстанавливается единственным образом. Возможен также случай, что есть две правильные кодовые последовательности, которые отличаются от полученной в двух разрядах. Тогда можно сигнализировать об ошибке, но определить, какая именно последовательность была передана, невозможно.

Описанный в задаче алгоритм является частным случаем *кода Хемминга* с исправлением и обнаружением ошибок.

Задача D

Для приведенных в условии ограничений (N не превосходит 1000) умело организованный перебор укладывается в требуемые временные рамки. Можно организовать перебор рекурсивно, тогда для каждого N следует рассматривать две возможности: вычесть из N самую большую возможную цифру или дополнять N до ближайшей цифры.

Задача E

Для правильного решения этой задачи следует запрограммировать структуру данных, в которой было бы удобно сохранять растущую в обе стороны последовательность доминошек. Такой тип данных называется *деком* (линейный список с возможностью добавления/доступа/удаления элементов с обоих концов). В данной задаче доминошки можно хранить в массиве, индексируемом от –27 до 28, с указателями на начало и конец последовательности.

Можно заметить, что следующая доминошка выкладывается единственным образом, за исключением случая, когда это доминошка типа $A{:}B$ (и $A{\neq}B$), а последовательность на столе оканчивается с одного конца на A, а с другого на B. Тогда возможны два способа выложить доминошку на стол. В одном случае на столе окажется последовательность, оканчивающаяся и начинающаяся на A, в другом – с обоих концов будет B. Теперь достаточно выяснить, какова же следующая доминошка, оказавшаяся на столе. Она не может быть типа $A{:}B$ (такая была только что выложена), а значит, в ней присутствует либо A, либо B, но не оба числа вместе. Это и дает возможность определить, к какому концу была приложена предыдущая доминошка.

Задача F

Во-первых, в этой задаче нужно реализовать несколько операций "длинной" арифметики, то есть сложение и вычитание длинных чисел, не укладывающихся ни в один целочисленный тип данных используемого языка программирования.

Затем следует придумать способ сортировки цифр в числе. Самым эффективным алгоритмом решения этой задачи является *цифровая сортировка*, которая сводится к подсчету различных цифр, используемых в записи числа. Тогда самым большим числом, получающимся из данного перестановкой цифр, будет число, составленное из тех же цифр, расположенных в убывающем порядке. Аналогично, самое маленькое число получится, если выписать цифры по возрастанию.

Можно предложить разные алгоритмы поиска цикла в последовательности. Однако в этой задаче можно реализовать и самый безыскусный, так как длина предпериода не превосходит 100, а длина цикла оказывается всегда меньше десяти. То есть можно выписывать члены последовательности до тех пор, пока не получится уже встречавшийся.

Задача G

В условии есть несколько неоднозначностей, которые следует разрешить, привлекая свои знания реального мира и анализируя приведенный пример. Например, ясно, что похитители не стали бы вытаскивать Буратино из багажника на перекрестке, то есть последняя запись была сделана посередине какого-нибудь квартала. Кроме того, перед поворотом машина может остановиться на красный свет, то есть две записи "1 раз остановились", "повернули направо" могут относиться к одному маневру поворота направо.

Заметим, что направление движения машины определено единственным образом для любой строчки из записки. Давайте будем отмечать те кварталы, через которые может проезжать машина, после каждой записи. Если аккуратно отслеживать все ее возможные перемещения, то, в конце концов, у нас окажется список тех кварталов, где могла остановиться машина.

Эту задачу можно решать несколько иначе, если заметить, что множество кварталов, через которые может проезжать машина, всегда является прямоугольником. Если смотреть за изменениями размеров и перемещениями этого прямоугольника, то можно серьезно повысить эффективность программы, сделав ее независимой от величины города.

НАШИ АВТОРЫ

Миронов Илья Лазаревич,
председатель жюри
Командной олимпиады школьников
(СПб, 1998 г.)