

VI Республиканская командная олимпиада по программированию среди школьников

30 января – 2 февраля 2009 г.

Задачи, решения и итоги

Якутск
2009

Условия

Задача 1. Форумчик

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	4 мегабайта

Генотип нового существа Форумчик представляет собой цепочку генов, в которой все гены различны. Различный порядок следования генов дает нам различных особей. Ученые, к сожалению, не могут сразу получать существо с нужным порядком следования генов. Но они научились выводить существо, гены которого расположены по порядку с 1-го до N -го. Научились они также делать мутации, каждая из которых представляет собой обмен местами генов на i -ом и j -ом местах.

В результате таких мутаций была выведена очень агрессивная особь данного вида. К сожалению, ученые не совсем точно знают, в каком же порядке у него встали гены. Но, поскольку существо было получено с помощью мутогена, то у ученых есть запись последовательности мутаций, которые были сделаны.

Формат входных данных

В первой строке входного файла записано число N генов в генотипе и число M мутаций, которое было проведено. $1 \leq N \leq 250$, $0 \leq M \leq 2500$. Затем идет последовательность мутаций, каждая из которых задается двумя числами i и j — местами генов, которые меняются в результате мутаций.

Формат выходных данных

В выходной файл вывести последовательность генов особо агрессивной особи Форумчик.

Пример

input.txt	output.txt
5 4	5 3 1 2 4
1 5	
4 2	
2 5	
2 3	
6 0	1 2 3 4 5 6

Задача 2. Уникальная строка

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	4 мегабайта

Задан текстовый файл, длины строк которого не превышают 255 символов. В этом файле есть строка, которая повторяется нечетное количество раз, тогда как все остальные строки повторяются четное количество раз. Необходимо найти эту уникальную строку.

Формат входных данных

Входной файл содержит множество строк. Список завершается символом «#». Этот символ находится первым в последней строке и к списку не относится. Количество строк не превышает 10 000.

Формат выходных данных

В выходном файле записывается уникальная строка.

Пример

input.txt	output.txt
Эта строка - уникальна?	Последняя строка уникальна.
Нет	
Эта строка - уникальна?	
Нет	
Последняя строка уникальна.	#
#	

Задача 3. Black & White

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	6 мегабайт

Имеется n черных и белых карточек, сложенных в стопку. Карточки раскладываются на стол в одну линию следующим образом: первая кладется на стол, вторая под низ стопки, третья — на стол, четвертая — под низ стопки и т.д., пока все карточки не будут выложены на стол. Каким должно быть исходное расположение карточек в стопке, чтобы разложенные на столе карточки чередовались по цвету: белая, черная, белая, черная и т.д.

Формат входных данных

Входной файл содержит заданное натуральное число n ($1 \leq n \leq 10^6$).

Формат выходных данных

В выходной файл вывести через пробел n чисел (цвета в исходном расположении карточек): 0 — если карточка черного цвета, 1 — белого.

Пример

input.txt	output.txt
3	1 1 0

Задача 4. Матрица

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	4 мегабайта

Постройте квадратную матрицу $A[n \times n]$ следующим образом: элементы, составляющие прямоугольник со сторонами, параллельными диагоналям матрицы, заполните по часовой стрелке числами натурального ряда, начиная с элемента $A[1, 1+k]$, где k — заданное число; остальные элементы матрицы равны нулю.

Формат входных данных

Входной файл содержит в первой строке размерность матрицы n ($3 \leq n \leq 1000$), во второй строке целое число k ($1 \leq k \leq n - 2$).

Формат выходных данных

В выходной файл вывести построчно полученную матрицу.

Пример

input.txt	output.txt
5	0 0 1 0 0
2	0 8 0 2 0
	7 0 0 0 3
	0 6 0 4 0
	0 0 5 0 0

input.txt	output.txt
6	0 0 0 0 1 0
4	0 0 0 10 0 2
	0 0 9 0 3 0
	0 8 0 4 0 0
	7 0 5 0 0 0
	0 6 0 0 0 0

Задача 5. Туристы

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	2 мегабайта

Туристическая фирма одной страны решила организовать туристический маршрут из столицы S в один из населенных пунктов B , в котором имеется много исторических памятников. Маршрут проходит через некоторые населенные пункты. Так как туристы покупают в населенных пунктах сувениры, правительство страны заинтересовано в том, чтобы маршрут проходил через максимальное количество населенных пунктов. В промежуточных населенных пунктах гостиниц нет, поэтому маршрут должен быть однодневным. В день туристическая группа может пройти максимум s километров. Зная карту расположения n населенных пунктов, организовать туристический маршрут проходящий через максимальное количество населенных пунктов (один населенный пункт дважды посещать нельзя).

Технические требования Все населенные пункты пронумерованы. Карта населенных пунктов представляет собой двумерную матрицу A . Элементы этой матрицы $A[i, j]$ представляют собой длины дорог между пунктами с номерами i, j ; $1 \leq i, j \leq n$. Если между i -м пунктом и j -м пунктом нет дороги, то $A[i, j] = 0$.

Формат входных данных

В первой строке входного файла находятся числа n, x, y, s ($2 \leq n \leq 20$), означающие, соответственно, число населенных пунктов, номер начального пункта маршрута (S), номер конечного пункта маршрута (B) и длину однодневного пути которое могут пройти туристы. В следующих n строках находятся элементы матрицы A .

Формат выходных данных

Выходной файл состоит из единственного числа означающего максимальное количество населенных пунктов, которые могут посетить туристы.

Пример

input.txt	output.txt
4 1 4 37 0 10 25 15 10 0 20 0 25 20 0 10 15 0 10 0	3

Задача 6. Две лестницы

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	4 мегабайта

На горизонтальной площадке возведены две параллельные вертикальные стены. Лестница длиной a поставлена на землю у основания одной стены и прислонена к другой, лестница b поставлена у основания второй стены и прислонена к первой.

Каким должно быть расстояние между стенами для того, чтобы лестницы перекрещивались на высоте h над уровнем площадки?

Формат входных данных

В первой строке входного файла записаны три числа — a, b, h .

Формат выходных данных

В выходной файл вывести единственное число — расстояние между стенами с точностью до 0,001 или -1 , если задача не имеет решения.

Пример

input.txt	output.txt
1 1 0.25	0.866

Решения

Задача 1. Форумчик

Читаем из файла *M И N*, строим исходный массив по правилу $A[i] = i$, $1 \leq i \leq N$. Затем считываем из файла пары чисел i и j , и для каждой пары переставляем i -й и j -й элемент массива:

```
r:=a[i]; a[i]:=a[j]; a[j]:=r;
```

Задача 2. Уникальная строка

Рассмотрим два варианта решения.

1 вариант. Считываем данные в массив строк. Сортируем методом быстрой сортировки, что потребует $n \times \log n$ операций. Затем за один проход находим строку, которая встречается нечетное количество раз.

2 вариант. Используем операцию побитного исключающего ИЛИ — XOR. Напомним, что $x \text{ XOR } y = 0$ тогда и только тогда, когда $x = y$. Считываем первую строку в строковую переменную *a*. Считывая каждую последующую строку, посимвольно применяем к ней и к переменной *a* операцию XOR. В итоге вклад в *a* всех строк, встречающихся четное число раз, будет нулевой. Остается только позаботиться о длине выводимой строки, так как длина *a* равна наибольшей из длин считанных строк.

```
readln(a); readln(b);
repeat
  minl:=length(a);
  if length(b) <= length(a) then minl:=length(b)
  else
    for i:=minl+1 to length(b) do a:=a+b[i];
  for i:=1 to minl do a[i]:=chr( ord(a[i]) XOR ord(b[i]) );
  readln(b);
until b='#';
for i:=1 to length(a) do
  if a[i]<>'# then write(a[i]) else break;
```

Задача 3. Black & White

Проще всего взять и промоделировать раскладку карточек: берем стопку из n черных и белых карточек и начинаем расклад как говорится в задаче — первая на стол, вторая под низ, третья на стол, и т. д. при этом первой выложенной карточке приписывается белый цвет, а каждой

последующей — цвет, не совпадающий с цветом предыдущей карточки. Повторяем так, пока не найдем раскраску всех n карточек.

Для этой задачи возьмем такую структуру данных, как *список*. У списка есть начало и конец, и каждый элемент его указывает на следующий за ним элемент. Будем считать, что у последнего элемента списка указатель на следующий за ним равен 0. Список можно представлять массивом. Например, если в списке 4 элемента, то массив будет выглядеть так:

	1	2	3	4
A	2	3	4	0

т. е. за первым элементом находится $A[1] = 2$ элемент, ..., за 4-ым — $A[4] = 0$ (т.к. 4-ый элемент списка последний). Пусть у нас есть указатель на начальный элемент списка и на последний (BEG и FIN соответственно). В списке n элементов. Рассмотрим процедуру удаления элемента из начала списка (это соответствует переносу карточки на стол):

```
BEG:=A[BEG]; {теперь новый первый элемент списка - второй}
           {элемент старого списка}
```

Рассмотрим операторы перестановки элемента из начала списка в конец (это соответствует перемещению карточки сверху стопки под низ ее):

```
A[FIN]:=BEG; {следующей за последним элементом - бывший}
           {первый}
FIN:=BEG;    {меняем ссылку на последний элемент}
BEG:=A[BEG]; {новый первый элемент}
A[FIN]:=0;   {корректировка ссылки у последнего}
           {элемента}
```

Фрагмент программы:

```
for i:=1 to N-1 do A[i]:=i+1;
A[N]:=0; {установка ссылок в списке}
BEG:=1; FIN:=N;
COLOR:=1; {белый цвет = 1, черный = 0}
while A[BEG]<>0 do {пока первый элемент не является}
                 {одновременно и последним}
begin
  BEFORE:=BEG; {сохраняем индекс начала списка}
  BEG:=A[BEG]; {удаляем первый элемент из списка}
  A[BEFORE]:=COLOR; {раскрашиваем удаленный элемент}
                 {в нужный цвет}
  COLOR:=1-COLOR; {меняем цвет}
  A[FIN]:=BEG;    {переставляем элемент из}
  FIN:=BEG;      {начала списка в конец}
```

```

BEG:=A[BEG];
A[FIN]:=0
end;
A[BEG]:=COLOR; {раскрашиваем последний элемент}
                {списка}
for i:=1 to N do {распечатка цветов}
  write(A[i], ' ');

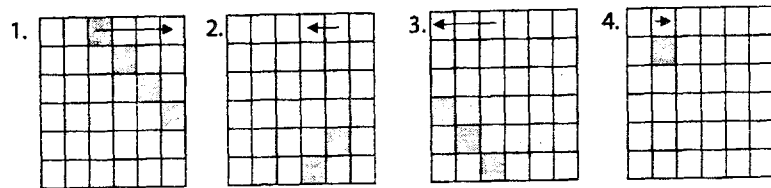
```

Задача 4. Матрица

Сначала все элементы матрицы обнуляем.

Заполнение прямоугольника числами натурального ряда идет по часовой стрелке по сторонам прямоугольника, параллельным главной и побочной диагоналям матрицы.

Для примера нарисуем случай $n = 6$, $k = 2$, выделяя рассматриваемую сторону цветом.



Первая сторона прямоугольника параллельна главной диагонали. Здесь номер столбца j увеличивается от $1+k$ до n , разность индексов $i-j$ равна $-k$.

Вторая сторона прямоугольника параллельна побочной диагонали. Здесь номер столбца j уменьшается от $n-1$ до $n-k$, сумма индексов $i+j$ равна $2n-k$.

Третья сторона прямоугольника параллельна главной диагонали. Здесь номер столбца j уменьшается от $n-k-1$ до 1 , разность индексов $i-j$ равна k .

Четвертая сторона прямоугольника параллельна побочной диагонали. Здесь номер столбца j увеличивается от 2 до k , сумма индексов $i+j$ равна $2+k$.

Фрагмент программы:

```

z:=0;
for j:=1+k to n do
  begin
    inc(z);
    a[j-k,j]:=z;
  end;
for j:=n-1 downto n-k do
  begin
    inc(z);
    a[2*n-k-j,j]:=z;
  end;
for j:=n-k-1 downto 1 do
  begin
    inc(z);
    a[j+k,j]:=z;
  end;
for j:=2 to k do
  begin
    inc(z);
    a[2+k-j,j]:=z;
  end;

```

Задача 5. Туристы

Применяется известный алгоритм проход графа в глубину. Для этого требуется вспомогательный одномерный массив r . В этом массиве отмечаются номера населенных пунктов, в которых туристы уже побывали. В еще одном вспомогательном одномерном массиве запоминаются номера населенных пунктов по пути из C в B . Имеются два счетчика, которые отмечают длину пройденного пути и количество пройденных населенных пунктов. Как только очередной путь дойдет до точки C происходит сравнение длины пройденного пути с s (максимально возможной длиной пути которую туристы могут пройти в день). В случае, если длина пройденного пути меньше s , происходит запоминание количества пройденных населенных пунктов. Если количество пройденных населенных пунктов какого-то очередного маршрута окажется больше, чем у предыдущих маршрутов, то это количество населенных пунктов будет считаться очередным максимальным количеством населенных пунктов пройденных по пути из C в B .

Задача 6. Две лестницы

Пусть AC — лестница длины a , BD — лестница длины b , $BC = x$ — расстояние между стенами, $AB = y$, $CD = z$, $EF = h$ и $BF = t$.

Тогда из подобия треугольников AGE и ABC следует, что $\frac{t}{x} = \frac{(y-h)}{y}$, а из подобия треугольников EFB и DCB — что $\frac{h}{z} = \frac{t}{x}$. Таким образом,

$$\frac{h}{z} = \frac{y-h}{y} \quad \text{и} \quad \frac{1}{y} + \frac{1}{z} = \frac{1}{h}.$$

Так как $y < a$, $z < b$, то необходимым условием существования решения является выполнение условия

$$\frac{1}{a} + \frac{1}{b} < \frac{1}{h}.$$

Если это условие не выполняется, то решения не существует.

Рассмотрим случаи, когда решение существует.

Так как x — расстояние между стенами, то по теореме Пифагора $x^2 = a^2 - y^2 = b^2 - z^2$, а отсюда следует соотношение $a^2 - b^2 = y^2 - z^2 = k^2$.

Если $a = b$, то $x = \sqrt{b^2 - 4h^2}$. Если $a \neq b$, то предположим, что $a > b$ и $y = \sqrt{z^2 + k^2}$, тогда верно уравнение

$$\frac{1}{\sqrt{z^2 + k^2}} + \frac{1}{z} = \frac{1}{h}.$$

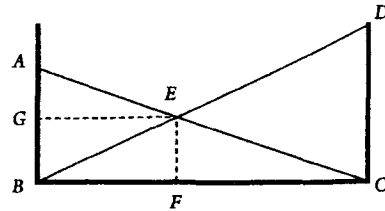
Это уравнение можно преобразовать к виду

$$h^2 z^2 = (z^2 + k^2)(z - h)^2.$$

Из предположения, что $a > b$ следует, что $h < z < 2h$. Рассмотрим функцию

$$f(z) = (z^2 + k^2)(z - h)^2 - h^2 z^2.$$

Имеем $f(h) = -h^4 < 0$, а $f(2h) = k^2 h^2 > 0$. Следовательно, между h и $2h$ обязательно существует корень, и притом только один. Он и является решением этого уравнения. Зная z , вычислим $x = \sqrt{b^2 - z^2}$. Найти корень z можно методом дихотомии.



Итоги

VI Республиканской командной олимпиады по программированию среди школьников

№	Команда	A	B	C	D	E	F	=	Время	Место	Диплом
1	Гаврильев, Никифоров, Федоров(РЛ)	+	+	+	+	2	+	6	735	1	I
2	Васильев, Сивцев, Дьячковский(Якутск)	+	+	4	+	1	-6	5	814	2	II
3	Семенов, Егоров, Иванов(Нюрбинский)	+	3	1	+		-1	4	493	3	II
4	Стальнов, Степанов, Николаев(Виллюйский)	+	+	-3	+			3	276	4	III
5	Егорова, Тихонова, Бортников(ЯГЛ, Якутск)	+	3	-2	+	-3	-2	3	319	5	III
6	Чарина, Слепцов, Самсонова(ЯГЛ, Якутск)	+	-8		+	+		3	407	6	III
7	Николаев, Миронов, Иванов(Таттинский)	+		-8	+			2	152	7	
8	Емельянов, Алексеев, Усов(Нюрбинский)	+	-2		+			2	177	8	
9	Абрамов, Ким, Байшев(РЛ)	+	-4	-2	+			2	198	9	
10	Гоголев, Чичигинарова, Мандарова(Чурапча)	1	-7	-4	+			2	203	10	
11	Евсеев, Семенов, Мекумянов(Сунтарский)	+	-7		+		-2	2	209	11	
12	Петальцев, Чернышев, Варгин(Нерюнгри)	+	-16	-4	1			2	217	12	
13	Пермяков, Никифоров, Заморщиков(РЛ)	+	-3	-8				1	75	13	
14	Сантаев, Иванов, Сергеев(В-Виллюйский)		-3	-5	+			1	142	14	
15	Антонов, Васильева, Васильев(В-Виллюйский)	+	-4					1	154	15	
16	Дмитриев, Григорьев, Нюрбинцев(Виллюйский)	2		-1				1	160	16	
17	Колодезников, Сыромятников, Захаров(Томпо)	3	-6					1	343	17	
18	Парняков, Проккопьев, Нератов(Намский)	4						1	365	18	
19	Григорьев, Габышев, Стручков(М-Кангал)	-2		-1	-1			0	0	19	
20	Спиридов, Ю, Назаров(Сунтарский)						-2	0	0	19	
21	Лукин, Антонов, Филиппов(Хангаласский)			-2				0	0	19	
22	Ануфриев, Федорова, Захарова(Томпонский)			-1				0	0	19	
23	Слепцов, Гоголев, Байшев(Хагассы)	-1		-4				0	0	19	
24	Никифоров, Шараборин, Неуструев(Хангал)							0	0	19	

Содержание

Условия	1
Задача 1. Форумчик	1
Задача 2. Уникальная строка	2
Задача 3. Black & White	2
Задача 4. Матрица	3
Задача 5. Туристы	4
Задача 6. Две лестницы	5
Решения	6
Задача 1. Форумчик	6
Задача 2. Уникальная строка	6
Задача 3. Black & White	6
Задача 4. Матрица	8
Задача 5. Туристы	9
Задача 6. Две лестницы	10
Итоги	11

Сборник содержит задачи VI командной олимпиады по программированию среди школьников. Олимпиада прошла с 30 января по 2 февраля 2009 г. на базе Физико-математического форума «Ленский край» в с. Чапаево.

Олимпиада прошла в один пробный и один основной тур. На основном туре 1 февраля участникам было предложено за 5 часов решить шесть задач. Правила были обычными для командных олимпиад по программированию: каждой команде из 3 школьников был выделен один компьютер, решения принимались и проверялись по сети автоматической системой.

Жюри олимпиады:

Н. Н. Павлов — председатель,
Ю. С. Антонов,
Н. В. Николаева,
А. В. Павлов,
Т. Г. Протодяконова,
Е. С. Рожина.

Компьютерная верстка А. В. Павлова