

**Московский государственный технический университет
имени Н.Э.Баумана**

Олимпиада школьников «Шаг в будущее»

Инженерное дело «Профессор Лебедев» ИНФОРМАТИКА 2 тур

2018-2019 учебный год

11 класс

Вариант 3

Задача 1

Для сокрытия своих записей некий волшебник решил использовать не общепринятую систему счисления. Однако, страдая забывчивостью, дописывал в каждом трактате и основание этой системы счисления, и обозначения элементов алфавита. Помогите чародею понять свои прошлые записи, но, чтобы не обидеть старика, дайте ответ в указанной им системе счисления.

На вход подаётся строка с основанием используемой системы счисления и перечислением элементов алфавита, а также выражение, к которому нужно указать ответ. Ответ дать в указанной системе счисления в формате строки.

Входная строка представляет собой число-основание системы счисления и последовательность разрядов, записанных через произвольное число пробелов по возрастанию значения, после чего – строковая запись выражения.

Выходной строкой является строка, содержащая ответ, который необходимо найти по заданию.

Изменять формат входных и выходных данных запрещено.

Оставлять в конце программы ожидание ввода запрещено.

Помните, что автоматическая проверка населена роботами.

Входная строка: 3 0 1 2 21+20

Результат: 111

Критерии: Простая задача – 5 баллов

N	Оценка	Входные данные	Выходные данные
1	1	4 0 1 2 3 11+10	21
2	1	5 0 1 2 3 4 31+20	101

3	1	3 0 1 2 21+20	111
4	1	7 0 1 2 3 4 5 6 51+50	131
5	1	5 a b c d e db+ca	bab

Решение:

```
#include "stdafx.h"
```

```
#include <iostream>
```

```
#include <string>
```

```
#include <cmath>
```

```
std::string convertSingleTo(int num, char *symbols)
```

```
{
    return std::string() + symbols[num];
}
```

```
std::string convertTo(int num, int base, char *symbols)
```

```
{
    if(abs(num) < base) return convertSingleTo(abs(num), symbols);
    if(num < 0)
        return "-" + (convertTo(abs(num)/base, base, symbols) +
convertSingleTo(abs(num)%base, symbols));
    return convertTo(abs(num)/base, base, symbols) + convertSingleTo(abs(num)%base,
symbols);
}
```

```
int convertSingleFrom(char num, char *symbols)
```

```
{
    int i=0;
    while(symbols[i] != num) i++;
    return i;
}
```

```
int convertFrom(std::string num, int base, char *symbols)
```

```
{
    int k=1;
    if(num[0] == '-')
    {
        k = -1;
        num.erase(num.begin());
    }
    int result = 0;
    for(int i=num.length()-1; i>=0; i--)
```

```

        result += convertSingleFrom(num[i], symbols)*pow(base, num.length()-1-i);
    return k*result;
}

```

```

int findOperandPos(std::string str)
{
    char *operands = new char[5];
    operands[0] = '+';
    operands[1] = '-';
    operands[2] = '*';
    operands[3] = '/';
    operands[4] = '%';
    for(int i=0; i<str.length(); i++)
        for(int k=0; k<5; k++)
            if(str[i] == operands[k]) return i;
    return -1;
}

```

```

std::string doTask(std::string task, int base, char *symbols)
{
    for(int i=task.length(); i>=0; i--)
        if((task[i] == ')') || (task[i] == '('))
            task.erase(task.begin() + i);

    int first = 1;
    if(task[0] == '-')
    {
        task.erase(task.begin());
        first = -1;
    }

    int operand = findOperandPos(task);
    first *= convertFrom(task.substr(0, operand), base, symbols);

    task.erase(task.begin(), task.begin() + operand);

    int second = 1;
    if(task[1] == '-')
    {
        task.erase(task.begin() + 1);
        second = -1;
    }
    second *= convertFrom(task.substr(1, task.length()-1), base, symbols);

    int result;
    switch(task[0])

```

```

        {
            case '+': result = first+second; break;
            case '-': result = first-second; break;
            case '*': result = first*second; break;
            case '/': result = first/second; break;
            case '%': result = first%second; break;
            default: result = -1; break;
        }

        return convertTo(result, base, symbols);
    }

int main()
{
    int base;
    std::cin>>base;
    char *symbols = new char[base];
    for(int i=0; i<base; i++)
        std::cin>>symbols[i];
    std::string task;
    std::cin>>task;

    std::string result = doTask(task, base, symbols);
    std::cout<<result<<"\n";
    return 0;
}

```

Задача 2

После того как ученик уронил тетрадь с примерами на пол, из тетради выпали все арифметические знаки (сложение, вычитание, умножение и целочисленное деление, а также остаток от деления). На вход программе подаётся строка с целыми числами, помогите расставить выпавшие знаки так, чтобы получилось верное равенство. Ответ запишите строкой, указывая через пробел символы арифметических операций (+, -, *, / – целочисленное деление, % – остаток от деления; операции указаны в порядке убывания частоты использования). В случае, если таких записей несколько укажите вариант с наиболее часто употребляемыми операциями.

Входная строка представляет собой последовательность чисел, записанные через произвольное число пробелов.

Выходной строкой является строка, которую необходимо найти по заданию.

Изменять формат входных и выходных данных запрещено.

Оставлять в конце программы ожидание ввода запрещено.

Помните, что автоматическая проверка населена роботами.

Входные данные: 2 9 9 9

Результат: * = +

Критерии: простая 15 баллов

N	Оценка	Входные данные	Выходные данные
1	1	5 2 14 7	+ = -
2	1	9 2 7 3 3	- - + =
3	1	2 9 9 9	* = +
4	1	3 6 10 8	* = +
5	5	1 2 3 2 17 8	+ + + % =
6	1	6 9 3	= -
7	5	81 27 9 0 6	/ - = -

Решение:

```
s=list(map(int, input().split()))
n=len(s)
slf={0:"+", 1:"- ", 2:"*", 3:"/", 4:"%", 5:"==", 6:"="}
a=[0]*(n-1)
```

```
f=0
```

```
while ( sum(a)<4*(n-1) and f==0):
```

```
    for j in range(n-1):
```

```
        q=a[j]
```

```
        a[j]=5
```

```
        ans=""
```

```
        for i in range(n-1):
```

```
            ans+=str(s[i])+slf[a[i]]
```

```
        ans+=str(s[-1])
```

```
    if eval(ans) and f==0:
```

```

f=1
for k in range(n-1):
    w=a[k]
    if w==5:
        w+=1
    print(slf[w], end=" ")
print()

```

a[j] = q

```

o=1
for i in range(n-1):
    a[i]+=o
    o=a[i]//5
    a[i]=a[i]%5

```

Задача 3

При анализе генома человека иногда есть необходимость находить повторяющиеся последовательности в строке, состоящей из алфавита А,С,Г,Т. Необходимо составить алгоритм, работающий для произвольного генома, определяющий в заданной строке наиболее длинную подстроку, которая повторяется заданное количество раз (или вернуть пустую строку).

Входная строка представляет собой число повторений и заданная строка, записанные через произвольное число пробелов.

Выходной строкой является строка, которую необходимо найти по заданию.

Изменять формат входных и выходных данных запрещено.

Оставлять в конце программы ожидание ввода запрещено.

Помните, что автоматическая проверка населена роботами.

Входная строка: 3 ACGCCGCTTTTCCGGTTCGC.

Результат: CGC

Пояснение: последовательность TT тоже подойдёт, но она более короткая, чем CGC.

Критерии: средняя 15 баллов

N	Оценка	Входные данные	Выходные данные
---	--------	----------------	-----------------

1	1	3 ACGCCGCTTTCCGGTTCGC	CGC
2	1	3 ухогорлоносоговорилородогурцов	ого
3	1	3 inhfa pf ghtdsitybt crjhjcnb cnj, inhfa pf gmzycndj ldf, inhfa pf ghjcnj nfr vyuj	inhfa pf
4	1	2 перпендикуляр, опущенный из заданной точки К на плоскость, будет также параллелен прямой АС, которая вместе с заданной точкой К образует плоскость.	плоскость
5	5	1000 На краю дороги стоял дуб. Он был, вероятно, в десять раз старше берез, составлявших лес, в десять раз толще и в два раза выше каждой березы. Это был огромный, в два обхвата дуб, с обломанными суками и корой, заросшей старыми болячками. С огромными, неуклюже, несимметрично растопыренными корявыми руками и пальцами, он старым, сердитым и презрительным уродом стоял между улыбающимися березами.	
6	5	0	
7	1	4 ехать на конференцию на верхней полке на две недели на поезде	на

Решение:

```

var
  n,i,i2,i3,c:longint;
  s,current: string;
  checked: array[1..100000] of string;
  function rec(st: string;depth: longint; j: longint):boolean;
  begin
    if not (st[depth]=s[j]) then begin rec:= false; exit; end;
    if depth = length(st) then begin rec:= true; exit; end else rec:=rec(st,depth+1,j+1);
  end;

function count(str: string):longint;
var
  j,cou,ans: longint;
begin
  ans:=0;

```

```

j:=1;
for j:=1 to length(s) do
begin
  if str[1]=s[j] then if rec(str,1,j) then ans:=ans+1;
end;
count:=ans;
end;

procedure check(str: string);
var
  j: longint;
begin
  for j:=1 to c do
  if checked[j]=str then exit;
  c:=c+1;
  checked[c]:=str;
  if count(str)=n then begin writeln(str); halt(); end;
end;

begin
read(n);
readln(s);
c:=0;
//      n:=2;
// s:='BCCBCCBCBCCBC';
current:="";
for i:=length(s) downto 1 do
for i2:=1 to length(s)-i do
begin
  for i3:=i2 to i2+i-1 do
  begin
    current:=current+s[i3];
  end;
  check(current);
  current:="";
end;
end.

```

Задача 4

Можно ли распределить группу студентов, с указанными во входной строке весами, на две примерно равные по общему весу группы? И если можно, то какими будут веса каждой группы? (ответ запишите в порядке убывания значения общего веса).

Необходимо написать консольное приложение.

Изменять формат входных и выходных данных запрещено.

Оставлять в конце программы ожидание ввода запрещено.

Помните, что автоматическая проверка населена роботами.

На вход подаётся строка с целыми числами, записанными через пробелы. Результатом является строка, состоящая из двух чисел, записанных через пробел.

Входные параметры	Выходные параметры
5 1 100 20 40 30 50	125 121
1 10	10 1

Критерии: средняя 18 баллов

N	Оценка	Входные данные	Выходные данные
1	5	5 1 100 20 40 30 50	125 121
2	5	1 10	10 1
3	8	5 10 95 75 100 115 20	210 210

Решение:

```
program Project4;
  type mas=array[1..100] of integer;
  var s2,s1:string; m:mas; i,x,y, n,c:integer; f:boolean;
  { procedure rec(s1,s2:string;x,y:integer) ;
  var a,c:integer; s3:string;
  begin
    if length(s1)=0 then
      if x=y then readln(s2);
    if length(s1)>0 then
      begin
        s3:=copy(s1,pos(' ',s1)+1,length(s1)-pos(' ',s1));
        val(copy(s1,1,pos(' ',s1)-1),a,c);
        if(length(s2)=0) then rec(s3,s2,a,y)
        else
          begin
            rec(s3,s2+"+",x+a,y);
            rec(s3,s2+"-",x-a,y);
            rec(s3,s2+"*",x*a,y);
```

```

        rec(s3,s2+ "/" ,x/a,y);
    end;

    end;
end;
}
function sum(x,y:integer;m:mas):integer;
var i,s:integer;
begin
    s:=0;
    for i:= x to y do
        s:= s+m[i];
        sum:=s;
    end;
begin
    readln(s1);
    n:=1;
    f:=true;
    while(pos(' ',s1)>0) do
        begin
            val(copy(s1,1,pos(' ',s1)-1),m[n],c);
            delete(s1,1,pos(' ',s1));
            n:=n+1;
            end;
        val(s1,m[n],c);

    while(f) do
        begin
            f:=false;
            for i:= 1 to n-1 do
                if m[i]<m[i+1] then
                    begin f:= true;c:= m[i];m[i]:=m[i+1];m[i+1]:=c;end;
                end;
            for i:= 1 to n do
                begin
                    if x>y then y:=y+m[i]
                    else x:=x+ m[i];
                end;
            if x>y then writeln(x,' ',y)
            else writeln(y,' ',x) ;
        end.

```

Задача 5

В Бородинском сражении солдаты шли в атаку в построении, сходным с прямоугольником со сторонами М и N человек. Во время движения важно «чувство локтя» в ряду, т.е. рост солдат в каждом ряду должен убывать от левого края к правому (от 0 индекса к старшему). Также важно, чтобы первый ряд состоял из наименее рослых солдат, а последний – наоборот. К сожалению, солдаты успели сдружиться рядами, поэтому перемещать солдат из одного ряда в другой нельзя. Расставьте солдат так, чтобы повысить качество их перемещения.

На вход подаётся матрица, где каждая ячейка – рост. На выходе – изменённая матрица в виде строки.

Входная строка представляет собой последовательность чисел, записанные через произвольное число пробелов. Первые два числа – количество столбцов и строк.

Выходной строкой является строка с числами, записанными через пробел, которую необходимо найти по заданию.

Изменять формат входных и выходных данных запрещено.

Оставлять в конце программы ожидание ввода запрещено.

Помните, что автоматическая проверка населена роботами.

Входная строка: 3 3 9 3 5 1 2 6 2 3 4

Результат: 6 2 1 4 3 2 9 5 3

Критерии: сложная 22 баллов

N	Оценка	Входные данные	Выходные данные
1	7	6 3 9 5 8 3 7 4 6 1 2 1 9 7 4 3 6 2 5 5	6 5 5 4 3 2 9 7 6 2 1 1 9 8 7 5 4 3
2	7	3 3 9 3 5 1 2 6 2 3 4	6 2 1 4 3 2 9 5 3
3	8	0 0	

Решение:

```
l = list(map(int,input().split()))
```

```
n,m = l[0],l[1]
```

```
l = l[2:]
```

```
#n,m = map(int, input().split())
```

```
a = []
```

```

for i in range(m):
    a.append(l[:n])
    l = l[n:]
    #a.append(list(map(int,input().split())))
    a[i].sort()
    a[i].reverse()
for i in range(len(a)):
    for j in range(i,len(a)):
        if j>i and sum(a[i])>sum(a[j]):
            arr = a[i]
            a[i] = a[j]
            a[j] = arr
ans = []
for i in range(m):
    ans = ans+a[i]
print(*ans)

```

Задача 6

Планктонный рачок церкопагис (*Cercopagis pengoi*) обладает длинным хвостом (иногда длина хвоста превосходит длину тела в 7 раз). Одной из гипотез о назначении подобного хвоста, который, кроме прочего, наделён шипами, является регуляция численности: хвосты спутываются и мешают свободному перемещению для добывания пищи, что приводит к голодной смерти животного. Очевидно, что чем длиннее хвост, тем больше собратьев могут в нём запутаться. Шипы разбросаны зонами по всей длине хвоста. Считается, что планктонные рачки спутываются именно этими зонами.

На вход подаётся количество зон с шипами у каждого рачка. Определите наибольшее число церкопагисов, которые могут спутаться в одну группу.

Входная строка представляет собой последовательность целых чисел, записанных через произвольное число пробелов.

Выходной строкой является строка, содержащая число, которое необходимо найти по заданию.

Изменять формат входных и выходных данных запрещено.

Оставлять в конце программы ожидание ввода запрещено.

Помните, что автоматическая проверка населена роботами.

Входные данные: 3 2 1 1 1 1 2 1

Результат: 6

Критерии: сложная 25 баллов

N	Оценка	Входные данные	Выходные данные
1	1	3 1 1 1 1 1 1 1 1 2 1 1 1 1	5
2	1	3 1 1 1 1 1 1 5 1 1 1 1 1 1 1 1 1 1	8
3	1	1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 4	8
4	1	3 2 1 1 1 1 1 2 1	6
5	8	9 5	2
6	7	2 2 2 2 2 2 8	8
7	6		0

Решение:

```
a = input().split()
rez = 0
lenth = 0
notfind = 1
for i in a:
    i = int(i)
    if notfind:
        rez += i+1
        notfind = 0
    else:
        rez += i-1
    lenth += 1
print(min(rez,lenth))
```