**Олимпиада школьников «Шаг в будущее» по общеобразовательному предмету "Информатика".**
**2015 год. Заключительный этап. 8-9 классы. Билет 1. Задачи и решения.**

**Задача 1 (12 баллов).** Вычислить сумму ряда с точность ε, если известно, что предел этой суммы существует.

$$\frac{1}{1\cdot 2}+\frac{1}{2\cdot 3}+\frac{1}{3\cdot 4}+\ldots+\frac{1}{n\cdot(n+1)}+\ldots$$

**Входные данные.** Стандартный входной поток содержит одно действительное число ε (0<ε<1).
**Выходные данные.** В стандартный выходной поток вывести одно действительное число – сумму ряда.

**Задача 2 (16 баллов).** Найти количество действительных корней алгебраического уравнения
$a_0x^5 + a_1x^4 + a_2x^3 + a_3x^2 + a_4x + a_5 = 0$ на отрезке $[-n, n]$.
**Входные данные.** Стандартный входной поток содержит шесть действительных чисел $a_0$, $a_1$, $a_2$, $a_3$, $a_4$, $a_5$ и одно целое число $n$ ($1 \le n \le 100$).
**Выходные данные.** В стандартный выходной поток вывести одно целое число – число корней уравнения.

| Примеры входных данных | Примеры выходных данных |
|---|---|
| 1.0 2.5 -1.0 -3.5 -1.0 0.1 10 | 5 |
| 1.0 1.0 1.0 -1.0 -1.0 0.1 5 | 3 |
| 1.0 -1.0 1.0 -1.0 1.0 -1.0 2.5 | 1 |

**Задача 3 (16 баллов).** Дана строка, в которой записано арифметическое выражение. Выражение может содержать пробелы и круглые скобки, в качестве операндов используются целые константы и идентификаторы, в качестве операторов используются знаки '+', '–', '*' и '/'. Целая константа может состоять из любого количества десятичных цифр. Идентификатор может состоять из любого количества латинских букв в верхнем или нижнем регистре. При этом прописные и строчные буквы не различаются. Подсчитать количество различных идентификаторов в выражении.
**Входные данные.** Стандартный входной поток содержит одну строку, в которой записано арифметическое выражение. Длина выражения не больше 100.
**Выходные данные.** В стандартный выходной поток вывести одно целое число – количество различных идентификаторов.

| Примеры входных данных | Примеры выходных данных |
|---|---|
| bBbBbBbBbBbBbBbBbBbB + bbbbbbbbbbbbbbbbbbbb | 1 |
| (a+a*(a+a*(a+a*(a)))) | 1 |
| (n - 1)*(n - 1)*(n - 1) + m*m*m | 2 |

**Задача 4 (24 балла).** На плоскости дан простой многоугольник (т. е. без самокасаний и самопересечений, но не обязательно выпуклый). Вычислить площадь многоугольника.
**Входные данные.** Стандартный входной поток содержит целое число $N$ ($3 \le N \le 1000$) – количество вершин многоугольника и последовательность из $N$ пар действительных координат вершин многоугольника. Все координаты по модулю не больше $10^6$. Вершины многоугольника заданы в порядке их обхода против часовой стрелки.
**Выходные данные.** В стандартный выходной поток вывести одно действительное число – площадь многоугольника. Результат вывести с точностью 0.0001.

| Примеры входных данных | Примеры выходных данных |
|---|---|
| 4 0 0 1 0 1 1 0 1 | 1.0000 |
| 4 0 0 3 0 1 1 0 3 | 3.0000 |

**Задача 5 (32 балла).** Дана целочисленная таблица размера $M$x$N$. Надо пройти из верхней строки в нижнюю строку, набрав максимальную сумму чисел в пройденных клетках. Начинать можно с любой клетки верхней строки и заканчивать в любой клетке нижней строки. Каждую клетку можно проходить только один раз. На каждом шаге можно переходить в одну из трех нижних соседних клеток: нижнюю, нижнюю левую или нижнюю правую. Из клеток первого и последнего столбцов возможны только два направления перехода.
**Входные данные.** Первая строка стандартного входного потока содержит два числа: $M$ ($2 \le M \le 100$) и $N$ ($2 \le N \le 100$). Следующие $M$ строк содержат по $N$ неотрицательных целых чисел. Значение клетки таблицы не больше 100.
**Выходные данные.** В стандартный выходной поток вывести два целых числа: номер (по порядку слева направо) клетки первой строки, из которой надо начать переход, и максимальную сумму чисел в пройденных клетках. Если есть несколько путей с максимальной суммой, вывести любой из них.

| Пример входных данных | Пример выходных данных |
|---|---|
| 4 5<br>0 12 10 0 5<br>0 20 10 5 2<br>7 5 2 3 0<br>9 10 10 2 0 | 2 49 |

**Решение задачи 1.**

```
#include "stdafx.h"

int _tmain(int argc, _TCHAR* argv[])
{
	int n = 1;
	double t = 0.5, s = t, eps;
	scanf( "%lf", &eps );
	while (t > eps)
	{
		n++;
		t = 1.0/n/(n+1);
		s += t;
	}
	printf( "%lf\n", s );
	return 0;
}
```

**Решение задачи 2.**

```
#include "stdafx.h"

double f( double a[], double x )
{
	return a[0]*x*x*x*x*x + a[1]*x*x*x*x + a[2]*x*x*x + a[3]*x*x + a[4]*x + a[5];
}

int _tmain(int argc, _TCHAR* argv[])
{
	double a[6];
	double n, x, h = 0.001;
	int k = 0;
	for (int i = 0; i < 6; i++)
		scanf( "%lf", &a[i] );
	scanf( "%lf", &n );
	x = -n;
	while (x < n)
	{
		if (f( a, x )*f( a, x+h ) < 0)
			k++;
		x += h;
	}
	printf( "%d\n", k );
	return 0;
}
```

**Решение задачи 3.**

```
#include "stdafx.h"
#include <string.h>

int find( char *word, char **arr, int size )
{
	for ( int i = 0; i < size; i++ )
		if ( strcmp( word, arr[i] ) == 0 )
			return i;
	return -1;
}

int _tmain(int argc, _TCHAR* argv[])
{
	char line[101];
	char *words[50];
	char seps[] = " \t+-*/()0123456789";
	char *copy, *token;
	int index, n = 0;
	for ( int i = 0; i < 50; i++ )
		words[i] = NULL;
	gets( line );
	copy = strupr( strdup( line ) );
	token = strtok( copy, seps );
	while ( token != NULL )
	{
```

```
            if ( (index = find( token, words, n ) ) == (-1) )
                  words[n++] = token;
            token = strtok( NULL, seps );
      }
      printf( "%d\n", n );
      return 0;
}
```

**Решение задачи 4.**

```
#include "stdafx.h"
#include  <math.h>

int _tmain(int argc, _TCHAR* argv[])
{
      int N;
      double x_curr, x_next, y_curr, y_next, x_start, y_start;
      double area = 0;
      scanf( "%d", &N );
      scanf( "%lf %lf", &x_next, &y_next );
      x_start = x_next;
      y_start = y_next;
      for (int i = 2; i <= N; i++)
      {
            x_curr = x_next;
            y_curr = y_next;
            scanf( "%lf %lf", &x_next, &y_next );
            area += 0.5*(y_next + y_curr)*(x_next - x_curr);
      }
      x_curr = x_next;
      y_curr = y_next;
      x_next = x_start;
      y_next = y_start;
      area += 0.5*(y_next + y_curr)*(x_next - x_curr);
      printf( "%12.4lf\n", abs( area ) );
      return 0;
}
```

**Решение задачи 5.**

```
uses
  SysUtils;

const
  MAXX = 100; MAXY = 100;
var
  XSz, YSz, // размеры таблицы
  i, j, // номера строк и столбцов
  m: // x-координата клетки с максимальной оценкой  в текущей строке
  byte;
  est: array[1..2, 1..MAXX] of integer; // предыдущая и текущая строки таблицы оценок
  choice: array[2..MAXY, 1..MAXX] of integer; // таблица выборов
  v, e: integer;

begin
  readln(XSz, YSz);
  // первая строка данных становится первой строкой таблицы оценок
  for j := 1 to XSz do read(est[1, j]);
  // остальные строки вычисляются с учетом предыдущих
  for i := 2 to YSz do
  begin
    // обрабатываем очередную строку входных данных
    for j := 1 to XSz do
    begin
      read(v);
      e := -1;
      if (j > 1) and (est[1, j-1] > e) then
      begin
        e := est[1, j-1];
        choice[i, j] := -1;
      end;
      if est[1, j] > e then
      begin
        e := est[1, j];
        choice[i, j] := 0;
```

```
      end;
      if (j < Xsz) and (est[1, j+1] > e) then
      begin
        e := est[1, j+1];
        choice[i, j] := 1;
      end;
      est[2, j] := v + e;
    end;
    // текущая строка станет предыдущей на следующем шаге
    for j := 1 to Xsz do est[1, j] := est[2, j]
  end;

  readln;
  // обратный ход
  e := -1;
  m := 0;
  // в последней строке ищется клетка с наибольшей оценкой
  for j := 1 to XSz do
    if est[1, j] > e then
    begin
      e := est[1, j];
      m := j;
    end;
  for i := YSz downto 2 do m := m + choice[i, m];
  writeln;
  writeln(m:10, e:10);
  readln;
end.
```

**Олимпиада школьников «Шаг в будущее» по общеобразовательному предмету "Информатика".**
**2015 год. Заключительный этап. 8-9 классы. Билет 2. Задачи и решения.**

**Задача 1 (12 баллов).** Вычислить сумму ряда с точность ε, если известно, что предел этой суммы существует.

$$\frac{1}{1\cdot 2\cdot 3}+\frac{1}{2\cdot 3\cdot 4}+\frac{1}{3\cdot 4\cdot 5}+\ldots+\frac{1}{n\cdot(n+1)\cdot(n+2)}+\ldots$$

**Входные данные.** Стандартный входной поток содержит одно действительное число ε ($0 < \varepsilon < 1$).
**Выходные данные.** В стандартный выходной поток вывести одно действительное число – сумму ряда.

**Задача 2 (16 баллов).** Определить количество локальных максимумов и локальных минимумов функции $f(x) = a_0x^5 + a_1x^4 + a_2x^3 + a_3x^2 + a_4x + a_5$ на отрезке $[-n , n]$.
**Входные данные.** Стандартный входной поток содержит шесть действительных чисел $a_0$, $a_1$, $a_2$, $a_3$, $a_4$, $a_5$ и одно целое число $n$ ($1 \le n \le 100$).
**Выходные данные.** В стандартный выходной поток вывести два целых числа – количество локальных максимумов и количество локальных минимумов.

| Примеры входных данных | Примеры выходных данных |
|---|---|
| 1.0 2.5 -1.0 -3.5 -1.0 0.1 10 | 2 2 |
| 1.0 1.0 1.0 -1.0 -1.0 1.0 2.5 | 1 1 |
| 1.0 -1.0 1.0 -1.0 1.0 -1.0 5 | 0 0 |

**Задача 3 (16 баллов).** Дана строка, в которой записано арифметическое выражение. Выражение может содержать пробелы и круглые скобки, в качестве операндов используются целые константы без знака и идентификаторы, в качестве операторов используются знаки '+', '–', '*' и '/'. Целая константа может состоять из любого количества десятичных цифр. Идентификатор может состоять из любого количества латинских букв в верхнем или нижнем регистре. При этом прописные и строчные буквы не различаются. Подсчитать количество различных целых констант в выражении.
**Входные данные.** Стандартный входной поток содержит одну строку, в которой записано арифметическое выражение. Длина выражения не больше 100.
**Выходные данные.** В стандартный выходной поток вывести одно целое число – количество различных целых констант.

| Примеры входных данных | Примеры выходных данных |
|---|---|
| 1+99999999999999999999999999999 | 2 |
| (2+2*(2+2*(2+2*(2)))) | 1 |
| (n - 1)*n*(n + 1)*(3*n + 2)/24 | 4 |

**Задача 4 (24 балла).** На плоскости даны простой многоугольник (т. е. без самокасаний и самопересечений, но не обязательно выпуклый) и точка. Требуется решить вопрос о принадлежности точки многоугольнику.
**Входные данные.** Стандартный входной поток содержит целое число $N$ ($3 \le N \le 1000$) – количество вершин многоугольника, последовательность из $N$ пар действительных координат вершин многоугольника и действительные координаты точки. Все координаты по модулю не больше $10^6$. Вершины многоугольника заданы в порядке их обхода против часовой стрелки.
**Выходные данные.** В стандартный выходной поток вывести слово YES, если точка принадлежит многоугольнику, и слово NO в противном случае.

| Примеры входных данных | Примеры выходных данных |
|---|---|
| 3 0 0 3 0 0 3 1 1 | YES |
| 4 0 0 4 0 1 1 0 4 1.5 1.5 | NO |

**Задача 5 (32 балла).** Дана целочисленная таблица размера $M$x$N$. Надо пройти из левого верхнего угла таблицы в правый нижний, на каждом шаге перемещаясь на одну клетку вправо или вниз. Найти путь с максимальной суммой чисел в пройденных клетках, а также количество «хороших» путей. «Хорошими» считаются пути, сумма которых отличается от максимальной не более чем на $K$.
**Входные данные.** Первая строка стандартного входного потока содержит три целых числа: $M$ ($2 \le M \le 100$), $N$ ($2 \le N \le 100$) и $K$($0 \le N \le 100$). Следующие $M$ строк содержат по $N$ неотрицательных целых чисел. Значение клетки таблицы не больше 100.
**Выходные данные.** В стандартный выходной поток вывести два целых числа: максимально возможную сумму и количество «хороших» путей.

| Пример входных данных | Пример выходных данных |
|---|---|
| 2 3 3<br>1 9 7<br>2 5 3 | 20 2 |

**Решение задачи 1.**

```
#include "stdafx.h"

int _tmain(int argc, _TCHAR* argv[])
{
	int n = 1;
	double t = 1.0/2.0/3.0, s = t, eps;
	scanf( "%lf", &eps );
	while (t > eps)
	{
		n++;
		t = 1.0/n/(n+1)/(n+2);
		s += t;
	}
	printf( "%lf\n", s );
	return 0;
}
```

**Решение задачи 2.**

```
#include "stdafx.h"

double f( double a[], double x )
{
	return a[0]*x*x*x*x*x + a[1]*x*x*x*x + a[2]*x*x*x + a[3]*x*x + a[4]*x + a[5];
}

int _tmain(int argc, _TCHAR* argv[])
{
	double a[6];
	double n, x, h = 0.001;
	int loc_min = 0, loc_max = 0;
	for (int i = 0; i < 6; i++) scanf( "%lf", &a[i] );
	scanf( "%lf", &n );
	x = -n;
	while (x < n)
	{
		if (f( a, x-h ) < f( a, x ) && f( a, x+h ) < f( a, x )) loc_max++;
		if (f( a, x-h ) > f( a, x ) && f( a, x+h ) > f( a, x )) loc_min++;
		x += h;
	}
	printf( "%d %d\n", loc_max, loc_min );
	return 0;
}
```

**Решение задачи 3.**

```
#include "stdafx.h"
#include <string.h>

int find( char *word, char **arr, int size )
{
	for ( int i = 0; i < size; i++ )
		if ( strcmp( word, arr[i] ) == 0 ) return i;
	return -1;
}

int _tmain(int argc, _TCHAR* argv[])
{
	char line[101];
	char *words[50];
	char seps[] = " \t+-*/()ABCDEFGHIJKLMNOPQRSTUVWXYZ";
	char *copy, *token;
	int index, n = 0;
	for ( int i = 0; i < 50; i++ ) words[i] = NULL;
	gets( line );
	copy = strupr( strdup( line ) );
	token = strtok( copy, seps );
	while ( token != NULL )
	{
		if ( (index = find( token, words, n ) ) == (-1) ) words[n++] = token;
		token = strtok( NULL, seps );
	}
```

```
	printf( "%d\n", n );
	return 0;
}
```

**Решение задачи 4.**

```
#include "stdafx.h"

struct Point {
    double x, y;
};

double min(double a, double b) { return (a < b) ? a : b; }
double max(double a, double b) { return (a > b) ? a : b; }
int Direction(Point pi, Point pj, Point pk)
{
	int Result;
	double Temp = ((pk.x-pi.x)*(pj.y-pi.y)-(pj.x-pi.x)*(pk.y-pi.y));
	if (Temp < 0) Result = -1;
	else if (Temp > 0) Result = 1;
	else Result = 0;
	return Result;
}

bool OnSegment(Point pi, Point pj, Point pk)
{
	if ( ((pk.x >= min(pi.x, pj.x)) && (pk.x <= max(pi.x, pj.x))) &&
		((pk.y >= min(pi.y, pj.y)) && (pk.y <= max(pi.y, pj.y))) ) return true;
	else return false;
}

bool SegmentIntersect(Point p1, Point p2, Point p3, Point p4)
{
	int Result, d1, d2, d3, d4;
	d1 = Direction(p3, p4, p1);
	d2 = Direction(p3, p4, p2);
	d3 = Direction(p1, p2, p3);
	d4 = Direction(p1, p2, p4);
	if ( (((d1 > 0) && (d2 < 0)) || ((d1 < 0) && (d2 > 0))) &&
		(((d3 > 0) && (d4 < 0)) || ((d3 < 0) && (d4 > 0))) ) Result = true;
	else if ( (d1 == 0) && OnSegment(p3, p4, p1) ) Result = true;
	else if ( (d2 == 0) && OnSegment(p3, p4, p2) ) Result = true;
	else if ( (d3 == 0) && OnSegment(p1, p2, p3) ) Result = true;
	else if ( (d4 == 0) && OnSegment(p1, p2, p4) ) Result = true;
	else Result = false;
	return Result;
}

int _tmain(int argc, _TCHAR* argv[])
{
	int N, k = 0;
	Point p, q, a[1000];
	scanf( "%d", &N );
	for (int i = 0; i < N; i++) scanf( "%lf %lf", &a[i].x, &a[i].y );
	scanf( "%lf %lf", &p.x, &p.y );
	q.x = 1000;
	q.y = p.y;
	for (int i = 0; i < N-1; i++)
		if ( SegmentIntersect(p, q, a[i], a[i+1] ) ) k++;
	if ( SegmentIntersect(p, q, a[N-1], a[0] ) ) k++;
	if ( (k % 2) == 0 ) printf( "NO\n" );
	else printf( "YES\n" );
	return 0;
}
```

**Решение задачи 5.**

```
uses
  SysUtils;
const
  MAXM = 100; MAXN = 100; MAXK = 100;
type
  Matrix = array[1..MAXM, 1..MAXN] of integer;

procedure inputData(var V: Matrix; var M, N, K: byte);
```

```
var
  i, j: byte;
begin
  readln(M, N, K);
  for i := 1 to M do
  begin
    for j := 1 to N do
      read(V[i, j]);
    readln;
  end;
end;

procedure run(var E, NW: Matrix; const V: Matrix; const m, n: byte);
var
  i, j: byte;
begin
  for j := 1 to n do NW[1, j] := 1;
  for i := 1 to m do NW[i, 1] := 1;
  E[1, 1] := V[1, 1];
  for j := 2 to n do E[1, j] := E[1, j-1] + V[1, j];
  for i := 2 to m do E[i, 1] := E[i-1, 1] + V[i, 1];
  for i := 2 to m do
    for j := 2 to n do
    begin
      if E[i-1, j] > E[i, j-1] then E[i, j] := E[i-1, j] + V[i, j]
      else E[i, j] := E[i, j-1] + V[i, j];
      if E[i-1, j] > E[i, j-1] then NW[i, j] := NW[i-1, j]
      else if E[i-1, j] < E[i, j-1] then NW[i, j] := NW[i, j-1]
      else NW[i, j] := NW[i-1, j] + NW[i, j-1];
    end;
end;

var
  M, N, K: byte;
  V, E, NW: Matrix;

begin
  inputData(V, M, N, K);
  printMatrix(V, M, N);
  run(E, NW, V, M, N);
  printMatrix(E, M, N);
  printMatrix(NW, M, N);
  writeln(E[M, N]:8, NW[M, N]:8);
  readln;
end.
```