

## Лабораторная работа №2

Ввод и вывод значений простых переменных и одномерных массивов. Создание функциональных тестов

Вариант 16

Выполнил: Татарников Максим группа А-07-22

### 1. Постановка задачи (ПЗ).

**Задание:** Разработать нисходящим способом алгоритм, отделив ввод и вывод от её решения, и написать программу на Delphi, создав консольное приложение для MS Windows, для решения задачи из нижеприведенного перечня задач.

**Условие:** При заданных  $A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_n$  и  $C_1, C_2, \dots, C_n$  для каждой из  $n$  троек вида  $(A_i, B_i, C_i)$  проверить, может ли быть построен треугольник со сторонами  $A_i, B_i, C_i$ , при этом подсчитать число треугольников и сумму их периметров.

### 2. Уточненная постановка задачи

Даны три одномерных вещественных положительных массива  $A, B, C$  из  $n$  ( $0 < n \leq 20$ ) элементов – длины сторон треугольников.

Найти:

$num\_triangle$  – количество треугольников, которые возможно составить из тройки  $A_i, B_i, C_i$ . (Сумма любых двух сторон должна быть больше третьей)

Если  $num\_triangle = n$ , то вывести “Из всех троек можно составить треугольники!”.

Если  $num\_triangle = 0$ , то вывести “Ни из одной тройки нельзя составить ни один треугольник!”

$perimeter$  – Сумма всех значений  $A_i, B_i, C_i$ , из которых возможно составить треугольник.

Если невозможно вычислить периметр ( $num\_triangle=0$ ), то вывести «Вычислить периметр невозможно»

### 3. Пример

Возьмем  $n = 10$ . И длины разных величин (положительные, отрицательные, нулевые):

<b>A</b>	1	3	8	4	2	7	10	7	6	7
<b>B</b>	1	4	5	7	3	8	5	11	2	7
<b>C</b>	1	5	3	3	5	9	6	5	9	1

Из условия только из 4 троек (3,4,5), (1,1,1), (7,8,9), (10,5,6) можно составить треугольник, следовательно:

$num\_triangle = 4$

$perimeter = 3+4+5+1+1+1+7+8+9+10+5+6=60$

### 4. Таблица данных

(начало таблицы данных)

Класс	Имя	Описание (смысл, диапазон, точность)	Тип	Структура	Формат в/в
Входные данные	<b>n</b>	число троек длин, $0 < n \leq 20$	цел	простая переменная	XX (:2)
	<b>A</b>	Первая сторона треугольника, $0 < A_i \leq 20$ , точн. 0.1	вещ	одномерный массив (20)	XX.X (:3:1)
	<b>B</b>	Вторая сторона треугольника, $0 < B_i \leq 20$ , точн. 0.1	вещ	одномерный массив (20)	XX.X (:3:1)
	<b>C</b>	Третья сторона треугольника, $0 < C_i \leq 20$ , точн. 0.1	вещ	одномерный массив (20)	XX.X (:3:1)

(продолжение таблицы данных)

Класс	Имя	Описание (смысл, диапазон, точность)	Тип	Структура	Формат в/в
Выходные данные	<i>num_triangle</i>	Количество возможных треугольников $0 \leq \text{num\_triangle} \leq 20$	цел	простая переменная	XX (:2)
	<i>perimeter</i>	Сумма всех длин троек, которые прошли условие $\text{perimeter} \geq 0$	вещ	простая переменная	XXX.X (:4:1)
Промежу- точные данные*	<i>i</i>	индекс текущего элемента, $0 \leq i \leq 21$	цел	простая переменная	---
					---
					---
					---

**5. Входная форма:**

1.1 Количество троек n:  
 Обр1  
 1.2 <n>  
 \_\_\_\_\_  
 Обр2.1 Длины сторон  
 треугольника:  
 \_\_\_\_\_  
 Обр2.2 <A[1]> <B[1]> <C[1]>  
 <A[2]> <B[2]> <C[2]>  
 .....  
 <A[n]> <B[n]> <C[n]>

**6. Выходная форма:**

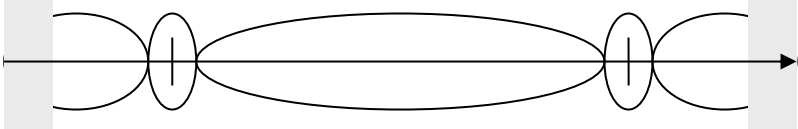
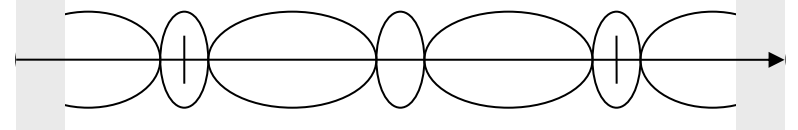
Обр3 Лаб.2  
 \_\_\_\_\_  
 Обр4 Количество троек n = <n>  
 \_\_\_\_\_  
 Обр5 Длины сторон  
 треугольника:  
 \_\_\_\_\_  
 Обр6 <A[1]> <B[1]> <C[1]>  
 <A[2]> <B[2]> <C[2]>  
 .....  
 <A[n]> <B[n]> <C[n]>  
 \_\_\_\_\_  
 Обр7 Количество треугольников =  
 <num\_triangle>  
 Обр8 Сумма периметров = <perimeter>  
 \_\_\_\_\_  
 Обр9 Из всех троек можно составить  
 треугольник (num\_triangle = n)  
 Обр10 Сумма периметров = <perimeter>  
 \_\_\_\_\_  
 Обр11 Ни из одной тройки нельзя  
 составить ни одного  
 треугольника  
 Обр12 Периметр = 0

## 7. Аномалии

№	Описание	Условие возникновения**	Реакция на аномалию
1	$n$ меньше минимально допустимого значения	$n < 1$	Сообщение: «Некорректное $n$ : $n < 1$ » (обр.1.2) Действие: Завершение работы программы
2	$A_i, B_i, C_i$ меньше минимально допустимого значения	$A_i$ или $B_i$ или $C_i < 0$	Сообщение: «Некорректная длина треугольника: $A, B, C < 0$ » (обр.2.2) Действие: Завершение работы программы

## 8. Функциональные тесты

```
num_triangle as n_t
perimeter as p
```

Исходные данные								Результаты		Тест №		
	аном	граница а	Средние значения			граница	аном	n_t	макс = 20	2		
									мин = 1 и не ноль	3		
n	<1	1	[2 , 19 ]			20	>20					
Тест №	---	3	1			2	---		сред = (2,19)	1		
									0	4		
A[i]	<-20	-20	(-20,0)		0	(0.1,20)			20	>20	не суц = не возможно	---
Тест №	---	2	3	1	1	2	---		Макс.выч.нагрузк а = 20 точек из 20 попали	2		
B[i]	<-20	-20	(-20,0)		0	(0.1,20)			20	>20		
Тест №	---	2	3	1	1	2	---		p	макс = 1200	2	
										мин = 0.3	3	
C[i]	<-20	-20	(-20,0)		0	(0.1,20)		20		>20		
Тест №	---	2	3	1	1	2	---	сред = (0.3, 1200)		1		
								0		...		
								не суц, когда n_t=0	4			
	Аном 3	аном3	Аном3	Аном 2	сред	граница	Аном1		Макс.выч.нагрузк а = 1200	2		

№ теста	Входные данные	Ожидаемый результат	Смысл теста																																	
1	<div><div><div>n=10</div><table><tr><td>A</td><td>1</td><td>3</td><td>8</td><td>4</td><td>2</td><td>7</td><td>10</td><td>7</td><td>6</td><td>7</td></tr><tr><td>B</td><td>1</td><td>4</td><td>5</td><td>7</td><td>3</td><td>8</td><td>5</td><td>11</td><td>2</td><td>7</td></tr><tr><td>C</td><td>1</td><td>5</td><td>3</td><td>3</td><td>5</td><td>9</td><td>6</td><td>5</td><td>9</td><td>1</td></tr></table></div></div>	A	1	3	8	4	2	7	10	7	6	7	B	1	4	5	7	3	8	5	11	2	7	C	1	5	3	3	5	9	6	5	9	1	<div><div>N_t = 7</div><div>P = 115</div></div>	Нормально-средние значение для общей проверки программы
A	1	3	8	4	2	7	10	7	6	7																										
B	1	4	5	7	3	8	5	11	2	7																										
C	1	5	3	3	5	9	6	5	9	1																										
2	<div><div><div>n=20</div><table><tr><td>A</td><td>20</td><td>20</td><td>...</td><td>20</td></tr><tr><td>B</td><td>20</td><td>20</td><td>...</td><td>20</td></tr><tr><td>C</td><td>20</td><td>20</td><td>...</td><td>20</td></tr></table></div></div>	A	20	20	...	20	B	20	20	...	20	C	20	20	...	20	<div><div>N_t = 20</div><div>P = 1200</div></div>	<div><div>Максимальное</div><div>значение всех исходных данных и результатов.</div><div>Максимальная вычислительная нагрузка для n_t и p.</div></div>																		
A	20	20	...	20																																
B	20	20	...	20																																
C	20	20	...	20																																
3	<div><div><div>n=1</div><table><tr><td>A</td><td>0.1</td></tr><tr><td>B</td><td>0.1</td></tr><tr><td>C</td><td>0.1</td></tr></table></div></div>	A	0.1	B	0.1	C	0.1	<div><div>N_t = 1</div><div>P = 0,3</div></div>	<div><div>Минимальные</div><div>значения n.</div><div>Минимальные значения A,B,C,n_t,p</div></div>																											
A	0.1																																			
B	0.1																																			
C	0.1																																			
4	<div><div>P – не сущ</div><div>Num_triangle = 0</div></div>	<div><div>Сообщение по обр. A1</div></div>	<div><div>Аномалия №1</div><div>...</div></div>																																	

Результаты							Тест №																		
N_t	макс =20						2																		
	мин и не 0: 1						3																		
	сред = (0,20)						1																		
	не сущ = невозможно						---																		
	0						4																		
	Макс.выч.нагрузка = 20						2																		
p	макс = 1200						2																		
	мин и не 0: n =20						3																		
	<table border="1"><tr><td>A</td><td>0,1</td><td>0</td><td>...</td><td>0</td><td>0</td></tr><tr><td>B</td><td>0.1</td><td>0</td><td>...</td><td>0</td><td>0</td></tr><tr><td>C</td><td>0.1</td><td>0</td><td>...</td><td>0</td><td>0</td></tr></table>							A	0,1	0	...	0	0	B	0.1	0	...	0	0	C	0.1	0	...	0	0
	A	0,1	0	...	0	0																			
	B	0.1	0	...	0	0																			
	C	0.1	0	...	0	0																			
	P = 0.1+0.1+0.1 = 0.3																								
	сред = (0.3, 1200)						1																		
не сущ, когда num_triangle = 0						4																			
0, не сущ						...																			
Макс.выч.нагрузка = 1200						2																			

## 9. Блок-схема



## 10 Программа

```

program Lab2;

const
  Nmax = 20; {верхняя граница индексов массива – максимальное количество точек}

  {раздел описания переменных:}
var
  n, i, num_triangle: integer;
  perimeter: real;
  A, B, C: array [0..Nmax] of real;

  {раздел операторов:}
begin

  //вывод заголовка в выходной документ:
  writeln('Лаб.2':40); {вывод с переходом на следующую строку}
  {ввод исходных данных: }
  writeln('Количество троек n (0<n<=20):'); readln(n); {обр 1.1,1.2}
  {ввод массивов A, B, C:}
  if (n <= 0) or (n > 20) then
  begin
    writeln('Ошибка задания условия');
    exit;
  end;

  writeln('Стороны треугольника:'); {обр2.1}
  i := 0;
  while i<n do
  begin
    readln(A[i], B[i], C[i]);
    if (A[i]<=0) or (B[i]<=0) or (C[i]<=0) or (A[i]>20) or (B[i]>20) or (C[i]>20)
  then
  begin
    writeln('Ошибка задания условия');
    exit;
  end;
    {обр2.2:ввод 3 элементов через пробел и переход на след. строку}
    i := i+1;
  end;

  {вывод входных данных в выходной документ для подтверждения: }
  for i := 1 to 80 do write('='); { отделим визуально чертой и строкой введенные
и выводимые значения }
  writeln;

  writeln('Количество троек =', n:3); {обр4}

  {вывод исходных массивов A, B, C}
  writeln('Стороны треугольника: ');
  i := 0;
  while i < n do {обр4,5}

```

```

begin
    writeln(' ':5, A[i]:3:1, ' ':8, B[i]:3:1, ' ':8, C[i]:3:1);
    i := i + 1;
end;
num_triangle := 0;
perimeter := 0;
i := 0;
while i < n do // перебираем по-очереди все n точек
begin
    if ((B[i] + C[i]) - A[i] > 0.01) and ((A[i] + C[i]) - B[i] > 0.01) and ((B[i]
+ A[i]) - C[i] > 0.01) then // Проверка усл на неравенство треугольника
    begin
        num_triangle := num_triangle + 1; // и количество увеличиваем
        perimeter := perimeter + A[i] + B[i] + C[i];
    end;
    i := i + 1;
end;
if num_triangle = n then
begin
    writeln('Из всех троек можно составить треугольник!':45);
    writeln('    Периметр = ', perimeter:3);
end
else if num_triangle = 0 then
begin
    writeln('Ни из одной тройки нельзя составить треугольник!');
    writeln('Вычислить периметр невозможно');
end
else
begin
    writeln('    Кол-во треугольников = ', num_triangle:2); {обр.7}
    writeln('    Периметр = ', perimeter:3); {обр.8}
end;
write('Press Enter...'); readln; {задерживаем экран до нажатия ENTER}
end.

```