**Лабораторная работа № 3**

**Тема:** Определение и вызов функций. Передача параметров и возврат результатов.

**Цель работы:** изучить правила определения и вызова функций в языке C#, получить практические навыки разработки спецификаций функций по обработке простых типов данных и функций по обработке структурированных типов данных, и создания программ, использующих функции программиста.

**Задание 1.** Разработать подпрограммы для вычисления площади полной поверхности и объёма конуса, если известны его высота и радиус основания.

**Математическая модель**

*Входные данные:*

*Rad (int)* – радиус конуса, вводится с клавиатуры(должен быть больше нуля)*.*

*Heig (int) –* высота конуса, вводится с клавиатуры(должен быть больше нуля)*..*

*Выходные данные:*

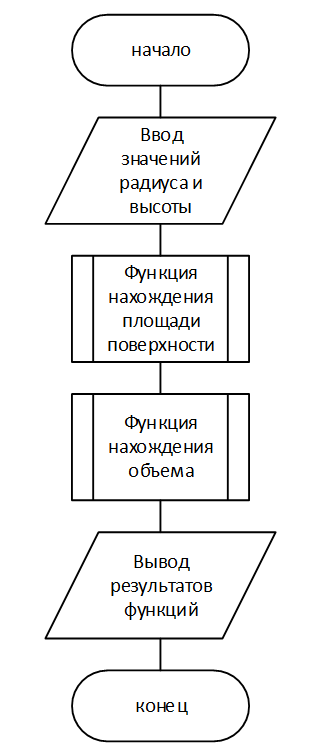
- *surfaceArea(double)* – функция нахождения площади поверхности конуса.

- *Volume(double)* – функция нахождения объема конуса.

*Связь:*

1. Создадим подпрограмму (функцию) для нахождения площади поверхности конуса, зададим аргументы радиуса и высоты, вычислим по формуле;
2. Создадим подпрограмму (функцию) для нахождения объема конуса, зададим аргументы радиуса и высоты, вычислим по формуле.

**Блок-схема**

****



**Код**

using System;

public class Cone

{

public static void Main(string[] args)

{

// Ввод данных

Console.Write("Введите радиус основания конуса: ");

double rad = double.Parse(Console.ReadLine());

Console.Write("Введите высоту конуса: ");

double heig = double.Parse(Console.ReadLine());

// Вычисление площади поверхности и объема

double surfaceArea = SurfaceArea(rad, heig);

double volume = Volume(rad, heig);

// Вывод результатов

Console.WriteLine($"Площадь полной поверхности конуса: {surfaceArea:F2}");

Console.WriteLine($"Объем конуса: {volume:F2}");

}

// Подпрограмма для вычисления площади полной поверхности конуса

static double SurfaceArea(double rad, double heig)

{

// Проверка на положительность радиуса и высоты

if (rad <= 0 || heig <= 0)

{

throw new ArgumentException("Радиус и высота конуса должны быть положительными числами.");

}

double slantHeight = Math.Sqrt(rad \* rad + heig \* heig);

double baseArea = Math.PI \* rad \* rad;

double lateralArea = Math.PI \* rad \* slantHeight;

return baseArea + lateralArea;

}

// Подпрограмма для вычисления объема конуса

static double Volume(double rad, double heig)

{

if (rad <= 0 || heig <= 0)

{

throw new ArgumentException("Радиус и высота конуса должны быть положительными числами.");

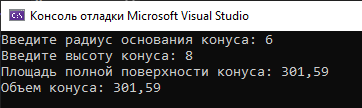
}

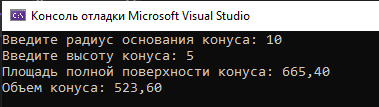
return (1.0 / 3.0) \* Math.PI \* rad \* rad \* heig;

}

}

**Результат**





* **Задание 2.** Заданы два одномерных массива разной размерности. Организовать ввод и вывод на экран массивов с использованием функций программиста;
* Найти сумму нечетных по значению элементов каждого массива, используя функцию программиста;
* В том массиве, где эта сумма больше всего, поменять местами максимальный и минимальный элемент массива. Обмен осуществить при помощи функции.

**Математическая модель**

*Входные данные:*

*Array1(int)* – массив целых чисел, генерируется случайным образом

*Array2(int) –* массив целых чисел, генерируется случайным образом

*Выходные данные:*

- *Array(int)* – массив со сменой позиций минимального и максимального элемента.

*Связь:*

1. Создадим подпрограмму для генерации массивов
2. Создадим подпрограмму для вывода массивов
3. Создадим подпрограмму для нахождения суммы нечетных элементов массива
4. Создадим подпрограмму для обмена максимального и минимального элементов массива

**Блок-схема**





**Код**

using System;

public class Program

{

private static Random random = new Random(); // Объявление случайного генератора как поля класса

public static void Main(string[] args)

{

// Создание случайных массивов

int[] array1 = GenerateArray(5); // Размер первого массива

int[] array2 = GenerateArray(8); // Размер второго массива

// Вывод массивов на экран

Console.WriteLine("Первый массив:");

PrintArray(array1);

Console.WriteLine("Второй массив:");

PrintArray(array2);

// Нахождение суммы нечетных элементов в каждом массиве

int sumOdd1 = SumOddElements(array1);

int sumOdd2 = SumOddElements(array2);

Console.WriteLine($"Сумма нечетных элементов в первом массиве: {sumOdd1}");

Console.WriteLine($"Сумма нечетных элементов во втором массиве: {sumOdd2}");

// Обмен максимального и минимального элементов в массиве с большей суммой нечетных элементов

if (sumOdd1 > sumOdd2)

{

SwapMinMax(ref array1);

Console.WriteLine("В первом массиве поменяны местами максимальный и минимальный элементы:");

PrintArray(array1);

}

else if (sumOdd2 > sumOdd1)

{

SwapMinMax(ref array2);

Console.WriteLine("Во втором массиве поменяны местами максимальный и минимальный элементы:");

PrintArray(array2);

}

else

{

Console.WriteLine("Суммы нечетных элементов в массивах равны.");

}

}

// Функция генерации случайного массива заданного размера

static int[] GenerateArray(int size)

{

if (size <= 0)

{

throw new ArgumentException("Размер массива должен быть больше нуля.");

}

int[] array = new int[size];

for (int i = 0; i < size; i++)

{

array[i] = random.Next(0, 101); // Случайные числа от 0 до 100

}

return array;

}

// Функция вывода массива на экран

static void PrintArray(int[] array)

{

Console.Write($"{element} ");

}

// Функция нахождения суммы нечетных элементов массива

static int SumOddElements(int[] array)

{

int sum = 0;

foreach (int element in array)

{

if (element % 2 != 0)

{

sum += element;

}

}

return sum;

}

// Функция обмена максимального и минимального элементов массива

static void SwapMinMax(ref int[] array)

{

int maxI = 0;

int minI = 0;

for (int i = 1; i < array.Length; i++)

{

if (array[i] > array[maxI])

{

maxI = i;

}

if (array[i] < array[minI])

{

minI = i;

}

}

int temp = array[maxI];

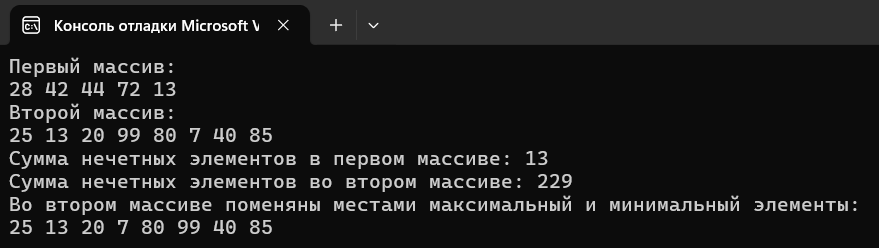
array[maxI] = array[minI];

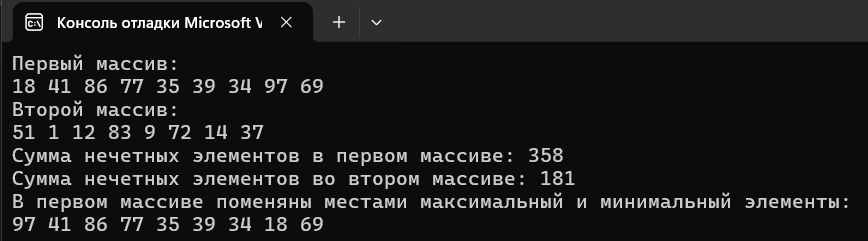
array[minI] = temp;

}

}

**Результат**

****

****