Лабораторная работа №3:
Функции

Цель:

Целью данной работы является получение опыта работы с функциями на языке высокого уровня C#, в среде программирования Microsoft Visual Studio

Задание:

Разработать и реализовать следующие функции:

1. Функцию **randomInt**, возвращающую случайно сгенерированное целое число в заданном диапазоне.

Прототип функции: int **randomInt**(int min, int max);

2. Функцию **randomFloat**, возвращающую случайно сгенерированное вещественное число в заданном диапазоне.

Прототип функции: double **randomFloat**(double min, double max);

3. Функцию **rounding**, возвращающую целое число, полученное из вещественного по правилам округления.

Прототип функции: int **rounding**(double number);

4. Функцию **getPercent**, возвращающую заданный процент от заданного числа.

Прототип функции: double **getPercent**(double number, double percent);

5. Рекурсивную функцию **drawPyramid**, выводящую в консоль пирамиду из символов ^, с основанием равным заданному числу

Прототип функции: void **drawPyramid**(int i, int N); Пример для основания 5:

```
    ^
   ^^^
  ^^^^^
```

Разработать и реализовать следующие функции для работы с одномерными массивами:

1. Заполнение массива случайно сгенерированными числами в заданном диапазоне.
2. Заполнение массива с клавиатуры.
3. Копирование значений из массива указанного в качестве первого параметра функции, в массив, указанный в виде второго параметра функции.
4. Вывод значений массива, передаваемого в функцию, на экран.

Реализовать версии функций для массивов целых чисел, вещественных чисел и двумерных массивов целых и вещественных чисел.

Все функции должны иметь **проверку на корректность передаваемых данных** и выдавать сообщение об ошибке, в случае, если передаются не корректные данные. (Пример: на преобразование строки в число, передаётся строка, содержащая буквы)

Справочная информация:

Сигнатуры функций могут выглядеть следующим образом (следует помнить, что в данных случаях передаются копии значений параметров):

```
//функция, не имеющая возвращаемого значения и имеющая один параметр типа int
void proc(int param)

//функция, не имеющая параметров и возвращающая значение типа int
int foo()

//функция, имеющая 2 параметра и возвращающая значение типа float
double randomFloat(double min, double max)
```

Простая программа, содержащая функцию вычисления степени числа:

```
internal class Program                      //класс программы
{

    Ссылок: 1
    static int power(int n, int p)          //функция, возводящая число n в степень p
    {
        int res = n;

        for (int i = 1; i < p; i++) res *= n;

        return res;                         //возврат значения
    }

    Ссылок: 0
    static void Main(string[] args)         //главная функция
    {
        int a = 2, b = 4;
        int res = power(a, b);              //возведение 2 в 4ю степень

        Console.WriteLine("{0}^{1} = {2}", a, b, res);

        Console.ReadKey();
    }
}
```

Пример программы, в которой в качестве параметров функции передаются адреса переменных:

```
internal class Program                      //класс программы
{

    Ссылок: 1
    static void swap(ref int a, ref int b) //функция обмена значениями
    {
        int c;
        c = b; b = a; a = c;
    }

    Ссылок: 0
    static void Main(string[] args)         //главная функция
    {
        int a = 2, b = 4;
        swap(ref a, ref b);                 //возведение 2 в 4ю степень

        Console.WriteLine("a = {0}, b = {2}", a, b);

        Console.ReadKey();
    }
}
```

В языке C#, массивы являются ссылочными переменными, поэтому при передаче массива, в качестве параметра функции, изменения происходящие внутри функции с массивом, будут иметь влияние на этот массив во всей программе.

Рекурсивной, называют функцию вызывающую саму себя. Пример программы, содержащей рекурсивную функцию, с трассировкой шагов выполнения:

```
internal class Program                          //класс программы
{

    Ссылок: 2
    static void rec(int i)                      //рекурсивная функция
    {
        Console.Write("{0} ", i);               //печать значения i на прямом ходу

        if (i == 0) Console.WriteLine();
        if (i > 0) rec(i - 1);

        Console.Write("{0} ", i);               //печать значения i на обратном ходу
    }

    Ссылок: 0
    static void Main(string[] args)             //главная функция
    {
        rec(4);                                 //возведение 2 в 4ю степень

        Console.ReadKey();
    }
}
```

Результат:
> 3 2 1 0
> 0 1 2 3

Пример программы, использующей стандартную функцию генерации псевдослучайных чисел:

```
internal class Program                          //класс программы
{
    Ссылок: 0
    static void Main(string[] args)             //главная функция
    {
        Random rnd = new Random();              //создание генератора псевдослучайных последовательностей чисел

        int n = rnd.Next(0, 5);                 //получение случайного числа в диапазоне от 0 до 5 (0, 1, 2, 3 или 4)
        double fn = rnd.NextDouble();           //получение случайного вещественного числа в диапазоне от 0 до 1

        Console.WriteLine("n = {0}, fn = {}", n, fn);

        Console.ReadKey();
    }
}
```