# Лабораторная работа №4

**Тема:** Разработка программ с использованием методов.

**Цель:** Научится разработке и перегрузке методов с параметрами различного статуса, использованию встроенных методов.

**Технологическое оснащение:** Ноутбук DELL, Windows 10, MS Word 2016, MS Visual Studio 2019, методическое указание к работе.

**Выполнение работы**

**Вариант 18**

## Пример 1:

Знакомство со статическими методами.

Код программы:

|  |
| --- |
| using System;  class StatMethDemo  {  // Статический метод для отображения текста,  // переданного аргументом методу:  static void show(string txt)  {  Console.WriteLine(txt);  }  // Статический метод для вычисления факториала числа,  // переданного аргументом методу:  static int factorial(int n)  { // Локальная переменная:  int s = 1;  // Вычисление произведения:  for (int k = 1; k <= n; k++)  {  // Умножение произведения на число:  s \*= k;  }  // Результат метода:  return s;  }  // Статический метод для возведения числа в степень.  // Число и степень передаются аргументами методу:  static double power(double x, int n)  {  // Локальная переменная:  double s=1;  // Вычисление результата (число в степени):  for (int k = 1; k <= n; k++)  {  // Текущее значение умножается на число:  s \*= x;  }  // Результат метода:  return s;  }  // Главный метод программы:  static void Main()  {  // Вызываем статический метод для отображения  // сообщения в консольном окне:  show("Начинаем вычисления:");  int m = 5; // Целочисленные переменные  double z=3, num; // Действительные переменные  // Вычисление факториала числа:  show(m + "!=" + factorial(m));  // Число в степени:  num = power(z, m);  // Отображение сообщения вызовом статического метода:  show(z+" в степени "+m+": "+num);  Console.ReadKey(true);  }  } |

Результат выполнения программы представлен на рисунке 1.

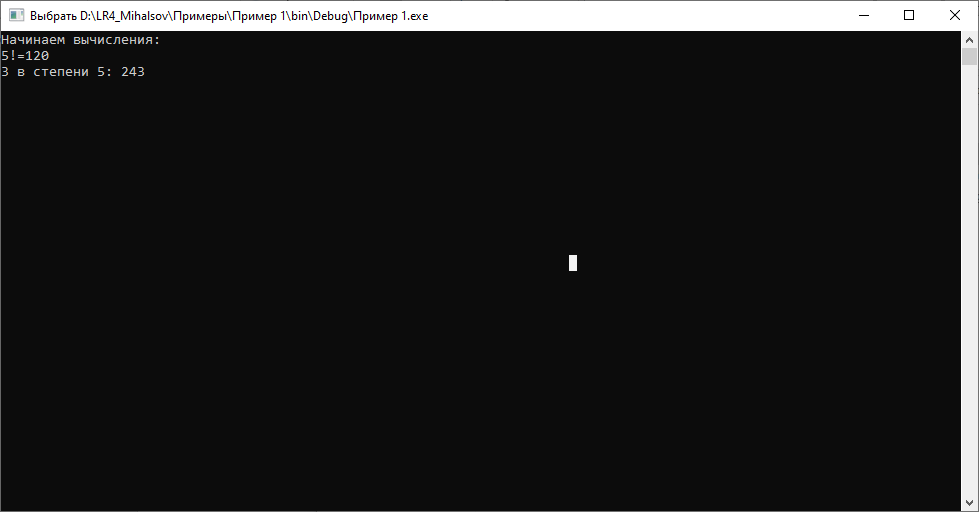


Рисунок 1 – Результат выполнения программы

## Пример 2:

Даны три действительных числа a, b и c. Определить: может ли существовать треугольник с такими длинами сторон.

Код программы:

|  |
| --- |
| using System;  namespace Prim\_Metod2  {  class Program  {  public static void Main(string[] args)  {  double a = 3, b = 4, c = 5;  Console.WriteLine("Длины сторон:\n a={0} b={1} c={2}", a, b, c); if (Treug(a, b, c))  Console.WriteLine("Да, треугольник существует");  else  Console.WriteLine("Нет, треугольник не существует");  Console.ReadKey(true);  }  public static bool Treug(double a, double b, double c)  {  if (a + b > c && a + c > b && b + c > a) return true; else return false;  }  }  } |

Результат выполнения работы представлен на рисунке 2.



Рисунок 2 – Результат выполнения программы

## Пример 3:

Написать метод, который позволял бы произвести обмен значений двух переменных.

Код программы:

|  |
| --- |
| using System;  namespace Prim\_Metod3  {  class Program  {  public static void Main(string[] args)  {  int a = 3, b = 4;  Console.WriteLine("До входа в метод:\n a={0} b={1}", a, b);  Obmen(ref a, ref b);  Console.WriteLine("После выхода из метода:\n a={0} b={1}", a, b);  Console.ReadKey(true);  }  public static void Obmen(ref int a, ref int b)  {  Console.WriteLine("В методе до обмена:\n a={0} b={1}", a, b); int c; c = a; a = b; b = c;  Console.WriteLine("В методе после обмена:\n a={0} b={1}", a, b);  }  }  } |

Результат выполнения программы представлен на рисунке 3.

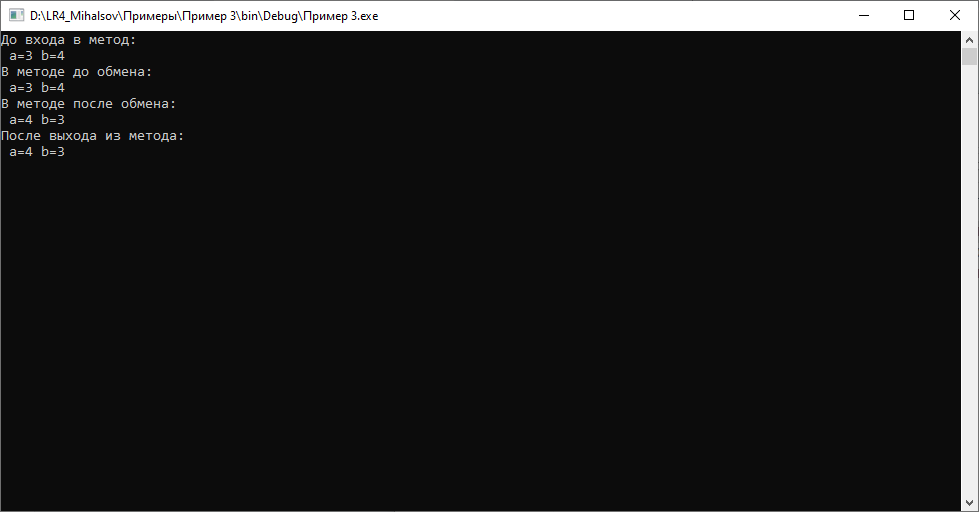


Рисунок 3 – Результат выполнения программы

## Пример 4:

Даны три числа. Вычислить их средние величины: среднее арифметическое, среднее геометрическое, среднее квадратическое.

Код программы:

|  |
| --- |
| using System;  namespace Prim\_Metod4  {  class Program  {  public static void Main(string[] args)  {  double a, b, c, ar, geo, kv; Console.Write("a=");  a = double.Parse(Console.ReadLine()); Console.Write("b="); b = double.Parse(Console.ReadLine()); Console.Write("c=");  c = double.Parse(Console.ReadLine());  Srednee(a, b, c, out ar, out geo, out kv);  Console.WriteLine("Среднее арифметическое = {0}", ar);  Console.WriteLine("Среднее геометрическое = {0}", geo);  Console.WriteLine("Среднее квадратическое = {0}", kv);  Console.ReadKey(true);  }  public static void Srednee(double a, double b, double c, out double ar, out double geo, out double kv)  {  ar = (a + b + c) / 3; geo = Math.Pow(a \* b \* c, 1.0 / 3); kv = Math.Sqrt(a \* a + b \* b + c \* c);  }  }  } |

Результат выполнения программы представлен на рисунке 4.

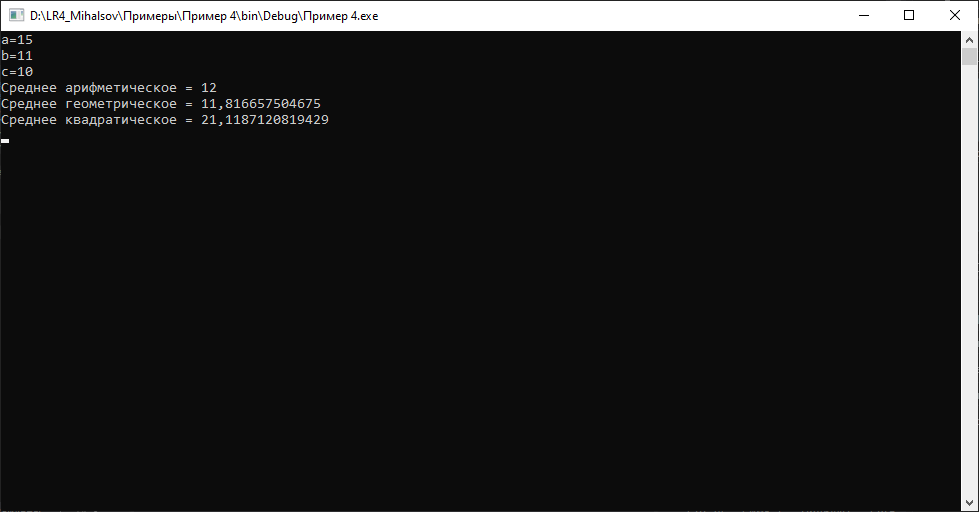


Рисунок 4 – Результат выполнения программы

## Пример 5:

Написать программу для вычисления суммы целых чисел. Суммирование оформим в виде метода. Обеспечим произвольное количество чисел для суммирования.

Код программы:

|  |
| --- |
| using System;  namespace Prim\_Metod5  {  class Program  {  public static void Main(string[] args)  {  int s; s = fSum(4, 3, 12);  Console.WriteLine("Сумма = " + s);  Console.ReadKey(true);  }  public static int fSum(params int[] x)  {  int s = 0; for (int i = 0; i < x.Length; i++) s += x[i]; return s;  }  }  } |

Результат выполнения программы представлен на рисунке 5.

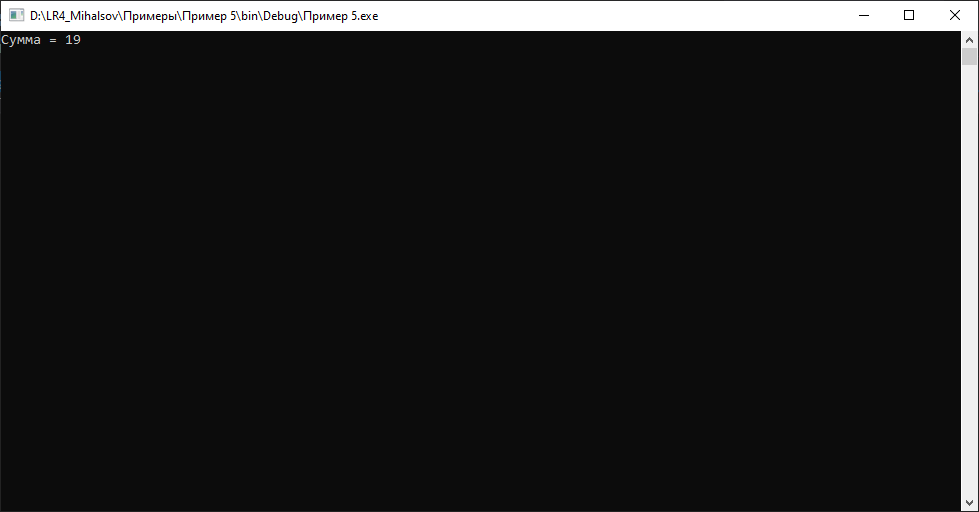


Рисунок 5 – Результат выполнения программы

## Пример 6:

Перегрузка статических методов.

Код программы:

|  |
| --- |
| using System;  class OverloadMethDemo  {  static void show(string txt)  {  Console.WriteLine("Текст: " + txt);  }  static void show(int num)  {  Console.WriteLine("Целое число: " + num);  }  static void show(double num) {  Console.WriteLine("Действительное число: "+num);  }  static void show(char s)  {  Console.WriteLine("Символ: " + s);  }  static void show(int num, char s)  {  Console.WriteLine("Аргументы {0} и {1}", num, s);  }  static void Main()  {  // Целочисленная переменная:  int num=5;  // Действительная числовая переменная:  double z=12.5;  // Символьная переменная:  char symb = 'W';  // Вызываем метод с символьным аргументом:  show(symb);  // Вызываем метод с текстовым аргументом:  show("Знакомимся с перегрузкой методов");  show(num);  // Вызываем метод с действительным аргументом:  show(z);  // Вызываем метод с двумя аргументами:  show(num,'Q');  Console.ReadKey(true);  }  } |

Результат выполнения программы представлен на рисунке 6.

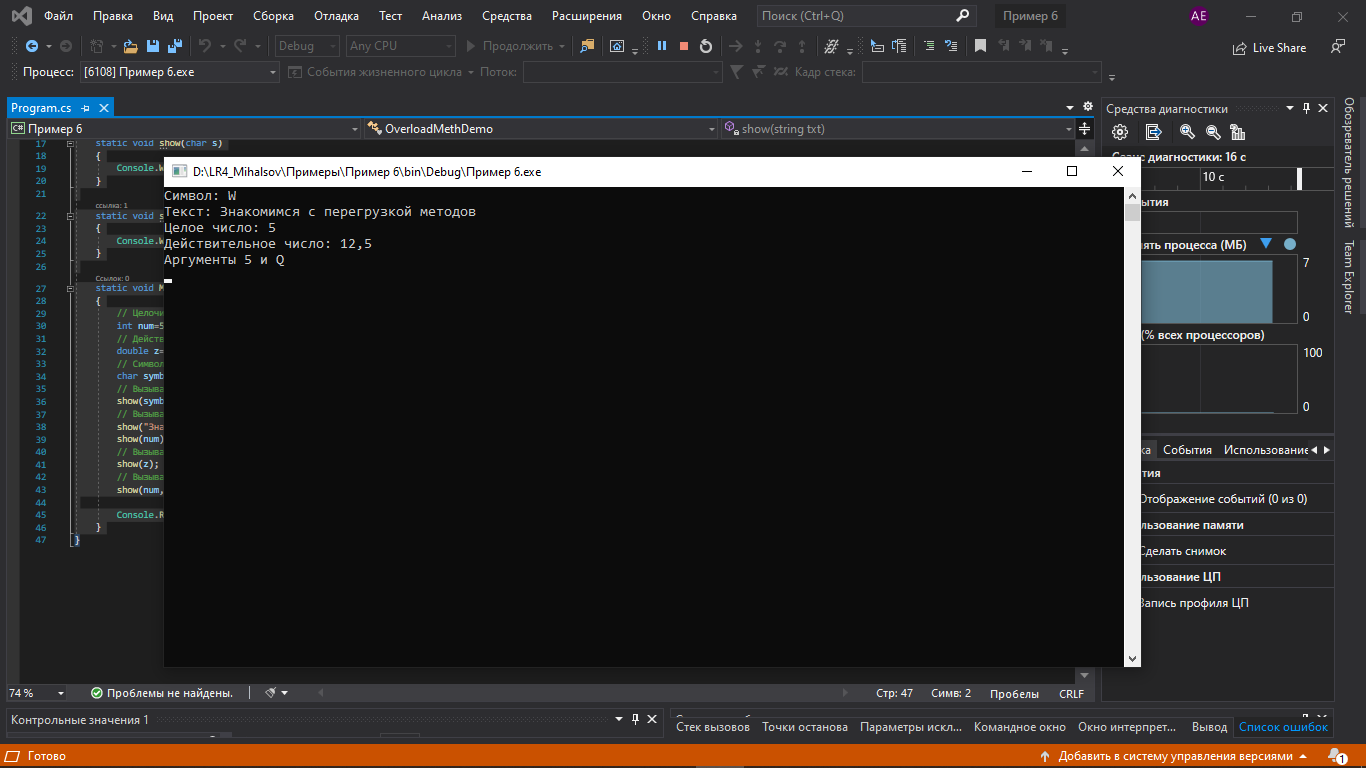


Рисунок 6 – Результат выполнения программы

## Задание 1:

Составить графическую схему алгоритма и программу для вычисления скорости в конце пути и путь, пройденный телом за время t с ускорением a и начальной скорости v0. Вычисление значений переменных оформить в виде метода, в который передать исходные данные в виде входных параметров. Исходные данные для отладки программы подобрать самостоятельно.

Блок схема программы (рисунок 7)



Рисунок 7 – Блок схема программы

Код программы:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace z1  {  class Program  {  static void SpeedWay(float t, float a, float v0, out double v, out double S)  {  v = v0 + (a \* t);  S = Math.Pow(v0, 2) + (a \* Math.Pow(t, 2) / 2);  }  static void Main(string[] args)  {  // v - конечная скорость  // S - пройденный путь  double v, S;  // Ввод значений с клавиатуры  Console.Write("Введите время пути (t): "); float t = float.Parse(Console.ReadLine());  Console.Write("Введите ускорение (a): "); float a = float.Parse(Console.ReadLine());  Console.Write("Введите начальную скорость (v0): "); float v0 = float.Parse(Console.ReadLine());  // Вызов функции расчёта  SpeedWay(t, a, v0, out v, out S);  // Вывод значений  Console.WriteLine("Конечная скорость (v): {0}", v);  Console.WriteLine("Пройденное растояние (S): {0}", S);  Console.ReadKey(true);  }  }  } |

Результат выполнения программы представлен на рисунке 8.

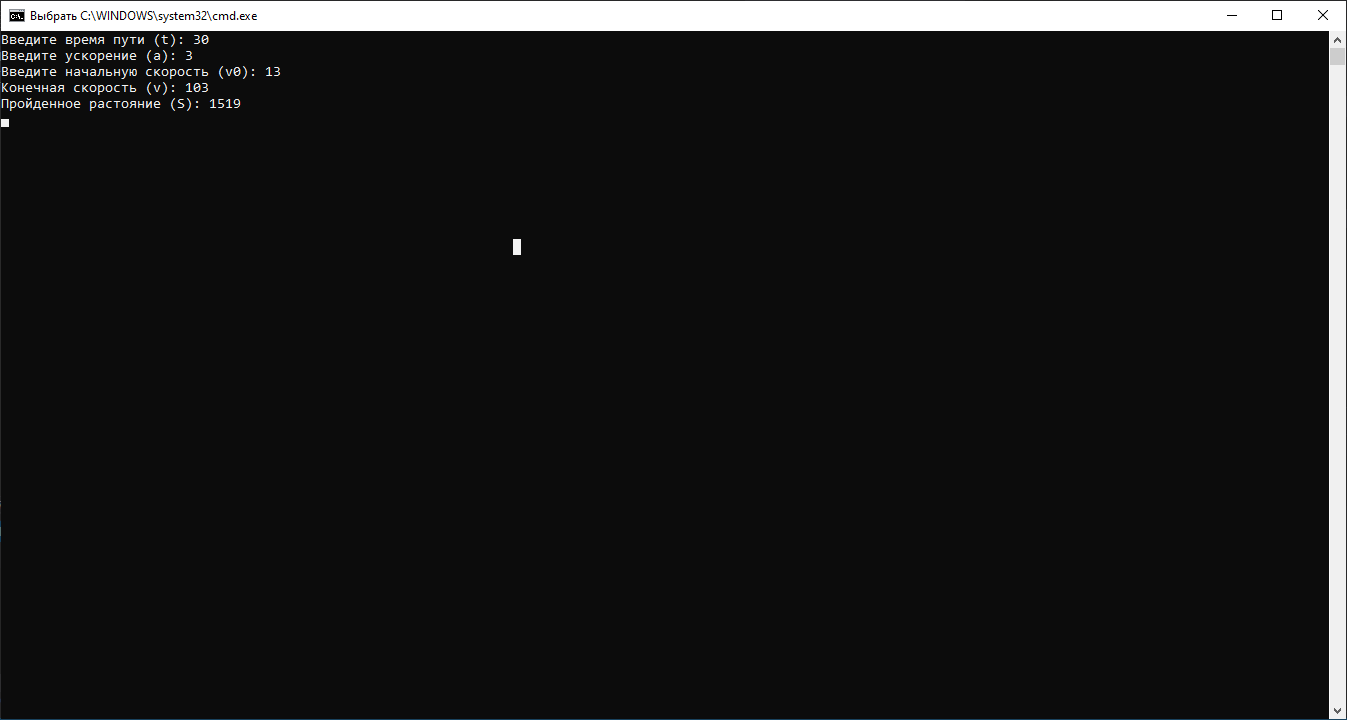


Рисунок 8 – Результат выполнения программы

## Задание 2:

Написать методы для вычисления приведенных функций.

Разработать алгоритм вычисления, выбранного пользователем метода. Выбор функции осуществлять с помощью меню простого выбора. Написать и отладить программу, реализующую этот алгоритм, с использованием созданных методов. Исходные данные для отладки программы подобрать самостоятельно. Подготовить полный набор тестов для отладки разработанных программ.

Блок схема программы (рисунок 9):



Рисунок 9 – Блок схема программы

Код программы:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace z2  {  class Program  {  static double f1(float x)  {  return Math.Pow(Math.Sin(x),4);  }  static double f2(float x)  {  return Math.Pow(x, 3 - 0.2 \* x);  }  static double f3(float x)  {  return Math.Pow(x,2) - 6 \* x + 18;  }  static void Main(string[] args)  {  // y - переменная для сохранеия подсчитанного значения  double y = 0;  // n - переменная для сохранеия выбранной пользователем функции  int n;  // Ввходим x  Console.Write("Введите x : "); float x = float.Parse(Console.ReadLine());  // Пока пользователь не введет значения в нужном диапозоне (0-3), начинать цикл заново  do  {  Console.Write("Введите номер функции для расчета (1 <= n <= 3) : ");  n = int.Parse(Console.ReadLine());  } while (!(n >= 1 && n <= 3));  // В соотведствии с выбранной функцией производим расчёт  switch (n)  {  case 1:  y = f1(x);  break;  case 2:  y = f2(x);  break;  case 3:  y = f3(x);  break;  }  // Вывод полученного значения  Console.WriteLine("y : {0}", y);  Console.ReadKey(true);  } }} |

Тестирование программы

Проверка по первому условию (рисунок 10)

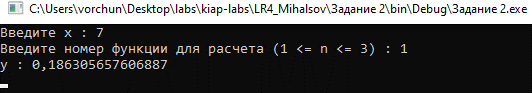


Рисунок 10 – Проверка первого условия

Проверка по второму условию (рисунок 11)

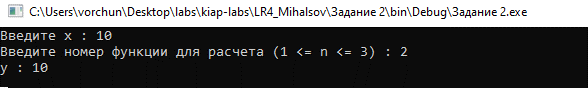


Рисунок 11 – Проверка по второму условию

Проверка по третьему условию (рисунок 12)

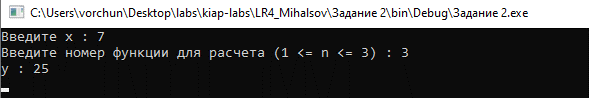


Рисунок 12 – Проверка по третьему условию

## Вывод:

В результате выполнения лабораторной работы были получены навыки по разработки программ с использованием статических методов, их перегрузке и передачи в них параметрами с различного статуса.