**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ**

***Институт Принтмедиа и информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № \_10\_**

**Дисциплина:** Введение в программирование

**Выполнил(а): студент(ка) группы \_191-726\_\_**

\_\_\_\_\_\_\_\_Щека С. А.\_\_\_\_\_\_\_\_\_\_

(Фамилия И.О.)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: \_Асс. Кононенко К.М.*\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2019**

Оглавление

[**Теория** 3](#_Toc20673720)

[**Задания** 4](#_Toc20673721)

[**Блок-схемы** 5](#_Toc20673722)

[**Коды программ** 7](#_Toc20673723)

[Листинг 1 7](#_Toc20673724)

[Листинг 2 8](#_Toc20673725)

[Листинг 3 9](#_Toc20673726)

[Листинг 4 10](#_Toc20673727)

[Листинг 5 11](#_Toc20673728)

[**Результаты выполнения програм** 12](#_Toc20673729)

# **Теория**

String Class Представляет текст как последовательность из частей кода UTF-16.

Строка — это последовательная коллекция символов, используемая для представления текста. Объект — это последовательная [System.Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) коллекция объектов, [System.Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) представляющих строку; объект соответствует блоку кода UTF-16. [String](https://docs.microsoft.com/ru-ru/dotnet/api/system.string?view=netframework-4.8) Значение [String](https://docs.microsoft.com/ru-ru/dotnet/api/system.string?view=netframework-4.8) объекта является содержимым последовательной [System.Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) коллекции объектов, которое является неизменяемым (т. е. доступно только для чтения). Дополнительные сведения о неизменности строк см. в разделе неизменяемость [и класс StringBuilder](https://docs.microsoft.com/ru-ru/dotnet/api/system.string?view=netframework-4.8#Immutability) далее в этом разделе. Максимальный размер [String](https://docs.microsoft.com/ru-ru/dotnet/api/system.string?view=netframework-4.8) объекта в памяти составляет 2 ГБ или около 1 000 000 000 символов.

Каждый символ в строке определяется скалярным значением Юникода, также называемым кодовой точкой Юникода или порядковым (числовым) значением символа Юникода. Каждая кодовая точка кодируется с помощью кодировки UTF-16, а числовое значение каждого элемента кодировки представлено [Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) объектом.

Один [Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) объект обычно представляет одну кодовую точку, то есть числовое значение [Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) равно кодовой точке. Например, кодовая точка для символа «a» — это U + 0061. Однако для кодовой точки может потребоваться более одного закодированного элемента (более одного [Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) объекта). Стандарт Юникода определяет два типа символов, которые соответствуют нескольким [Char](https://docs.microsoft.com/ru-ru/dotnet/api/system.char?view=netframework-4.8) объектам: графемес, и дополнительные кодовые точки Юникода, которые соответствуют символам в дополнительных плоскостях Юникода.

# **Задания**

1. Дано вещественное число — цена 1 кг конфет. Вывести стоимость 0.1, 0.2, . . . , 1 кг конфет.

2. Дано целое число N (> 0). Найти произведение 1.1 · 1.2 · 1.3 · . . . (N сомножителей).

3. Дано целое число N (> 0). Найти квадрат данного числа, используя для его вычисления следующую формулу: N2 = 1 + 3 + 5 + . . . + (2·N − 1). После добавления к сумме каждого слагаемого выводить текущее значение суммы

4. Дано вещественное число A и целое число N (> 0). Используя один цикл, найти сумму 1 + A + A2 + A3 + . . . + AN

5. Дано вещественное число A и целое число N (> 0). Используя один цикл, найти значение выражения

1 − A + A2 − A3 + . . . ± AN .

Условный оператор не использовать.

**Блок-схемы**

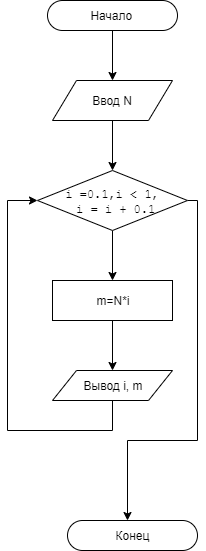


Рисунок 1 — Блок-схема к заданию 1.

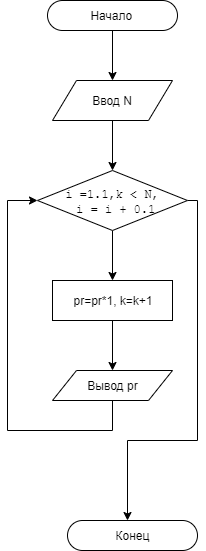


Рисунок 2 — Блок-схема к заданию 2.

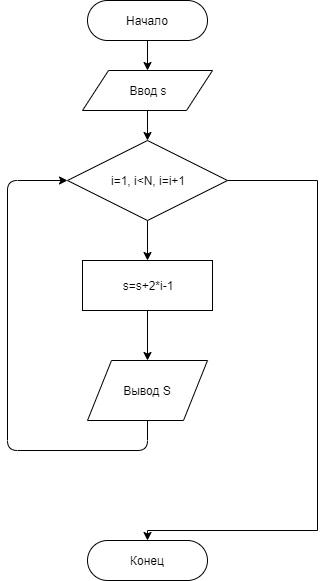


Рисунок 3 — Блок-схема к заданию 3.

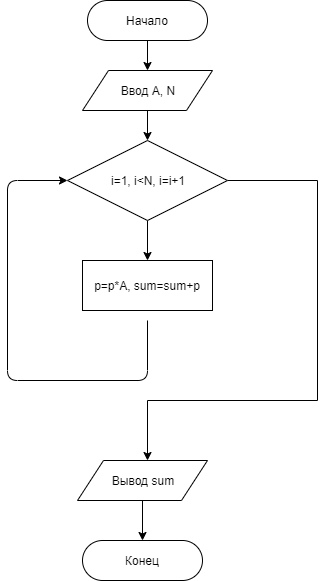


Рисунок 4 — Блок-схема к заданию 4.

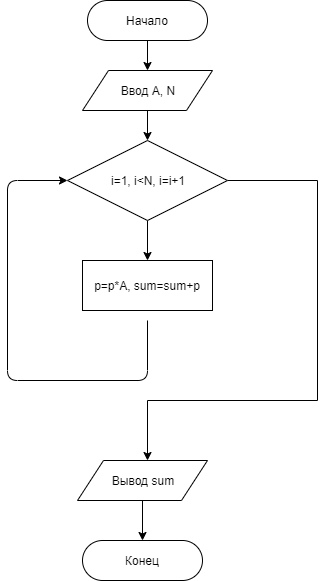


Рисунок 5 — Блок-схема к заданию 5.

# **Коды программ**

Листинг 1 —Задание 1(Задача с дробными числами)

1. ﻿using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks;
6. namespace ConsoleApp1
7. {
8. class Program
9. {
10. static void Main(string[] args)
11. {
12. double i, m ;
13. Console.Write(" Введите сумму за 1кг ");
14. double N = Convert.ToDouble(Console.ReadLine());
15. for ()
16. {
17. m = N \* i;
18. Console.Write("" + i);
19. Console.WriteLine("кг = "+ m);
20. }
21. Console.ReadLine();
22. }
23. }
24. }

Листинг 2 —Задание 2 (Нахождение произведения элементов)

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks;
6. namespace ConsoleApp2
7. {
8. class Program
9. {
10. static void Main(string[] args)
11. {
12. double i, k=0, pr=1;
13. Console.Write(" Введите кол-во элементов ");
14. int N = Convert.ToInt32(Console.ReadLine());
15. for(i=1.1; k < N; i = i + 0.1)
16. {
17. pr = pr \* i;
18. k = k + 1;
19. }
20. Console.WriteLine(" произведение = " +pr );
21. Console.ReadLine();
22. }
23. }
24. }

Листинг 3 —Задание 3 (Поиск суммы по формуле)

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks;
6. namespace ConsoleApp3
7. {
8. class Program
9. {
10. static void Main(string[] args)
11. {
12. int s=0, i;
13. Console.Write(" Введите N ");
14. double N = Convert.ToDouble(Console.ReadLine());
15. for(i= 1; i<N; i++)
16. {
17. s = s + 2 \* i - 1;
18. Console.WriteLine( "sum now = " + s );
19. }
20. Console.ReadLine();
21. }
22. }
23. }

Листинг 4 —Задание 4(Поиск суммы и произведения)

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks;
6. namespace ConsoleApp4
7. {
8. class Program
9. {
10. static void Main(string[] args)
11. {
12. double sum=1, p=1,i;
13. Console.Write(" Введите A ");
14. double A = Convert.ToDouble(Console.ReadLine());
15. Console.Write(" Введите N ");
16. double N = Convert.ToDouble(Console.ReadLine());
17. for (i = 1; i < N; i++)
18. {
19. p = p \* A;
20. sum = sum + p;
21. }
22. Console.Write(" sum = " +sum);
23. Console.ReadLine();
24. }
25. }
26. }

Листинг 5 - Задание 5 (Поиск суммы и произведения с условием)

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks;
6. namespace ConsoleApp4
7. {
8. class Program
9. {
10. static void Main(string[] args)
11. {
12. double sum = 1, p = 1, i;
13. Console.Write(" Введите A ");
14. double A = Convert.ToDouble(Console.ReadLine());
15. Console.Write(" Введите N ");
16. double N = Convert.ToDouble(Console.ReadLine());
17. for (i = 1; i < N; i++)
18. {
19. p = (-1)\*p \* A;
20. sum = sum + p;
21. }
22. Console.Write(" sum = " + sum);
23. Console.ReadLine();
24. }
25. }
26. }

# 

# **Результаты выполнения программ**

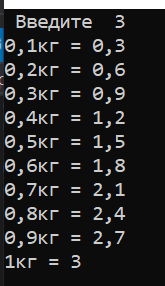


Рисунок 6 — результат выполнения программы 1.

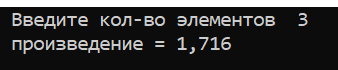


Рисунок 7 — результат выполнения программы 2.

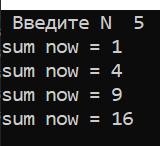


Рисунок 8 — результат выполнения программы 3.

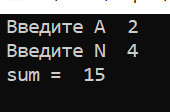


Рисунок 9— результат выполнения программы 4.

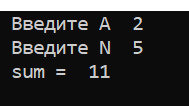


Рисунок 10 — результат выполнения программы 5.