

Лабораторная работа №2

Тема: Построение блок-схем итерационных вычислительных процессов

Порядок выполнения работы:

- ознакомиться с описанием лабораторной работы;
- получить задание у преподавателя, согласно своему варианту;
- написать программу и отладить ее на ЭВМ;
- оформить отчет.

Теоретические сведения

Теоретический материал по лабораторной работе приведен в:

- 1) соответствующем лекционном материале;
- 2) работах [1-6] из списка литературы;
- 3) <http://msdn.microsoft.com/ru-ru/visualc/default.aspx>
- 4) В разделе «Help» меню Visual Studio 2017;

Теоретические сведения. Существует ряд циклических вычислительных процессов, называемых *итерационными*. Итерационный процесс продолжается до тех пор, пока разность между соседними, уточняемыми на каждом шаге цикла значениями (итерациями), не окажется меньше либо равной некоторой заданной величине (точности). Характерной особенностью итерационного процесса является то, что количество циклов (итераций) в нем заранее не известно и становится определенным только после окончания вычислений. Выход из цикла происходит тогда, когда анализируемая величина или величины на очередной итерации отличаются от эталонных величин (например, заданной точности). В таком цикле, как правило, результаты вычислений предыдущего шага цикла используются как исходные данные при выполнении следующего шага цикла (пример 2.1).

Пример 2.1. Составить блок-схему нахождения приближённого значения квадратного корня $y = \sqrt{x}, x > 0$ как предела последовательности y_1, y_2, \dots, y_k ,

где $y_{k+1} = \frac{1}{2} \left(y_k + \frac{x}{y_k} \right)$. Точность полученного приближения оценить по

величине отклонения двух соседних приближений y_k и y_{k+1} оценить по величине $\Delta_k = |y_k - y_{k+1}| \leq \varepsilon$, где $\varepsilon > 0$ заданная точность вычислений (рис 2.1).

Пример 2.2. Составить блок-схему вычисления с точностью $\varepsilon > 0$ отрезка

степенного ряда для функции $\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \dots + (-1)^{m-1} \frac{x^{2m-1}}{(2m-1)!} + \dots$,

$x \in \mathbb{R}$. Точность считать достигнутой, если последний член ряда не превосходит абсолютной величины заданного ε (рис 2.2).

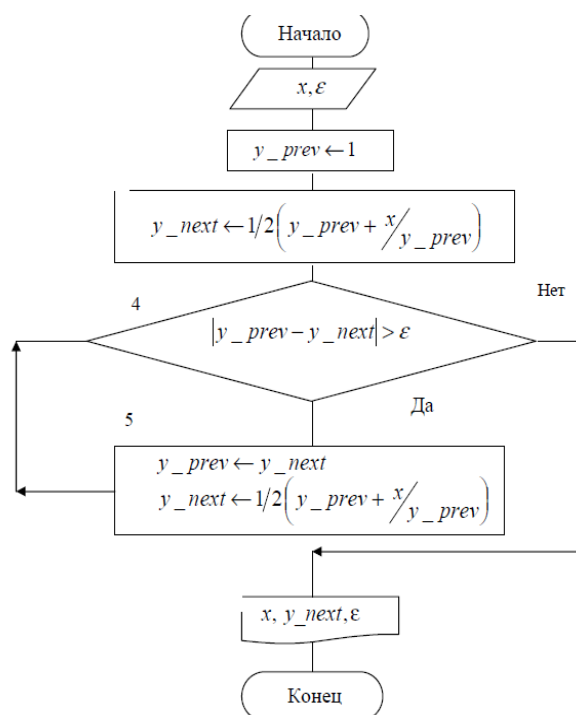


Рис 2.1. Блок-схема примера 2.1

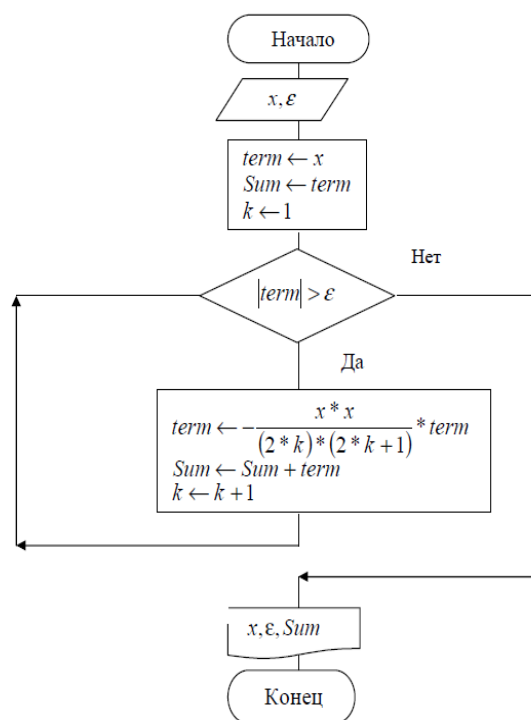


Рис 2.2. Блок-схема примера 2.2

Варианты заданий для самостоятельной работы

1. Даны действительные числа x , n ($x \neq 0$, $n > 0$). Вычислить приближенное значение конечной суммы. (табл. 2.1).

Таблица 2.1

№	Вычислить сумму	Вычислить произведения
1	2	3
1	$Y = \frac{\sum_{n=1}^{10} n^2 + \sin x}{x + 2}$	$z = y^3 + \prod_{n=1}^3 n^2$
2	$Y = 2 \sum_{i=1}^5 \ln(ix) + x^5$	$Z = \prod_{k=1}^5 \sin(kx) + 2$
3	$Y = \frac{\sum_{k=1}^{10} \sin(kx) + 2x}{x^2 + 5}$	$Z = 3 \sin y + \prod_{m=2}^{10} \ln(my)$
4	$Y = \ln\left(\sum_{i=1}^5 i\right) + x^2$	$Z = \prod_{i=1}^5 \arctg(ix) + \sqrt{x}$

Продолжение таблицы 5.1

1	2	3
5	$Y = \frac{\cos\left(\sum_{n=1}^5 n^3\right) + \ln x}{\arctg^2 x}$	$Z = \sin\left(\prod_{i=1}^{10} \ln(ix)\right)$
6	$Y = \sum_{n=1}^{10} \ln(nx) - x^2$	$Z = \arctg x + \prod_{n=1}^5 \ln(nx)$
7	$Y = \frac{\cos x + \sum_{k=1}^5 \ln(kx)}{0.5 + x^5}$	$Z = y^5 + \prod_{k=1}^5 \ln(ky)$
8	$Y = \sum_{n=1}^5 \cos(nx) + 5x^2$	$Z = \cos \prod_{m=2}^5 \ln(my)$
9	$Y = \frac{\cos x + \sum_{i=1}^{10} \ln(ix)}{x^2 + 2}$	$Z = \prod_{n=1}^5 \arctg(nx)$
10	$Y = \arctg\left(\sum_{i=1}^5 i + 2x\right)$	$Z = x^5 + \prod_{k=1}^{10} k^3$
11	$Y = \frac{\sin \sum_{i=1}^5 (ix) + x^2}{x^4 + 2}$	$Z = y^3 + \prod_{m=1}^5 m^3$
12	$Y = \cos x + \sum_{i=1}^3 \ln(ix)$	$Z = \frac{\sin y + 2 \prod_{m=1}^5 (my)}{y^3 + 3}$
13	$Y = \frac{\arctg x^2 + \sum_{n=1}^5 \cos n}{x + 5}$	$Z = \prod_{k=1}^{10} \ln(ky) + y^2$
14	$Y = \frac{\ln\left(\sum i^3\right) + 2 \sin^2 x}{x^2 + 5}$	$Z = \prod_{m=1}^5 \cos(my) + 2y$
15	$Y = \cos\left(\sum_{i=1}^5 \sin(ix)\right) + 2x$	$Z = \frac{x^5 + \sqrt{x} + 2x}{\prod_{k=1}^{10} \ln(kx)}$

Завершение таблицы 5.1

1	2	3
16	$Y = \frac{\ln \sum_{n=1}^5 \cos(nx) + 2x}{x^4 + 5}$	$Z = \prod_{n=1}^{10} \ln(nx) + 2x^2$
17	$Y = \frac{\arctg x + 5}{\sum_{k=1}^5 k + x^4}$	$Z = \frac{\cos\left(\prod_{i=1}^5 i\right) + 2 \sin y}{y^2 + 5}$
18	$Y = \frac{x^3 + \sin^2 x - 3}{\sum_{k=1}^5 k + x^4}$	$Z = \arctg\left(\prod_{m=1}^5 m\right) + y^5$
19	$Y = \ln x + 2 \sum_{k=1}^{10} k$	$Z = \frac{y^5 + \sqrt[3]{y} - 2}{\prod_{k=1}^5 \ln(ky)}$
20	$Y = \frac{\sin\left(\sum_{k=1}^{10} k\right) + x^2}{x^2 + 2}$	$Z = \prod_{k=1}^{10} k + 2 \ln y$

2. Даны действительные числа x , ε ($\varepsilon > 0$). Вычислить с заданной точностью ε приближенное значение бесконечной суммы и сравнить его с точным, используя программное решение в C#.

№	Задание	Точное значение
1	$x^2 - \frac{(-x^2)^2}{2} + \frac{(-x^2)^3}{3} + \dots + (-1)^{m-1} \frac{(-x^2)^m}{m} + \dots;$	$\ln(1 - x^2),$ $x \in]-1; 1[$
2	$1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \dots + (-1)^m \frac{x^{2m}}{(2m)!} + \dots;$	$\cos x, x \in R$
3	$x + \frac{x^3}{3!} + \frac{x^5}{5!} + \dots + \frac{x^{2m-1}}{(2m-1)!} + \dots;$	$\operatorname{sh} x, x \in R$
4	$1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \dots + \frac{x^{2m}}{(2m)!} + \dots;$	$\operatorname{ch} x, x \in R$
5	$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots + (-1)^{m-1} \frac{x^m}{(m)};$	$\ln(1 + x), x \in]-1; 1[$
6	$x - \frac{x^3}{3} + \frac{x^5}{5} + \dots + (-1)^{m-1} \frac{x^{2m-1}}{(2m-1)};$	$\arctg x, x \in [-1; 1]$

7	$-x - \frac{x^2}{2} - \frac{x^3}{3} - \frac{x^4}{4} - \dots - \frac{x^m}{m} - \dots;$	$\ln(1-x), x \in]-1;1[$
8	$1 - x + x^2 - \dots + (-1)^n x^n + \dots;$	$\frac{1}{1+x}, x \in]-1;1[$
9	$1 + x + x^2 + \dots + x^n + \dots;$	$\frac{1}{1-x}, x \in]-1;1[$
10	$1 + \alpha x + \frac{\alpha(\alpha-1)}{2!} x^2 + \dots + \frac{\alpha(\alpha-1)\dots(\alpha-n+1)}{n!} x^n + \dots;$	$(1+x)^\alpha, x \in]-1;1[$
11	$2 \left(x + \frac{x^3}{3} + \frac{x^5}{5} + \dots + \frac{x^{2m-1}}{(2m-1)} + \dots \right);$	$\ln \frac{(1+x)}{(1-x)}, x \in]-1;1[$
12	$1 - \frac{1}{2}x + \frac{3}{8}x^2 - \frac{5}{16}x^3 + \dots + (-1)^{m-1} \frac{(2n-3)!!}{(2n)!!} x^n + \dots;$	$\sqrt{\frac{1}{1-x}}, x \in]-1;1[$
13	$2 - \frac{2^3 x^3}{3!} + \frac{2^5 x^5}{5!} - \dots + \frac{(-1)^{m-1}}{(2m-1)!} 2^{2m-1} x^{2m-1} + \dots;$	$\frac{\sin 2x}{x}, x \in R$
14	$1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \dots + \frac{x^{2m}}{(2m)!} + \dots;$	$chx, x \in R$
15	$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots + (-1)^{m-1} \frac{x^m}{(m)};$	$\ln(1+x), x \in]-1;1[$
16	$1 - 3x^3 + (3x^3)^2 - \dots + (-1)^n 3^n x^{3n} + \dots;$	$\frac{1}{1+3x^3}, x \in \left] -\frac{1}{\sqrt[3]{3}}; \frac{1}{\sqrt[3]{3}} \right[$
17	$\left(x - \frac{3^2 x^3}{2!} + \frac{3^4 x^5}{4!} + \dots + (-1)^n \frac{3^{2n} x^{2n+1}}{(2n)!} + \dots \right);$	$x \cos 3x,$ $x \in R$
18	$1 + \alpha x + \frac{\alpha(\alpha-1)}{2!} x^2 + \dots + \frac{\alpha(\alpha-1)\dots(\alpha-n+1)}{n!} x^n + \dots;$	$(1+x)^\alpha, x \in]-1;1[$
19	$2 \left(x + \frac{x^3}{3} + \frac{x^5}{5} + \dots + \frac{x^{2m-1}}{(2m-1)} + \dots \right);$	$\ln \frac{(1+x)}{(1-x)}, x \in]-1;1[$
20	$1 - \frac{1}{2}x + \frac{3}{8}x^2 - \frac{5}{16}x^3 + \dots + (-1)^{m-1} \frac{(2n-3)!!}{(2n)!!} x^n + \dots;$	$\sqrt{\frac{1}{1-x}}, x \in]-1;1[$

Пример выполнения 2 задания

Запустим на выполнение среду Microsoft Visual Studio.NET через кнопку «Пуск» панели задач. В открывшемся окне выберем пункт File|New|Project.

В диалоговом окне New Project необходимо выбрать тип проекта Visual C# Projects и шаблон приложения - Console Application.

В поле наименование (Name) ввести новое имя проекта, либо согласиться с именем, которое предлагает.

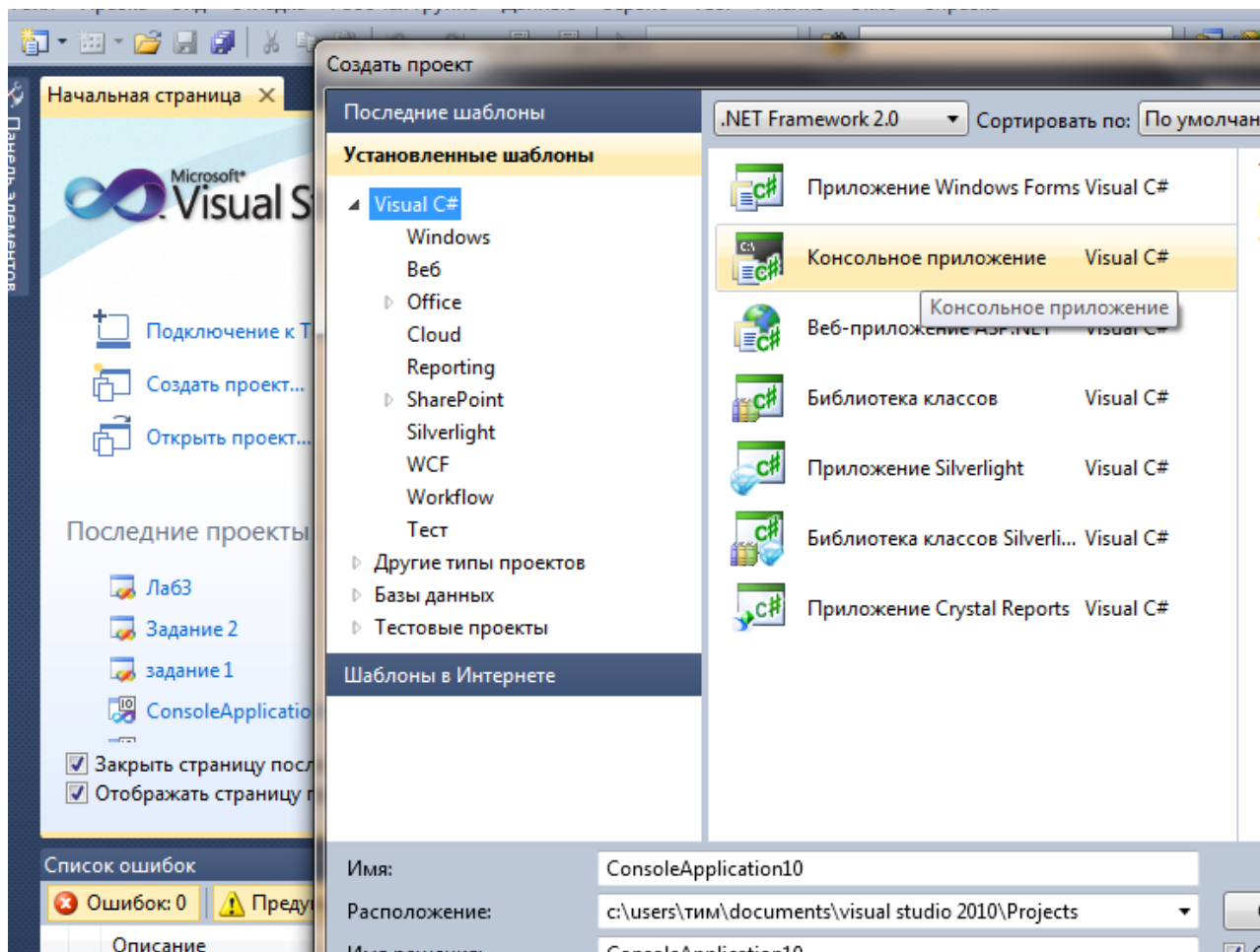


Рис 5.3. Окно New Project

При этом появится шаблон приложения, подготовленный для нас мастером рис. 2.4. Первая строка проекта *using System;*, включает в себя директиву *using*, которая сообщает компилятору, где он должен искать классы (типы), не определенные в данном пространстве имен. Мастер, по умолчанию, указывает стандартное пространство имен *System*, где определена большая часть типов среды .NET.

Следующей строкой *namespace ConsoleApplication1* мастер предложения определяет пространство имен для нашего приложения. По умолчанию в качестве имени выбирается имя проекта. Область действия пространства имен определяется блоком кода, заключенного между открывающей и закрывающей фигурными скобками. Пространство имен обеспечивает способ хранения одного набора имен отдельно от другого.

Имена, объявленные в одном пространстве имен не конфликтуют, при совпадении, с именами, объявленными в другом пространстве имен.

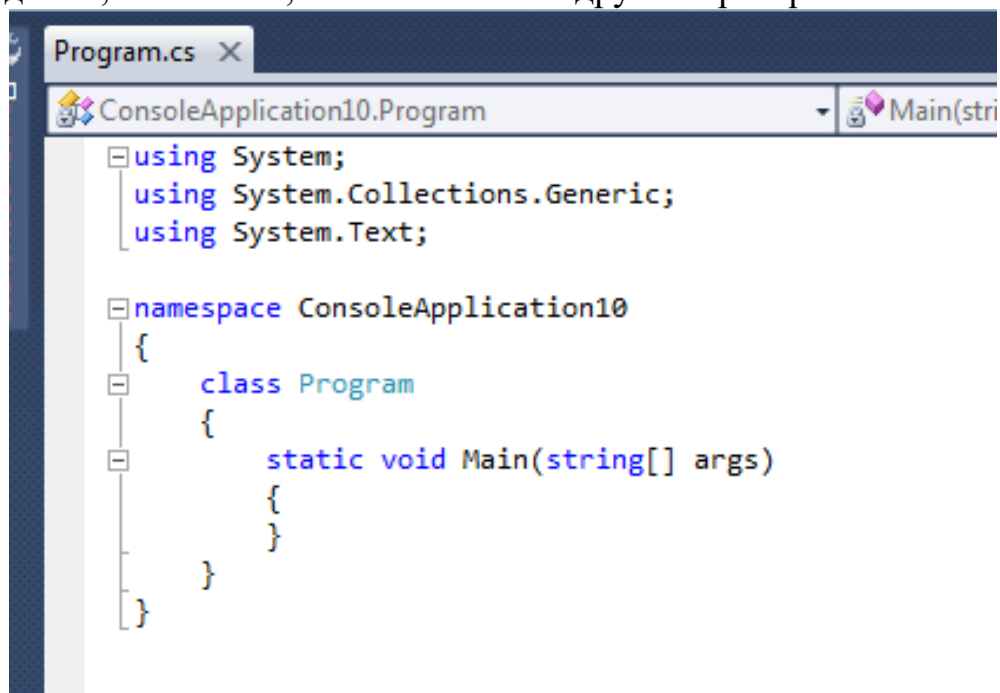


Рис 5.4. Окно Console Application

В шаблоне приложения имеется множество строк, которые являются комментариями.

В C# определены три вида комментариев:

- многострочный (/*...*/)
- однострочный (//...)
- XML (///) – комментарий для поддержки возможности создания самодокументированного кода.

Пример задания 2.

1	$1 + \frac{x}{1!} + \frac{x^2}{2!} + \dots + \frac{x^n}{n!} + \dots;$	$e^x, x \in R$
----------	---	----------------

Листинг задания 2.

namespace ConsoleApplication10

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("_Введите число x");

string buf; //строка - буфер для ввода чисел

double s, i, k, factorial; //описываем все переменные,

//что будут задействованы в программе

buf = Console.ReadLine(); //задание x

double x = Convert.ToDouble(buf); //считывание x

Console.WriteLine("_x = " + x); //вывод x на дисплей

Console.WriteLine("_Введите число n");

```

buf = Console.ReadLine();
double n = Convert.ToDouble(buf);
Console.WriteLine("_n = " + n);
double c = Math.Exp(x);           //расчёт точного значения
Console.WriteLine("_exp(x) = " + c);
                                   //вывод точного значения на дисплей
                                   //расчёт приближённого значения

```

экспоненты

```

s = 1;
for (i = 1; i <= n; i = i + 1)    //задание цикла for
{
    factorial=1;
    for (k = 1; k <= i; k = k+1)    // расчёт факториала
    { factorial = factorial * k; }
    s=s+Math.Pow(x,i)/factorial;
}
Console.WriteLine("_расчётное= " + s);
                                   // вывод приближённого значения
double pohubka=Math.Abs(c-s);
Console.WriteLine("_погрешность={0:f8}", pohubka);
                                   // вывод приближённого значения
Console.ReadKey();//ожидание программы
}
}
}

```

Для того, чтобы проект запустить на выполнения необходимо воспользоваться кнопкой F5 или на панели инструментов нажать на кнопку отладки (рис 2.5)

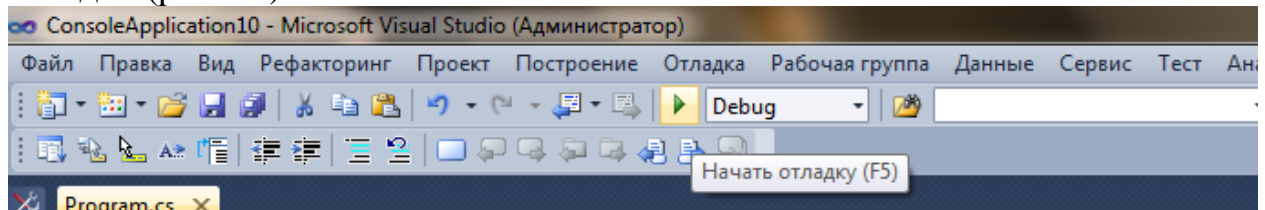


Рис 5.5. Панель управления C#

После чего получим искомый результат(рис 2.6).

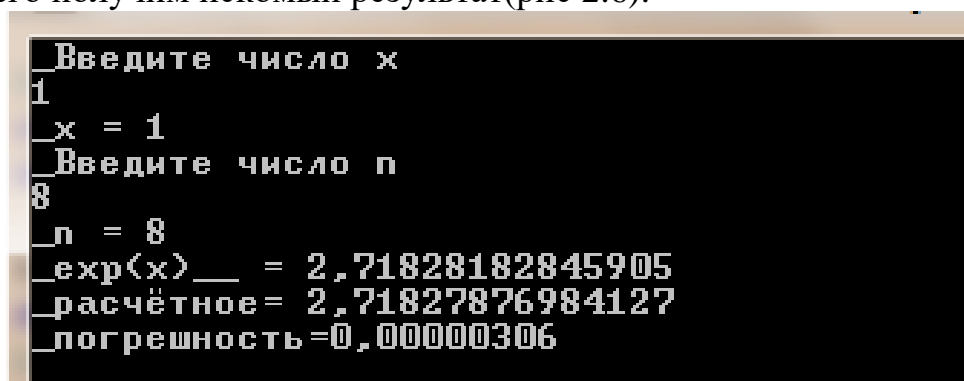


Рис 2.6. Результат выполнения задания 2

Лабораторная работа №3

Тема: Алгоритмизация и программирование на языке высокого уровня

Цель работы:, приобретение навыков практического применения знаний для создания простейших программ с неразветвлённым вычислительным процессом в среде Microsoft Visual C#.

Порядок выполнения работы:

- ознакомиться с описанием лабораторной работы;
- получить задание у преподавателя, согласно своему варианту;
- написать программу;
- оформить отчет.

Теоретические сведения

Теоретический материал по лабораторной работе приведен в:

- 1) соответствующем лекционном материале;
- 2) работах [1-5, 7] из списка литературы;

Краткие теоретические сведения

Типы данных

Язык C# имеет набор встроенных типов, которые рассматриваются как псевдонимы типов в пространстве имен System. Например, тип string - это псевдоним типа System.String, а тип int - псевдоним типа System.Int32. Все встроенные типы подразделены на группы: целочисленные типы; вещественные типы; логический тип; символьные типы; объектный тип (object). Описание типов приведено в таблице 3.1.

Таблица 6.1

Тип данных	Ключевое слово	Псевдоним класса библиотеки NET	Описание	Размер (бит)
логический	bool	System.Boolean	-	-
целый	Int	System.Int32	со знаком	32
	short	System.Int16	со знаком	16
	byte	System.Byte	без знака	8
	sbyte	System.SByte	со знаком	8
	long	System.Int64	со знаком	64
вещественный	float	System.Single	7 цифр	32
	double	System.Double	15 цифр	64
Строковый символьный	string	System.String	строка	-
	char	System.Char	Символов Unicode	16
Любой тип	object	System.Object	объектный	-

Иерархия классов NET Framework имеет один общий корень - класс System.Object. Все типы разделяются на две категории: размерные типы и ссылочные типы.

При создании переменной размерного типа под нее в стеке выделяется определенный объем памяти, соответствующий типу этой переменной. При передаче такой переменной в качестве параметра выполняется передача значения, а не ссылки на него. Значение размерного типа не может быть равным null. К размерным типам, например, относятся целочисленные и вещественные типы, структуры.

При создании переменной ссылочного типа память под созданный объект выделяется в другой области памяти, называемой кучей. Ссылка всегда указывает на объект заданного типа.

Структура приложения на языке C#.

Проектом называется совокупность файлов, содержащих информацию об установках, конфигурации, ресурсах проекта, а также файлов исходного кода и заголовочных файлов.

Интегрированная среда проектирования Visual Studio позволяет для создания проектов на разных языках программирования использовать различные инструментальные средства проектирования (например, Microsoft Visual Basic, Microsoft Visual C#).

Любое приложение на языке C#, разрабатываемое в среде проектирования Visual Studio, реализуется как отдельный проект. Приложение на языке C# может состоять из нескольких модулей. Каждый модуль C# может содержать код нескольких классов (при создании приложения в среде Visual Studio.NET каждый класс C# автоматически помещается в отдельный модуль - файл с расширением cs).

Для консольного приложения один из классов, реализуемых модулем, должен содержать метод Main. В языке C# нет аппарата заголовочных файлов, используемого в языке C++, поэтому код модуля должен содержать как объявление, так и реализацию класса. По умолчанию весь код класса, представляющего консольное приложение, заключается в одно пространство имен, одноименное с именем приложения.

Точкой входа в программу на языке C# является метод Main. Этот метод может записываться как без параметров, так и с одним параметром типа string - указателем на массив строк, который содержит значения параметров, введенных при запуске программы. В отличие от списка параметров, задаваемых при запуске C-приложения, список параметров C#-приложения не содержит в качестве первого параметра имя самого приложения. Код метода указывается внутри фигурных скобок:

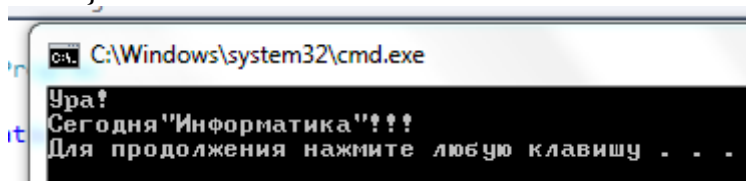
```
static void Main(string[] args)
{
}
```

Ключевое слово static определяет, что метод Main является

статическим методом, вызываемым без создания экземпляра объекта типа класса, в котором этот метод определен. Метод, не возвращающий никакого значения, указывается с ключевым словом `void`. Однако метод `Main` может возвращать значение типа `int`.

Пример 6.1. Вывод сообщения на консоль.

```
static void Main()
{
    Console.WriteLine("Ура!\nСегодня\"Информатика\"!!!");
}
```



Замечание. Для отладки можно использовать команду меню `Debug\Start Without Debugging`. На экране появится окно с результатом исполнения. Обратите внимание на надпись в конце программы: `Press any key to continue`, которая не была предусмотрена. При нажатии любой клавиши окно закрывается. Это результат срабатывания встроенной разработчиками компилятора функции «остановки экрана» для того, чтобы можно было бы сколько угодно долго его рассматривать.

Можно использовать команду `Debug\Start Debugging`, но тогда окно закроется и мы не сможем рассмотреть искомый результат. Для того чтобы обойти это неудобство, следует при разработке программы предусмотреть собственную остановку экрана. Для этого используется команда `Console.Read();`

Константы

Это неизменяемые в процессе выполнения программы величины.

Целые константы - наиболее распространенный тип `int`. Это целое число, которое может быть отрицательным, положительным или нулем -12, 5, 0 (все целые со знаком 32 бита). Их можно записывать с суффиксом -12L (длинное целое 64 бита), 5u (целое без знака 8 бит)

Вещественные константы с фиксированной точкой. При записи константы типа `float` (32 бита) необходимо, чтобы за значением шел суффикс символ `f` или `F` 1.2, -1.234, при записи константы типа `double` (64 бита) можно записать суффикс «`d`» или «`D`», но это не является обязательным условием: 1234.5678, 12.3d. Дробная часть отделяется от целой части точкой.

Вещественные константы с плавающей точкой. При записи константы типа `float` (32 бита) необходимо, чтобы за значением шел суффикс символ `f` или `F`: 1.2E-3f (число 0.0012), при записи константы типа `double` (64 бита) -1.34E5 (число -134000) наличие суффикса не требуется.

Символьные константы. Символьная константа `char` может представлять собой 16-битный символ Unicode ('a') или управляющие символы (возврат каретки ('\r'), перевод страницы ('\f'), горизонтальную табуляцию ('\t'), и другие), заключенный в апострофы.

Строковые константы - это последовательность символов, заключенная в кавычки, или константы string. Строка, состоящая из символов, например "Ура!\nСегодня\Информатика\!!!!"

Логическая константа. Задается одним из двух значений true («истина») или false («ложь»). Используется в С# в логических выражениях, операторах условного перехода.

Именованные константы. Применяются для того, чтобы вместо значений констант, использовать в программе их имена, например константа р вещественная одинарной точности

```
const float p = 3.14159f
```

Переменные

Переменная - именованная область памяти, для хранения данных определенного типа. При выполнении программы значение переменной величины можно изменять. Все переменные должны быть описаны явно, при описании переменной задается ее значение и тип. При объявлении переменной может быть задано начальное значение.

Имя переменной может содержать буквы, цифры и символ подчеркивания. Прописные и строчные буквы различаются. Например, переменные Long, LONG, long - три разных переменные.

Имя переменной может начинаться с буквы или знака подчеркивания, но не цифры. Имя переменной не должно совпадать с ключевыми словами. Не рекомендуется начинать имя с двух подчеркиваний (такие имена зарезервированы для служебного использования).

Правильные имена переменных: MaxLen, iMaxLen, Max_Len

Неправильные имена переменных: 2Len, Le#

Примеры описания переменных:

```
int a = -14;                // числовая целая 32 бита
float c = -0.00151f;        // числовая вещественная 32 бита
double i = 1234.56789;      // числовая вещественная 64 бита
bool l = false;             // логическая 16 бит
string name = "Petrov";     // строковая
```

Выражение - состоит из одного или более операндов (которые могут быть переменными, константами, функциями или символьными значениями), знаков операций и круглых скобок.

Примеры выражений:

2 * 2 + 1 полученное значение 5

1 - 3 полученное значение -2

1 / 2 - 3 полученное значение - 2.5

Присвоение значения переменной представляет оператор присваивания (знаки основных операций приведены в таблице 1.2): $y = 2 * x * x + 3 * x - 1$.

В этом примере сначала производятся вычисления правой части оператора присваивания « = », а затем полученное значение присваивается переменной у. Для текстовых данных выражение можно записать в

следующем виде:

```
string kaf = "Кафедра" + "ИСиДТ";
```

В этом примере строки по правую сторону от оператора присваивания объединяются, чтобы получить строку “Кафедра + ИСиДТ”, которая затем присваивается переменной kaf.

Таблица 3.2 Знаки операций

Знак операции	Название
+	Сложение
-	Вычитание
*	Умножение
/	Деление
%	Остаток от деления

Если в арифметических выражениях используются целые числа, то результатом вычислений будет целое число, и любой остаток от деления будет отброшен. Для получения остатка можно использовать соответствующую операцию %, например $10 \% 3$ возвращает остаток от целочисленного деления, равный 1. Когда в арифметических выражениях используются числа с плавающей точкой, то результатом деления $10f / 3f$ будет число 3,333333. **Математические функции**

C# содержит большое количество встроенных математических функций которые реализованы в классе Math пространства имен System.

Рассмотрим краткое описание некоторых математических функций, подробнее с ними можно познакомиться в справочной системе VS или технической документации. Особое внимание следует обратить на типы операндов и результатов, т. к. каждая функция может иметь несколько перегруженных версий.

Замечание. Использование нескольких функций с одним и тем же именем, но с различными типами параметров, называется перегрузкой функции. Например, функция Math.Abs(), вычисляющая модуль числа, имеет 7 перегруженных версий: double Math.Abs(double x), float Math.Abs(float x), int Math.Abs(int x), и т. д. (таблица 3.3)

Таблица 6.3 Математические функции

№	Название	Описание
1	2	3
1.	Math.Abs(выражение)	Модуль
2.	Math.Ceiling(выражение)	Округление до большего целого
3.	Math.Cos(выражение)	Косинус
4.	Math.E	Число e
5.	Math.Exp(выражение)	Экспонента

6.	Math.Floor(<i>выражение</i>)	Округление до меньшего целого
7.	Math.Log(<i>выражение</i>)	Натуральный логарифм
8.	Math.Log10(<i>выражение</i>)	Десятичный логарифм
9.	Math.Max(<i>выражение1</i> , <i>выражение2</i>)	Максимум из двух значений
10.	Math.Min(<i>выражение1</i> , <i>выражение2</i>)	Минимум из двух значений

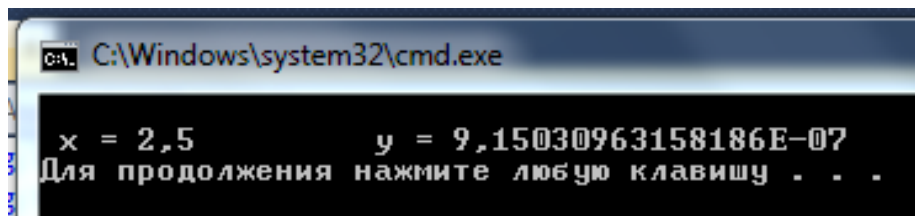
Завершение таблицы 3.3

1	2	3
11.	Math.PI	Число π
12.	Math.Pow(<i>выражение1</i> , <i>выражение2</i>)	Возведение в степень
13.	Math.Round(<i>выражение</i>) <i>Math.Round(выражение, число)</i>	Простое округление Округление до заданного числа цифр
14.	Math.Sign(<i>выражение</i>)	Знак числа
15.	Math.Sin(<i>выражение</i>)	Синус
16.	Math.Sqrt(<i>выражение</i>)	Квадратный корень
17.	Math.Tan(<i>выражение</i>)	Тангенс

Пример 6.2. Вычислить значения $Y = \frac{\cos \pi x}{1 + x^2}$ функции при $x=2,5$

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace ConsoleApplication1
{
    class Example2    // начало описания класса
    // Example2
    {
        static void Main()
        {
            double p = 3.14159;
            double x = 2.5;
            double y = Math.Cos(p * x) / (1 + x * x);
            Console.WriteLine();
            Console.WriteLine(" x = {0} \t y = {1} ", x, y);
        }
    }
}
```

Эта программа выводит следующее окно с результатом:



Организация ввода-вывода данных.

Программа при вводе данных и выводе результатов взаимодействует с внешними устройствами. Совокупность стандартных устройств ввода (клавиатура) и вывода (экран) называется консолью. В языке C# нет операторов ввода и вывода. Вместо них для обмена данными с внешними устройствами используются специальные объекты. В частности, для работы с консолью используется стандартный класс `Console`, определенный в пространстве имен `System`.

Ввод данных

Для ввода данных обычно используется метод `ReadLine`, реализованный в классе `Console`. Особенностью данного метода является то, что в качестве результата он возвращает строку (`string`).

Пример:

```
static void Main()
{
    string s = Console.ReadLine();
    Console.WriteLine(s);
}
```

Для того чтобы получить числовое значение необходимо воспользоваться преобразованием данных.

Пример:

```
static void Main()
{
    string s = Console.ReadLine();
    int x = int.Parse(s); // преобразование строки в
    // число
    Console.WriteLine(x);
}
```

Или сокращенный вариант:

```
static void Main()
{
    //преобразование введенной строки в число
    int x = int.Parse(Console.ReadLine());
    Console.WriteLine(x);
}
```

Для преобразования строкового представления целого числа в тип `int` мы используем метод `int.Parse()`, который реализован для всех числовых типов данных. Таким образом, если нам потребуется преобразовать

строковое представление в вещественное, мы можем воспользоваться методом `float.Parse()` или `double.Parse()`. В случае, если соответствующее преобразование выполнить невозможно, то выполнение программы прерывается и генерируется исключение `System.FormatException` (входная строка имела неверный формат).

Вывод данных

В приведенных выше примерах мы уже рассматривали метод `WriteLine`, реализованный в классе `Console`, который позволяет организовывать вывод данных на экран. Однако существует несколько способов применения данного метода (таблица 6.4):

Таблица 6.4. Способы вывода

<code>Console.WriteLine(x);</code>	на экран выводится значение идентификатора <code>x</code>
<code>Console.WriteLine("x=" + x + "y=" + y);</code>	на экран выводится строка, образованная последовательным слиянием строки <code>"x="</code> , значения <code>x</code> , строки <code>"y="</code> и значения <code>y</code>
<code>Console.WriteLine("x={0} Y={1}", x, y);</code>	на экран выводится строка, формат которой задан первым аргументом метода, при этом вместо параметра <code>{0}</code> выводится значение <code>x</code> , а вместо <code>{1}</code> - значение <code>y</code>

Если использовать при выводе вместо метода `WriteLine` метод `Write`, вывод будет выполняться без перевода строки.

Использование управляющих последовательностей.

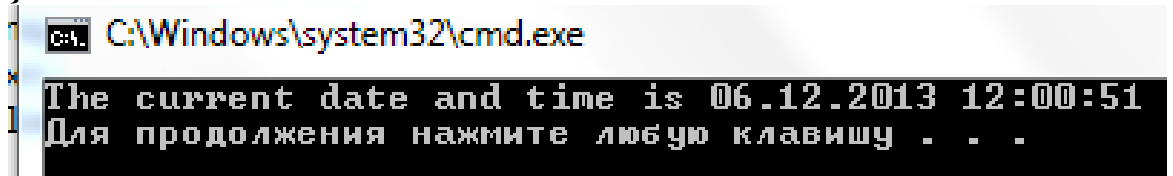
Управляющей последовательностью называют определенный символ, предваряемый обратной косой чертой. Данная совокупность символов интерпретируется как одиночный символ и используется для представления кодов символов, не имеющих графического обозначения (например, символа перевода курсора на новую строку) или символов, имеющих специальное обозначение в символьных и строковых константах (например, апостроф). Рассмотрим управляющие символы (таблица 6.5):

Таблица 6.5. Управляющие символы

Вид	Наименование	Вид	Наименование
<code>\a</code>	Звуковой сигнал	<code>\t</code>	Горизонтальная табуляция
<code>\b</code>	Возврат на шаг назад	<code>\v</code>	Вертикальная табуляция
<code>\f</code>	Перевод страницы	<code>\\</code>	Обратная косая черта
<code>\n</code>	Перевод строки	<code>\'</code>	Апостроф
<code>\r</code>	Возврат каретки	<code>\"</code>	Кавычки

Пример 3.3. Вывести сообщение о версии установленной операционной системы, текущую дату и время.

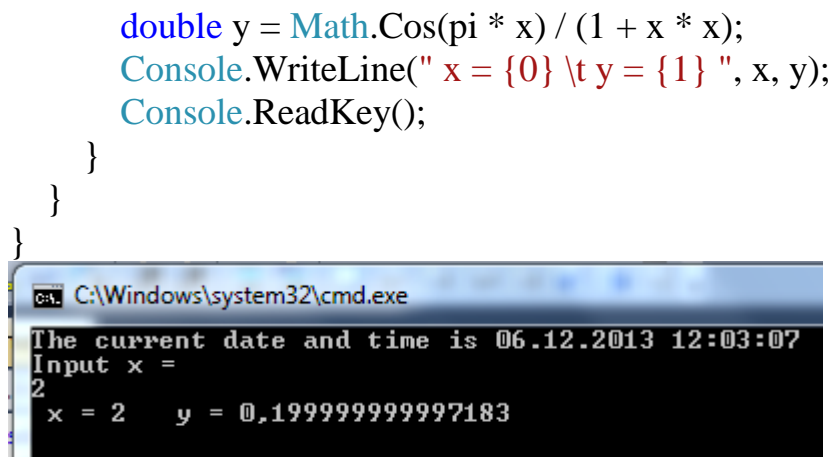
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            // вывести версию операционной системы
            OperatingSystem os = System.Environment.OSVersion;
            Console.WriteLine("Platform: {0}",os.Platform);
            System.Console.WriteLine("The current date and time is " +
System.DateTime.Now);
            // дата и время System.Console.ReadLine();
        }
    }
}
```



Пример 3.4. Использование консольного ввода для вычисления значений функции

$$Y = \frac{\cos \pi x}{1 + x^2}$$

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace lab0
{
    class Program
    {
        static void Main(string[] args)
        {
            System.Console.WriteLine("The current date and time is " +
System.DateTime.Now);
            double pi = 3.14159;
            Console.WriteLine("Input x =\r");
            double x = Convert.ToDouble(Console.ReadLine());
```



Задание 3.1

Задачи на **ввод и вывод данных, оператор присваивания, арифметические операторы, стандартные функции**. Все входные и выходные данные в заданиях этой группы являются вещественными числами.

Задание 3.1. Дано двухзначное целое число b . Найти сумму его цифр.

Текстуальная форма алгоритма задачи:

- 1) Ввести число b с клавиатуры в ЭВМ;
- 2) Для выделения первой цифры воспользуемся известным в математике способом деления числа b на 10, и возьмем целое число от этого деления;
- 3) Для выделения второй цифры разделим число b на 10, и возьмем дробную часть от этого деления;
- 4) Найдем сумму найденных цифр;
- 5) Полученный результат выдать на экран дисплея.

Графическая форма представления алгоритма дана в виде схемы (рис. 1), использующей блоки для обозначения каждого действия и приведен текст программы.

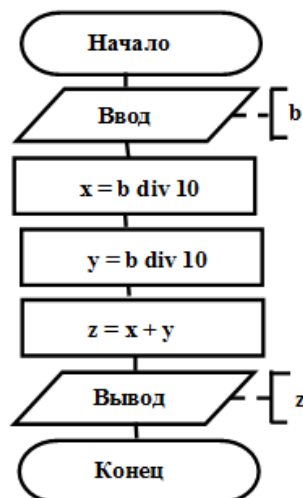


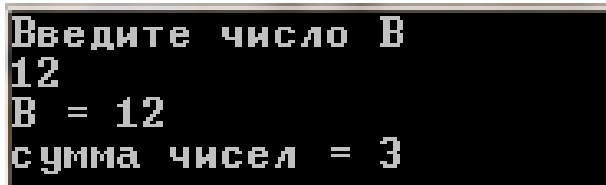
Рис. 6.1. Схема алгоритма к заданию 6.1

Напишем программу вычисления суммы двух вещественных чисел- (консольное приложение). *File- New Project - Console Application.*

Листинг задания 3.1 в программной среде C#

```
using System;
using System.Collections.Generic;
using System.Text;

namespace ConsoleApplication12
{
    class Program
    {
        static void Main(string[] args)
        {
            string buf; //строка - буфер для ввода чисел
            double b, c, d, sum; //задаём тип переменных
            Console.WriteLine("Введите число B"); //вывод сообщения на
экран
            buf = Console.ReadLine();
            b = Convert.ToDouble(buf); //переводим строковую переменную в
double-переменную
            Console.WriteLine("B = " + b); //вывод на экран исходного числа
            c = Math.Truncate(b / 10); //целая часть от деления
            d = Math.IEEERemainder(b, 10); //дробная часть от деления
            sum = c + d; //сумма цифр
            Console.WriteLine("сумма чисел = " + sum); //вывод результата на
экран
            Console.ReadKey(); //ожидание действия пользователя
        }
    }
}
```



```
Введите число B
12
B = 12
сумма чисел = 3
```

Рис. 3.2 Результат работы программы

Варианты заданий для самостоятельной работы

1. Даны катеты прямоугольного треугольника a и b . Найти его гипотенузу c и периметр P .
2. Даны два круга с общим центром и радиусами R_1 и R_2 ($R_1 > R_2$). Найти длины дуг этих кругов S_1 и S_2 , а также площадь S_3 кольца, внешний радиус которого равен R_1 , а внутренний радиус равен R_2 .
3. Дана длина L окружности. Найти ее радиус R и площадь S круга,

ограниченного этой окружностью.

4. Даны три точки А, В, С на числовой оси. Точка С расположена между точками А и В. Найти произведение длин отрезков АС и ВС.

5. Даны координаты двух противоположных вершин прямоугольника: (x_1, y_1) , (x_2, y_2) . Стороны прямоугольника параллельны осям координат. Найти периметр и площадь данного прямоугольника.

6. Даны координаты трех вершин треугольника: (x_1, y_1) , (x_2, y_2) , (x_3, y_3) . Найти его периметр и площадь, используя формулу для расстояния между двумя точками на плоскости. Для нахождения площади треугольника со сторонами a , b , c использовать формулу Герона: $S = (p \cdot (p - a) \cdot (p - b) \cdot (p - c))^{1/2}$, где $p = (a + b + c)/2$ — полупериметр.

7. Дано значение температуры T в градусах Фаренгейта. Определить значение этой же температуры в градусах Цельсия. Температура по Цельсию T_C и температура по Фаренгейту T_F связаны следующим соотношением: $T_C = (T_F - 32) \cdot 5/9$. Градус Фаренгейта

8. Известно, что X кг шоколадных конфет стоит A рублей, а Y кг ирисок стоит B рублей. Определить, сколько стоит 1 кг шоколадных конфет, 1 кг ирисок, а также во сколько раз шоколадные конфеты дороже ирисок.

9. Скорость лодки в стоячей воде V км/ч, скорость течения реки U км/ч ($U < V$). Время движения лодки по озеру T_1 ч, а по реке (против течения) — T_2 ч. Определить путь S , пройденный лодкой (путь = время · скорость). Учесть, что при движении против течения скорость лодки уменьшается на величину скорости течения.

10. Скорость первого автомобиля V_1 км/ч, второго — V_2 км/ч, расстояние между ними S км. Определить расстояние между ними через T часов, если автомобили первоначально движутся навстречу друг другу.

11. Дано трехзначное число. Найти число, полученное при прочтении его цифр справа налево.

12. Дано натуральное число n ($n > 9$). Найти число единиц в нем.

13. Дано трехзначное число. Найти сумму его крайних цифр.

14. Дано четырехзначное число. Найти произведение и сумму второй и последней цифр заданного числа.

15. Даны два целых числа. Разделить каждое число на целое число d . Найти произведение их остатков от деления.

16 Даны два целых числа. Переменной d присвоить результат деления первого числа на второе число, а переменной f присвоить результат деления второго числа на первое число. Найти произведение их остатков от деления.

17 Дано четырехзначное число. Найти сумму и произведение его цифр.

18 Дано натуральное число n ($n > 99$). Найти число десятков в нем.

19 Даны два целых числа. Разделить каждое число на целое число d . Найти сумму их целой части от деления.

20. Дано трехзначное число. Найти число, полученное при перестановке первой и последней цифр заданного числа.

Задание 3.2

Задачи на целочисленные операции. Все входные и выходные данные в заданиях этой группы являются целыми числами. Все числа, для которых указано количество цифр (двузначное число, трехзначное число и т. д.), считаются положительными.

Задачи по вариантам

1. Даны целые положительные числа A и B ($A > B$). На отрезке длины A размещено максимально возможное количество отрезков длины B (без наложений). Используя операцию деления нацело, найти количество отрезков B , размещенных на отрезке AB .
2. Дано трехзначное число. Найти сумму и произведение его цифр.
3. Дано трехзначное число. Вывести число, полученное при прочтении исходного числа справа налево.
4. Дано трехзначное число. В нем зачеркнули первую слева цифру и приписали ее справа. Вывести полученное число.
5. Дано трехзначное число. В нем зачеркнули первую справа цифру и приписали ее слева. Вывести полученное число.
6. Дано трехзначное число. Вывести число, полученное при перестановке цифр сотен и десятков исходного числа (например, 123 перейдет в 213).
7. Дано целое число, большее 999. Используя только целочисленные операции, найти цифру, соответствующую разряду сотен в записи этого числа.
8. Даны целые положительные числа A , B , C . На прямоугольнике размера $A * B$ размещено максимально возможное количество квадратов со стороной C (без наложений). Найти количество квадратов, размещенных на прямоугольнике, а также площадь незанятой части прямоугольника.
9. Дан номер некоторого года (целое положительное число). Определить соответствующий ему номер столетия, учитывая, что, к примеру, началом 20 столетия был 1901 год.
10. Дано трехзначное число. Вывести число, полученное при перестановке цифр десятков и единиц

Задание 3.3

Задачи на использование логических операторов, операторов отношения. Во всех заданиях данной группы требуется вывести логическое значение True, если приведенное высказывание для предложенных исходных данных является истинным, и значение False в противном случае. Все числа, для которых указано количество цифр (двузначное число, трехзначное число и т. д.), считаются целыми положительными. Не требуется выполнять проверку введенных пользователем данных. Использование IF и оператора "?" недопустимо.

Логические операции делятся на две категории: одни выполняются над

логическими значениями операндов, другие осуществляют выполнение логической операции над битами операндов. По этой причине в C# существуют две унарные операции отрицания - логическое отрицание, заданное операцией «!», и побитовое отрицание, заданное операцией «~». Первая из них определена над операндом типа bool, вторая - над операндом целочисленного типа, начиная с типа int и выше (int, uint, long, ulong). Результатом операции во втором случае является операнд, в котором каждый бит заменен его дополнением.

Бинарные логические операции «&& - условное И» и «|| - условное ИЛИ» определены только над данными типа bool. Операции называются условными или краткими, поскольку, вычисление второго операнда зависит от уже вычисленного значения первого операнда. Ценность условных логических операций заключается в их эффективности по времени выполнения. Часто они позволяют вычислить логическое выражение, имеющее смысл, но в котором второй операнд не определен. Приведем в качестве примера классическую задачу поиска по образцу в массиве, когда разыскивается элемент с заданным значением (образец). Такой элемент в массиве может быть, а может и не быть. Вот типичное решение этой задачи в упрощенном виде, но передающем суть дела:

```
//Условное And - &&
int[ ] ar = { 1, 2, 3 };
int search = 7;
int i = 0;
while ((i < ar.Length) && (ar[i] != search)) {
    i++;
}
if (i < ar.Length) Console.WriteLine("Образец найден");
else Console.WriteLine("Образец не найден");
```

Если значение переменной search (образца) не совпадает ни с одним из значений элементов массива ar, то последняя проверка условия цикла while будет выполняться при значении i, равном ar.Length. В этом случае первый операнд получит значение false, и, хотя второй операнд при этом не определен, цикл нормально завершит свою работу. Вторым операндом не определен в последней проверке, поскольку индекс элемента массива выходит за допустимые пределы (в C# индексация элементов начинается с нуля).

Три бинарные побитовые операции - «& - AND», «| - OR», «^ - XOR» используются двояко. Они определены как над целыми типами выше int, так и над булевыми типами. В первом случае они используются как побитовые операции, во втором - как обычные логические операции. Иногда необходимо, чтобы оба операнда вычислялись в любом случае, тогда без этих операций не обойтись. Вот пример первого их использования:

```
//Логические побитовые операции And, Or, XOR (&,&,^)
```

```
int k2 = 7, k3 = 5, k4, k5, k6;  
k4 = k2 & k3;  
k5 = k2 | k3;  
k6 = k2 ^ k3;  
Console.WriteLine("k4 = " + k4 + " k5 = " + k5 + " k6 = " + k6);
```

Результаты вывода:

k4 = 5 k5 = 7 k6 = 2

Приведем пример поиска по образцу с использованием логического AND: **i = 0;**

```
search = ar[ar.Length - 1];  
while ((i < ar.Length) & (ar[i] != search)) i++;  
if (i < ar.Length) Console.WriteLine("Образец найден");  
else cConsole.WriteLine("Образец не найден");
```

В данном фрагменте гарантируется наличие образца поиска в массиве, и фрагмент будет успешно выполнен. В тех же случаях, когда массив не содержит элемента search, будет инициировано исключение. Содержательный смысл такой процедуры - появление исключения - может быть признаком ошибки в данных, что требует специальной обработки ситуации.

Задачи по вариантам

1. Дано целое число А. Проверить истинность высказывания: «Число А является положительным».

2. Дано целое число А. Проверить истинность высказывания: «Число А является нечетным».

3. Дано целое число А. Проверить истинность высказывания: «Число А является четным».

4. Даны два целых числа: А, В. Проверить истинность высказывания: «Справедливы неравенства $A > 2$ и $B \leq 3$ ».

5. Даны два целых числа: А, В. Проверить истинность высказывания: «Справедливы неравенства $A \geq 0$ или $B < -2$ ».

6. Даны три целых числа: А, В, С. Проверить истинность высказывания: «Справедливо двойное неравенство $A < B < C$ ».

7. Даны три целых числа: А, В, С. Проверить истинность высказывания: «Число В находится между числами А и С».

8. Даны два целых числа: А, В. Проверить истинность высказывания: «Каждое из чисел А и В нечетное».

9. Даны два целых числа: А, В. Проверить истинность высказывания: «Хотя бы одно из чисел А и В нечетное».

10. Даны два целых числа: А, В. Проверить истинность высказывания: «Ровно одно из чисел А и В нечетное».

11. Даны два целых числа: А, В. Проверить истинность высказывания:

«Числа А и В имеют одинаковую четность».

12.Даны три целых числа: А, В, С. Проверить истинность высказывания: «Каждое из чисел А, В, С положительное».

13.Даны три целых числа: А, В, С. Проверить истинность высказывания: «Хотя бы одно из чисел А, В, С положительное».

14.Даны три целых числа: А, В, С. Проверить истинность высказывания: «Ровно одно из чисел А, В, С положительное».

15.Даны три целых числа: А, В, С. Проверить истинность высказывания: «Ровно два из чисел А, В, С являются положительными».

16.Дано целое положительное число. Проверить истинность высказывания: «Данное число является четным двузначным».

17.Дано целое положительное число. Проверить истинность высказывания: «Данное число является нечетным трехзначным».

18.Проверить истинность высказывания: «Среди трех данных целых чисел есть хотя бы одна пара совпадающих».

19.Проверить истинность высказывания: «Среди трех данных целых чисел есть хотя бы одна пара взаимно противоположных».

20.Дано трехзначное число. Проверить истинность высказывания: «Все цифры данного числа различны».

ПРИМЕР ВЫПОЛНЕНИЯ КОМПЛЕКСНОГО ЗАДАНИЯ 3.1-3.3

1. Даны два круга с общим центром и радиусами R_1 и R_2 ($R_1 > R_2$). Найти площади этих кругов S_1 и S_2 , а также площадь S_3 кольца, внешний радиус которого равен R_1 , а внутренний радиус равен R_2 :

2. Даны координаты двух различных полей шахматной доски x_1, y_1, x_2, y_2 (целые числа, лежащие в диапазоне 1–8). Проверить истинность высказывания: «Ладья за один ход может перейти с одного поля на другое».

3. Дано трехзначное число. Найти сумму и произведение его цифр.

```
using System;
using System.Collections.Generic;
using System.Text;
using System;
using System.Collections.Generic;
using System.Text;

namespace MathTask
{
    class Program
    {
        #region Переменные круга
        private double r1;
        private double r2;
        private double s1;
        private double s2;
        private double s3;
        #endregion

        #region Переменные задачи о ладье

        private byte x1;
        private byte y1;
        private byte x2;
        private byte y2;

        #endregion

        #region Переменные задачи о числе

        private string value;
        private byte a;
        private byte b;
```

```

private byte c;
private int result;
#endregion
static void Main(string[] args)
{
    Program ex = new Program();
    ex.FirstTask();
    ex.SecondTask();
    ex.ThirdTask();
    Console.ReadLine();
}

private void FirstTask()
{
    do
    {
        Console.WriteLine("Введите радиус 1-го круга:");
        this.r1 = Convert.ToDouble(Console.ReadLine());
        Console.WriteLine("Введите радиус 2-го круга:");
        this.r2 = Convert.ToDouble(Console.ReadLine());
        if (r1 <= r2)
        {
            Console.WriteLine("Радиус первого круга должен быть
больше второго!");
        }
    } while (r1 <= r2);

    this.s1 = Math.PI * Math.Pow(r1, 2);
    this.s2 = Math.PI * Math.Pow(r2, 2);

    this.s3 = s1 - s2;

    Console.WriteLine("Площадь первого круга = {0}", s1);
    Console.WriteLine("Площадь второго круга = {0}", s2);
    Console.WriteLine("Площадь кольца между ними = {0}", s3);
}

private void SecondTask()
{
    //Проверить может ли попасть ладья за один ход из поля
заданного
    //координатами x1,y1 в поле заданное координатами x2,y2
    Console.WriteLine("Введите вертикаль на которой находится
ладья:");
    this.x1 = Convert.ToByte(Console.ReadLine());

```

```

        Console.WriteLine("Введите горизонталь на которой находится
ладья:");
        this.y1 = Convert.ToByte(Console.ReadLine());

        Console.WriteLine("Введите вертикаль поля на которое должна
попасть ладья:");
        this.x2 = Convert.ToByte(Console.ReadLine());
        Console.WriteLine("Введите горизонталь поля на которое должна
попасть ладья:");
        this.y2 = Convert.ToByte(Console.ReadLine());

        if (x1 == x2 || y1 == y2)
        {
            Console.WriteLine("Данный ход ладьи возможен.");
        }
        else
        {
            Console.WriteLine("Данный ход ладьи невозможен!");
        }
    }
    private void ThirdTask()
    {
        //Дано трёхзначное число, найти сумму произведения его цифр
        do
        {
            Console.WriteLine("Введите число:");
            this.value = Console.ReadLine();
        } while (value.Length != 3);

        this.a = byte.Parse(value[0].ToString());
        this.b = byte.Parse(value[1].ToString());
        this.c = byte.Parse(value[2].ToString());

        this.result = (a * b) + (a * c) + (b * c);

        Console.WriteLine("Сумма произведения трёх цифр:{0}",
this.result);
    }
}

```

Содержание отчета

1. Титульный лист по стандартной форме на сброшюрованных листах формата А4 (210х297мм).
2. Постановка задачи.
3. Необходимые математические выкладки (и/или поясняющая картинка).
4. Алгоритм решения задачи (в виде блок-схемы или в текстуальной форме).
5. Тексты программы.
6. Тестовые наборы (тест – это проверка работоспособности программы по контрольным значениям данных).
7. Результаты вычислений по каждому тесту.
8. Список использованной литературы и Интернет-ресурсов (обязательно).

Отчет для лабораторной работы составляется в одном экземпляре и подлежит защите. Для защиты лабораторной работы студент должен подготовиться к ответу на **Контрольные вопросы**.

Вопросы для защиты работы

1. В чем особенности формализованного языка?
2. Что понимают под вычислительным алгоритмом?
3. Какие требования предъявляются к алгоритмам?
4. В чём суть основных этапов подготовки и компьютерного решения задач?
5. Общие правила построения программ.
6. Как получают исполняемую программу?
7. Какие операции можно выполнять над величинами целого типа? Укажите приоритет их выполнения при расчете значения арифметического выражения.
8. Как определить остаток от деления одной величины целого типа на другую?
9. Как оформляется оператор вывода результатов на экран? Что можно указывать в качестве элементов списка вывода? Какой символ используется для разделения элементов списка вывода? Как должен быть оформлен оператор вывода, чтобы информация выводилась на экран с новой строки?
10. Как оформляется оператор ввода? Что можно указывать в качестве элементов списка ввода? Как работает оператор ввода (что происходит при его выполнении)?
11. Как оформляется оператор присваивания? Как он работает (что происходит при его выполнении)?
12. Как проверить, правильно ли работает программа, в которой проводятся какие-то вычисления?

