МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ГЛАВНОЕ УПРАВЛЕНИЕ ПО ОБРАЗОВАНИЮ

МОГИЛЕВСКОГО ОБЛАСТНОГО ИСПОЛНИТЕЛЬНОГО КОМИТЕТА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

|  |  |
| --- | --- |
| Специальность | 2– 40 01 01 |
| Учебная группа | ПО-455 |
|  |  |

Учебная дисциплина Конструирование программ

и языки программирования

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

**РАЗРАБОТКА ПРОГРАММ С ИСПОЛЬЗОВАНИЕМ ОПЕРАТОРОВ ВЫБОРА, ЦИКЛА И ПЕРЕДАЧИ УПРАВЛЕНИЯ**

Выполнил Савич А.О.

Проверил Карманов А.В.

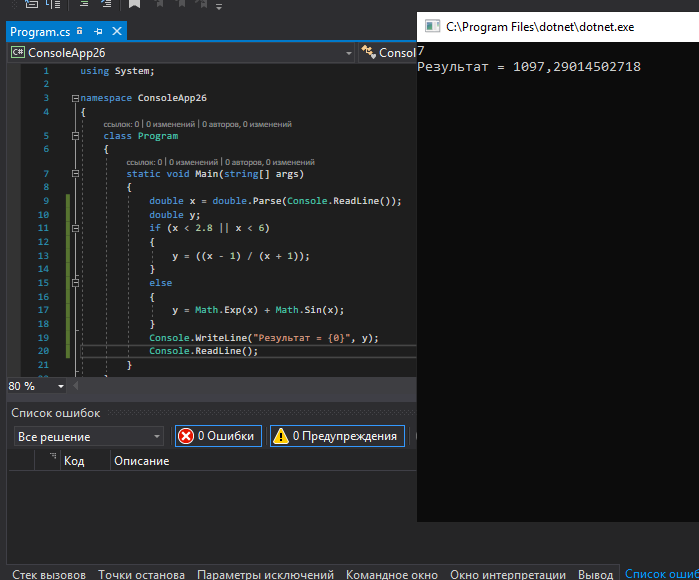
2021

1. Цель работы
   1. Формирование умений применять операторы ветвления и цикла
   2. Формирование умений разрабатывать программы с использованием операторов выбора, цикла, передачи управления.
2. Индивидуальное задание

Вариант 8

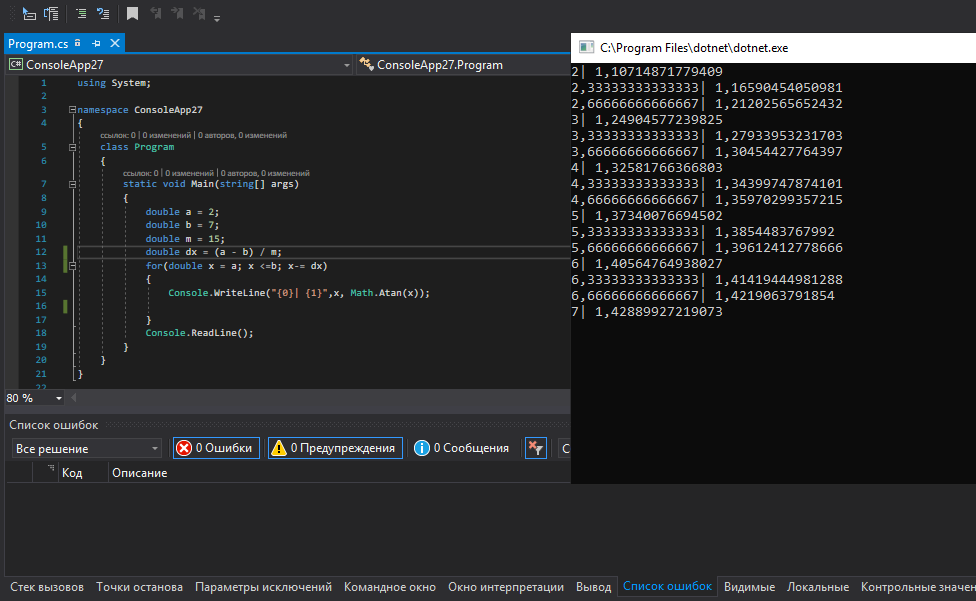
**Задание 1.** Написать программу, которая вычисляет значение функции *у*:





**Задание 2.** Написать программу, которая вычисляет значение функции в промежутке от А до В с числом разбиений М. Функцию, параметры А, В, М взять из приведенной ниже таблицы согласно варианта

**C:\Users\User\AppData\Local\Microsoft\Windows\INetCache\Content.Word\у2йуас12уасв1.jpg**

****

1. Контрольные вопросы

1 Приведите синтаксис и опишите алгоритм выполнения оператора ветвления

Операторы ветвления

Операторы ветвления позволяют изменить порядок выполнения операторов в программе. К операторам ветвления относятся условный оператор if и оператор выбора switch.

Условный оператор if

Условный оператор if используется для разветвления процесса обработки данных на два направления. Он может иметь одну из форм: сокращенную или полную.

Форма сокращенного оператора if:

if (B) S;

где В - логическое или арифметическое выражение, истинность которого проверяется; S - оператор: простой или составной.

При выполнении сокращенной формы оператора if сначала вычисляется выражение B, затем проводится анализ его результата: если B истинно, то выполняется оператор S ; если B ложно, то оператор S пропускается. Таким образом, с помощью сокращенной формы оператора if можно либо выполнить оператор S, либо пропустить его.

Форма полного оператора if:

if (B) S1; else S2;

где B - логическое или арифметическое выражение, истинность которого проверяется; S1, S2 - оператор: простой или составной.

При выполнении полной формы оператора if сначала вычисляется выражение B, затем анализируется его результат: если B истинно, то выполняется оператор S1, а оператор S2 пропускается; если B ложно, то выполняется оператор S2, а S1 - пропускается. Таким образом, с помощью полной формы оператора if можно выбрать одно из двух альтернативных действий процесса обработки данных.

Рассмотрим несколько примеров записи условного оператора if:

if (a > 0) x=y; // Сокращенная форма c простым оператором

if (++i) {x=y; y=2\*z;} // Сокращенная форма c составным оператором

if (a > 0 || b<0) x=y; else x=z; // Полная форма с простым оператором

if (i+j-1) { x= 0; y= 1;} else {x=1; y:=0;} // Полная форма с составными операторами

Рассмотрим пример использования условного оператора.

static void Main()

{

Console.Write("x= ");

float x = float.Parse(Console.ReadLine());

Console.Write("y=");

float y = float.Parse(Console.ReadLine());

if (x < y ) Console.WriteLine("min= "+x);

else Console.WriteLine("min= "+y);

}

2 Приведите синтаксис и опишите алгоритм выполнения оператора выбора

Оператор выбора switch

Оператор выбора switch предназначен для разветвления процесса вычислений по нескольким направлениям. Формат оператора:

switch ( <выражение> )

{

case <константное\_выражение\_1>:

[<оператор 1>]; <оператор перехода>;

case <константное\_выражение\_2>:

[<оператор 2>]; <оператор перехода>;

...

case <константное\_выражение\_n>:

[<оператор n>]; <оператор перехода>;

[default: <оператор>; ]

}

Замечание. Выражение, записанное в квадратных скобках, является необязательным элементом в операторе switch. Если оно отсутствует, то может отсутствовать и оператор перехода.

Выражение, стоящее за ключевым словом switch, должно иметь арифметический, символьный, строковый тип или тип указатель. Все константные выражения должны иметь разные значения, но их тип должен совпадать с типом выражения, стоящим после switch или приводиться к нему. Ключевое слово case и расположенное после него константное выражение называют также меткой case.

Выполнение оператора начинается с вычисления выражения, расположенного за ключевым словом switch. Полученный результат сравнивается с меткой case. Если результат выражения соответствует метке case, то выполняется оператор, стоящий после этой метки, за которым обязательно должен следовать оператор перехода: break, goto и т.д. При использовании оператора break происходит выход из switch и управление передается оператору, следующему за switch. Если же используется оператор goto, то управление передается оператору, помеченному меткой, стоящей после goto.

Пример. По заданному виду арифметической операции (сложение, вычитание, умножение и деление) и двум операндам, вывести на экран результат применения данной операции к операндам.

static void Main()

{

Console.Write("OPER= ");

char oper=char.Parse(Console.ReadLine());

bool ok=true;

Console.Write("A= ");

int a=int.Parse(Console.ReadLine());

Console.Write("B= ");

int b=int.Parse(Console.ReadLine());

float res=0;

switch (oper)

{

case '+': res = a + b; break; //1

case '-': res = a - b; break;

case '\*': res = a \* b; break;

case ':': if (b != 0)

{

res = (float)a / b; break;

}

else goto default;

default: ok = false; break;

}

if (ok) Console.WriteLine("{0} {1} {2} = {3}", a, oper, b, res);

else Console.WriteLine("error");

}

3 Приведите синтаксис и опишите алгоритм выполнения цикла с параметром

Оператор for выполняет оператор или блок операторов, пока определенное логическое выражение равно значению true. В следующем примере показана инструкция for, выполняющая тело пока целочисленный счетчик меньше трех:

C#

Выполнить

for (int i = 0; i < 3; i++)

{

Console.Write(i);

}

// Output:

// 012

В предыдущем примере показаны элементы оператора for:

Раздел инициализатора, который выполняется только один раз перед входом в цикл. Как правило, в этом разделе объявляется и инициализируется локальная переменная цикла. Доступ к объявленной переменной извне оператора for невозможен.

В разделе инициализатора в предыдущем примере объявляется и инициализируется целочисленная переменная-счетчик:

C#

int i = 0

Раздел условия, в котором определяется, следует ли выполнять следующую итерацию в цикле. Если для него получено значение true или значение отсутствует, выполняется следующая итерация; в противном случае цикл завершается. Раздел условия должен быть логическим выражением.

В разделе условия в предыдущем примере проверяется, меньше ли трех значение счетчика.

C#

i < 3

Раздел итератора, который определяет, что происходит после каждого выполнения тела цикла.

Раздел итератора в предыдущем примере увеличивает значение счетчика:

C#

i++

Тело цикла которое должно быть оператором или блоком операторов.

Раздел итератора может содержать ноль или более следующих выражений оператора, разделенных запятыми:

префиксное или постфиксное выражение приращения, такое как ++i или i++

префиксное или постфиксное выражение декремента, такое как --i или i--

присваивание

вызов метода

выражение await

создание объекта с помощью оператора new

Если переменная цикла не объявлена в разделе инициализатора, в разделе инициализатора можно также использовать ноль или более выражений из предыдущего списка. В следующем примере показано несколько менее распространенных вариантов использования разделов инициализатора и итератора: присваивание значения внешней переменной цикла в разделе инициализатора, вызов метода в разделах инициализатора и итератора и изменение значения двух переменных в разделе итератора.

Все разделы оператора for необязательны.

4 Приведите синтаксис и опишите алгоритм выполнения цикла с предусловием

Цикл с предусловием — цикл, который выполняется пока истинно некоторое условие, указанное перед его началом. Это условие проверяется до выполнения тела цикла, поэтому тело может быть не выполнено ни разу (если условие с самого начала ложно).

Цикл называется итерационным, если число его повторений не задается, а определяется в ходе выполнения цикла. В этом случае одно повторение цикла называется итерацией.

Пример синтаксиса

while (условие)

{

действия цикла

}

5 Приведите синтаксис и опишите алгоритм выполнения цикла с постусловием

В цикле с предусловием предварительной проверкой определяется, выполнять тело цикла или нет, до первой итерации. Если это не соответствует логике алгоритма, то можно использовать цикл с постусловием.

икл с постусловием реализован конструкцией

do оператор while (условие);

здесь условие — логическое или целочисленное выражение, оператор — любой оператор языка С(С++). Если тело цикла состоит более чем из одного оператора:

do

{

оператор\_1;

оператор\_2;

...

оператор\_n;

}

while (условие);

Работает цикл следующим образом. В начале выполняется оператор, представляющий собой тело цикла. Затем вычисляется условие. Если оно истинно (не равно нулю), оператор тела цикла выполняется ещё раз. В противном случае цикл завершается, и управление передаётся оператору, следующему за циклом.

Таким образом, не трудно заметить, что цикл с постусловием всегда будет выполнен хотя бы один раз, в отличие от цикла с предусловием, который может не выполниться ни разу.

6 Приведите синтаксис и опишите алгоритм выполнения бесконечного цикла

Бесконечный цикл – это цикл, написанный таким образом, что он никогда не прекратит своё выполнение, так как никогда не выполнится условие выхода из этого цикла.

В целом, бесконечные циклы считаются плохим тоном и почти всегда являются следствием неправильно написанной программы. Однако случаются моменты, когда такие циклы бывают довольно полезны. Например, если, необходимо, чтобы программа не закрывалась или чтобы из неё не было выхода. Операционные системы и некоторые программы микроконтроллеров зачастую также представляют собой бесконечные циклы.

В большинстве языках программирования создать бесконечные циклы можно тремя способами – при помощи циклов for, while и do…while.

Создать бесконечный цикл крайне просто. Например для того, чтобы сделать бесконечным цикл for, необходимо просто оставить пустые поля в скобках при нём: for ( ; ; ).

Давайте посмотрим, как это выглядит в программе. Для начала мы объявим и инициализируем переменную i нулем. С неё будет вестись отсчёт чисел. Затем идёт цикл for без условий в скобках. Внутри цикла мы сделаем инкремент переменной i, иными словами – прибавим к переменной единицу. Ну и, наконец, так же в цикле будем выводить значение переменной i в консоль.

namespace infinite\_loop

{

class Program

{

static void Main(string[] args)

{

int i = 0;

for (; ; )

{

i++;

Console.WriteLine(i);

}

}

}

}

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

namespace infinite\_loop

{

class Program

{

static void Main(string[] args)

{

int i = 0;

for (; ; )

{

i++;

Console.WriteLine(i);

}

}

}

}

Итого у нас получается следующий процесс выполнения программы: инициализация i=0, вход в цикл for, проверка условий for, отсутствие в цикле условий, переход к операциям внутри цикла – прибавка к i единицы (i становится равно 1), вывод i в консоль, возвращение к проверке условий for, отсутствие в цикле условий, снова прибавка к i единицы (i становится равно 2), вывод i в консоль, возвращение к проверке условий for, отсутствие в цикле условий и так далее до бесконечности.

7 Опишите работу логических операций

Следующие операторы выполняют логические операции с использованием логических операндов:

Унарный ! (логическое отрицание) оператор.

Бинарные & (логическое И), | (логическое ИЛИ), а также ^ (логическое исключающее ИЛИ) операторы. Эти операторы всегда обрабатывают оба операнда.

Бинарные && (условное логическое И) и || (условное логическое ИЛИ) операторы. Эти операторы вычисляют правый операнд, только если это необходимо.

Для операндов целочисленных типов операторы &, | и ^ выполняют побитовые логические операции. Дополнительные сведения см. в разделе Побитовые операторы и операторы сдвига.

8 Объясните понятия «итерация цикла», «счетчик цикла», «параметр цикла»

— тело цикла — набор команд, которые будут повторяться. Когда говорится «команда написана внутри цикла», то имеют в виду именно тело.

— итерация цикла — однократное выполнение тела цикла.

— параметр цикла — логическое выражение, от которого зависит, будет ли выполняться ещё одна итерация. (Тут возможна путаница с конструкциями ветвления)

Счетчик цикла — это целочисленная переменная, которая объявляется с единственной целью: считать, сколько раз выполнился цикл. В вышеприведенных примерах переменная count является счетчиком цикла.