**Лабораторная работа №2**

**Разработка программ с использованием операторов ветвления**

**Цель работы:**

Познакомиться с операторами ветвления, научиться использовать их в программах с использованием среды **Visual Studio.Net.**

**1. Теоретические сведения**

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 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 выражение, истинность которого проверяется; S1, S2- оператор:

простой или составной.

При выполнении полной формы оператора if сначала вычисляется значение выражения 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>0) {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

Операторы S1 и S2 могут также являться операторами if. Такие операторы называют вложенными. При этом ключевое слово else связывается с ближайшим предыдущим словом if, которое еще не связано ни с одним else.

**Пример 1.** Найдем наибольшее значение из трех вещественных чисел:

static void Main()

{

Console.Write("x= ");

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

Console.Write("y=");

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

Console.Write("z=");

double z = double.Parse(Console.ReadLine());

double max;

if (x > y && x>z)

{

max=x;

}

else

{

if (y>z)

{

max=y;

}

else

{

max=z;

}

}

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

}

***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>; ]

}

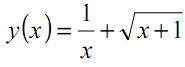
Выражение, стоящее за ключевым словом switch, должно иметь арифметический, символьный, строковый тип или тип указатель. Все константные выражения должны иметь разные значения, но их тип должен совпадать с типом выражения, стоящего внутри скобок switch или приводиться к нему. Ключевое слово case и расположенное 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. В случае отсутствия оператора перехода компилятор выдаст сообщении об ошибке. 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.

**2. Примеры решения практических задач**

**1.** Для произвольных значений аргументов вычислить значение функции, заданной

следующим образом:



Если в некоторой точке вычислить значение функции окажется невозможно, то вывести на экран сообщение «функция не определена».

using System;

namespace Example

{

class Program

{

static void Main()

{

Console.Write("x= ");

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

if (x==0 || x<-1)

{

Console.WriteLine("Функция не определена");

}

else

{

double y=1/x+Math.Sqrt(x+1);

Console.WriteLine("y({0:f2})={1:f2}",x,y);

}

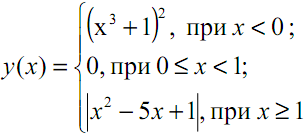
}

}

}

**2.** Для произвольных значений аргументов вычислить значение функции, заданной

следующим образом:



*Указания по решению задачи*. Вся числовая прямая Оx разбивается на три непересекающихся интервала, (-∞; 0), [0; 1), [1; +∞). На каждом интервале функция задается своею ветвью. Заданная точка x может попасть только в один из указанных интервалов.

Чтобы определить, в какой из интервалов попала точка, воспользуемся следующим алгоритмом. Если x<0, то x попадает в первый интервал, и функцию высчитываем по первой ветви, после чего проверка заканчивается. Если это условие ложно, то истинно условие x≥0, и для того чтобы точка попала во второй интервал достаточно, чтобы выполнялось условие x<1. Если выполняется это условие, то точка x попадает во второй интервал, и мы определяем функцию по второй ветви, после чего заканчиваем вычисления. В противном случае, точка может принадлежать только третьему интервалу, поэтому дополнительная проверка не проводится, а сразу вычисляем функцию по третьей ветви. Приведенный алгоритм можно реализовать с помощью вложенных операторов if.

using System;

namespace Example

{

class Program

{

static void Main()

{

Console.Write("x= ");

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

double y;

if (x<0) //проверяем условие первой ветви

{

y=Math.Pow(Math.Pow(x,3)+1,2);

}

else

{

if (x<1) //проверяем условие второй ветви

{

y=0;

}

else

{

y=Math.Abs(x\*x-5\*x+1);

}

}

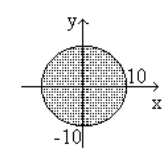
Console.WriteLine("y({0:f2})={1:f2}",x,y);

}

}

}

**3.** Дана точка на плоскости с координатами (х, у). Составить программу, которая выдает одно из сообщений «Да», «Нет», «На границе» в зависимости от того, лежит ли точка внутри заштрихованной области, вне заштрихованной области или на ее границе.



using System;

namespace Example

{

class Program

{

static void Main()

{

Console.Write("x= ");

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

Console.Write("y= ");

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

const int r=10;

if (x\*x+y\*y<r\*r)

{

Console.WriteLine("да");

}

else

{

if (x\*x+y\*y>r\*r)

{

Console.WriteLine("нет");

}

else

{

Console.WriteLine("на границе");

}

}

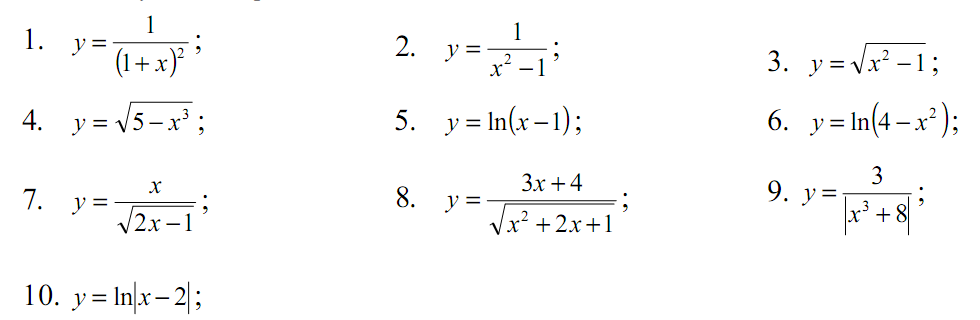
}

}

}

**Задания для самостоятельной работы**

I. Для произвольных значений аргументов вычислить значение функции, заданной следующим образом:



11.



12.



13.



II. Дана точка на плоскости с координатами (х, у). Составить программу, которая выдает одно из сообщений «Да», «Нет», «На границе» в зависимости от того, лежит ли точка внутри заштрихованной области, вне заштрихованной области или на ее границе. Области задаются графически следующим образом:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 |  | 2 |  |
| 3 |  | 4 |  |
| 5 |  | 6 |  |
| 7 |  | 8 |  |
| 9 |  | 10 |  |
| 11 |  | 12 |  |
| 13 |  | 14 |  |

III. Составить программу.

*Замечание*. При решении данных задач можно использовать как оператор switch, так и вложенные операторы if. Свой выбор обоснуйте.

1) Дан порядковый номер месяца, вывести на экран его название.

2) Дан номер масти m (1 ≤ m ≤ 4), определить название масти. Масти нумеруются: «пики» - 1, «трефы» - 2, «бубны» - 3, «червы» - 4.

3) Дан номер карты k (6 ≤ k ≤ 14), определить достоинство карты. Достоинства определяются по следующему правилу: «туз» - 14, «король» - 13, «дама» - 12, «валет» - 11, «десятка» - 10, …, «шестерка» - 6.

4) Дан номер масти m (1 ≤ m ≤ 4) и номер достоинства карты k (6 ≤ k ≤ 14). Определить полное название соответствующей карты в виде «дама пик», «шестерка бубен» и т.д.

5) Дано расписание приемных часов врача. Вывести на экран приемные часы врача в заданный день недели (расписание придумать самостоятельно).

6) Проведен тест, оцениваемый в целочисленный баллах от нуля до ста. Вывести на экран оценку тестируемого в зависимости от набранного количества баллов: от 90 до 100 – «отлично», от 70 до 89 – «хорошо», от 50 до 69 – «удовлетворительно», менее 50 – «неудовлетворительно».

7) Дан год. Вывести на экран название животного, символизирующего этот год по восточному календарю.

8) Дан возраст человека мужского пола в годах. Вывести на экран возрастную категорию: до года – «младенец», от года до 11 лет – «ребенок», от 12 до 15 лет – «подросток», от 16 до 25 лет – «юноша», от 26 до 70 лет – «мужчина», более 70 лет – «пожилой человек».

9) Дан пол человека: м – мужчина, ж – женщина. Вывести на экран возможные мужские и женские имена в зависимости от введенного пола.

10) Дан признак транспортного средства: a – автомобиль, в – велосипед, м – мотоцикл, с – самолет, п – поезд. Вывести на экран максимальную скорость транспортного средства в зависимости от введенного признака.