# Лабораторная работа № 2 Разработка программ разветвляющейся структуры и Организация циклов

**Цель лабораторной работы**

Закрепление теоретических знаний по разработке программ нелинейной структуры с применением операторов if, else, switch и циклов for, while, do...while.

**Постановка задачи**

**Условные конструкции**

Условные конструкции - один из базовых компонентов многих языков программирования, которые направляют работу программы по одному из путей в зависимости от определенных условий.

В языке C# используются следующие условные конструкции: if..else и switch..case

**Конструкция if/else**

Конструкция if/else проверяет истинность некоторого условия и в зависимости от результатов проверки выполняет определенный код:

int num1 = 8;

int num2 = 6;

if(num1 > num2)

{

Console.WriteLine($"Число {num1} больше числа {num2}");

}

После ключевого слова if ставится условие. И если это условие выполняется, то срабатывает код, который помещен далее в блоке if после фигурных скобок. В качестве условий выступают ранее рассмотренные операции сравнения.

В данном случае у нас первое число больше второго, поэтому выражение num1 > num2 истинно и возвращает true, следовательно, управление переходит к строке Console.WriteLine("Число {num1} больше числа {num2}");

Но что, если мы захотим, чтобы при несоблюдении условия также выполнялись какие-либо действия? В этом случае мы можем добавить блок else:

int num1 = 8;

int num2 = 6;

if(num1 > num2)

{

Console.WriteLine($"Число {num1} больше числа {num2}");

}

else

{

Console.WriteLine($"Число {num1} меньше числа {num2}");

}

Но при сравнении чисел мы можем насчитать три состояния: первое число больше второго, первое число меньше второго и числа равны. Используя конструкцию else if, мы можем обрабатывать дополнительные условия:

int num1 = 8;

int num2 = 6;

if(num1 > num2)

{

Console.WriteLine($"Число {num1} больше числа {num2}");

}

else if (num1 < num2)

{

Console.WriteLine($"Число {num1} меньше числа {num2}");

}

else

{

Console.WriteLine("Число num1 равно числу num2");

}

Также мы можем соединить сразу несколько условий, используя логические операторы:

int num1 = 8;

int num2 = 6;

if(num1 > num2 && num1==8)

{

Console.WriteLine($"Число {num1} больше числа {num2}");

}

В данном случае блок if будет выполняться, если num1 > num2 равно true и num1==8 равно true.

**Конструкция switch**

Конструкция **switch/case** аналогична конструкции if/else, так как позволяет обработать сразу несколько условий:

Console.WriteLine("Нажмите Y или N");

string selection = Console.ReadLine();

switch (selection)

{

case "Y":

Console.WriteLine("Вы нажали букву Y");

break;

case "N":

Console.WriteLine("Вы нажали букву N");

break;

default:

Console.WriteLine("Вы нажали неизвестную букву");

break;

}

После ключевого слова **switch** в скобках идет сравниваемое выражение. Значение этого выражения последовательно сравнивается со значениями, помещенными после оператора **сase**. И если совпадение будет найдено, то будет выполняться определенный блок **сase**.

В конце каждого блока сase должен ставиться один из операторов перехода: **break**, **return** или **throw**. Как правило, используется оператор **break**. При его применении другие блоки case выполняться не будут.

int number = 1;

switch (number)

{

case 1:

Console.WriteLine("case 1");

case 3:

Console.WriteLine("case 3");

break;

case 5:

Console.WriteLine("case 5");

break;

default:

Console.WriteLine("default");

break;

}

Если мы хотим также обработать ситуацию, когда совпадения не будет найдено, то можно добавить блок **default**, как в примере выше.

Применение оператора return позволит выйти не только из блока case, но и из вызывающего метода. То есть, если в методе Main после конструкции switch..case, в которой используется оператор return, идут какие-либо операторы и выражения, то они выполняться не будут, а метод Main завершит работу.

Оператор throw применяется для выброса ошибок и будет рассмотрен в одной из следующим тем.

**Тернарная операция**

Тернарную операция имеет следующий синтаксис: [первый операнд - условие] ? [второй операнд] : [третий операнд]. Здесь сразу три операнда. В зависимости от условия тернарная операция возвращает второй или третий операнд: если условие равно true, то возвращается второй операнд; если условие равно false, то третий. Например:

int x=3;

int y=2;

Console.WriteLine("Нажмите + или -");

string selection = Console.ReadLine();

int z = selection=="+"? (x+y) : (x-y);

Console.WriteLine(z);

Здесь результатом тернарной операции является переменная z. Если мы выше вводим "+", то z будет равно второму операнду - (x+y). Иначе z будет равно третьему операнду.

**Циклы**

Циклы являются управляющими конструкциями, позволяя в зависимости от определенных условий выполнять некоторое действие множество раз. В C# имеются следующие виды циклов:

* for
* foreach
* while
* do...while

**Цикл for**

Цикл for имеет следующее формальное определение:

for ([инициализация счетчика]; [условие]; [изменение счетчика])

{

// действия

}

Рассмотрим стандартный цикл for:

for (int i = 0; i < 9; i++)

{

Console.WriteLine($"Квадрат числа {i} равен {i\*i}");

}

Первая часть объявления цикла - int i = 0 - создает и инициализирует счетчик i. Счетчик необязательно должен представлять тип **int**. Это может быть и другой числовой тип, например, float. И перед выполнением цикла его значение будет равно 0. В данном случае это то же самое, что и объявление переменной.

Вторая часть - условие, при котором будет выполняться цикл. Пока условное выражение возвращает true, будет выполняться цикл. В данном случае цикл будет выполняться, пока счетчик i не достигнет 9.

И третья часть - приращение счетчика на единицу. Опять же нам необязательно увеличивать на единицу. Можно уменьшать: i--.

В итоге блок цикла сработает 9 раз, пока значение i не станет равным 9. И каждый раз это значение будет увеличиваться на 1.

Нам необязательно указывать все условия при объявлении цикла. Например, мы можем написать так:

int i = 0;

for (; ;)

{

Console.WriteLine($"Квадрат числа {++i} равен {i \* i}");

}

Формально определение цикла осталось тем же, только теперь блоки в определении у нас пустые: for (; i <;). У нас нет инициализированной переменной-счетчика, нет условия, поэтому цикл будет работать вечно - бесконечный цикл.

Мы также можем опустить ряд блоков:

int i = 0;

for (; i<9;)

{

Console.WriteLine($"Квадрат числа {++i} равен {i \* i}");

}

Этот пример по сути эквивалентен первому примеру: у нас также есть счетчик, только создан он вне цикла. У нас есть условие выполнения цикла. И есть приращение счетчика уже в самом блоке for.

**Цикл do**

В цикле do сначала выполняется код цикла, а потом происходит проверка условия в инструкции while. И пока это условие истинно, цикл повторяется. Например:

int i = 6;

do

{

Console.WriteLine(i);

i--;

}

while (i > 0);

Здесь код цикла сработает 6 раз, пока i не станет равным нулю. Но важно отметить, что цикл do гарантирует хотя бы единократное выполнение действий, даже если условие в инструкции while не будет истинно. То есть мы можем написать:

int i = -1;

do

{

Console.WriteLine(i);

i--;

}

while (i > 0);

Хотя у нас переменная i меньше 0, цикл все равно один раз выполнится.

**Цикл while**

В отличие от цикла do цикл **while** сразу проверяет истинность некоторого условия, и если условие истинно, то код цикла выполняется:

int i = 6;

while (i > 0)

{

Console.WriteLine(i);

i--;

}

**Операторы continue и break**

Иногда возникает ситуация, когда требуется выйти из цикла, не дожидаясь его завершения. В этом случае мы можем воспользоваться оператором **break**.

Например:

for (int i = 0; i < 9; i++)

{

if (i == 5)

break;

Console.WriteLine(i);

}

Хотя в условии цикла сказано, что цикл будет выполняться, пока счетчик i не достигнет значения 9, в реальности цикл сработает 5 раз. Так как при достижении счетчиком i значения 5, сработает оператор break, и цикл завершится.

Теперь поставим себе другую задачу. А что если мы хотим, чтобы при проверке цикл не завершался, а просто пропускал текущую итерацию. Для этого мы можем воспользоваться оператором **continue**:

for (int i = 0; i < 9; i++)

{

if (i == 5)

continue;

Console.WriteLine(i);

}

В этом случае цикл, когда дойдет до числа 5, которое не удовлетворяет условию проверки, просто пропустит это число и перейдет к следующей итерации:

**Цикл foreach**

Цикл foreach перебирает коллекции, например, массивы, и будет рассмотрен далее в теме массивов.

**Задание на лабораторную работу**

1. Разработать консольное приложение.
2. Реализовать возможность выбора задания.
3. Выполнить задания согласно варианту.
4. Подготовить входные данные для проверки работы приложения.

**Варианты заданий на лабораторную работу**

**Задание 1.**

1. Дано действительное а. Для функций f(a), график которой представлен на рисунке 2.1, вычислить f(a).

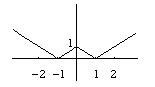


Рисунок 2.1 – График функции для варианта 1

1. Даны действительные положительные числа x, y, z. Выяснить, существует ли треугольник с длинами сторон x, y, z. Если существует, то найти его площадь.
2. Дано действительное а. Для функции f(a), график которой представлен на рисунке 2.2, вычислить f(a).

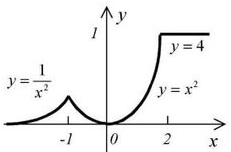


Рисунок 2.2 – График функции для варианта 3

1. Пусть D – заштрихованная часть плоскости, представленная на рисунке 2.3, и пусть u определяется по x и y следующим образом (запись (х, у)єD) означает, что точка с координатами х, у принадлежит D):



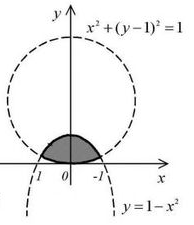


Рисунок 2.3 – График функции для варианта 4

1. Даны три действительных числа. Выбрать из них те, которые принадлежат интервалу (1, 3).
2. Даны действительные числа x, y. Если x, y отрицательны, то каждое значение заменить его модулем; если отрицательное только одно из них, то оба значения увеличить на 0.5; если оба значения не отрицательны и ни одно из них не принадлежит отрезку [0.5, 2.0], то оба значения уменьшить в 10 раз; в остальных случаях x, y оставить без изменения.
3. Определить и вывести на экран номер квадранта, в котором расположена точка М(x,y.z), x, y и z заданные вещественные числа.
4. Из величин, определяемых выражениями a=sinx, b=cosx, c=ln|x| при заданном х, определить и вывести на экран минимальное и максимальное значение.
5. Определить, какая из проекций на координатные плоскости двух точек ‑ M1(x1,y1,z1) или M2(x2,y2,z2) ‑ расположена ближе к началу координат. Вывести на экран дисплея координаты этой точки и длину проекции.
6. Определить, какая из двух фигур (круг или квадрат) имеет большую площадь. Известно, что сторона квадрата равна а, радиус круга r. Вывести на экран название и значение площади большей фигуры.
7. Определить, попадает ли точка M(x,y) в круг радиусом r с центром в точке (x0,y0)
8. Даны действительные числа a, b, c, x, y. Выяснить, пройдет ли кирпич с ребрами a,b,c в прямоугольное отверстие со сторонами x и y. Просовывать кирпич в отверстие разрешается только так, чтобы каждое из его ребер было параллельно или перпендикулярно каждой из сторон отверстия.
9. Сможет ли шар радиуса R пройти в ромбообразное отверстие со стороной P и острым углом Q?
10. Написать программу, которая печатает True или False в зависимости от того, выполняются или нет заданные условия:
11. а) квадрат заданного трехзначного числа равен кубу суммы цифр этого числа;
12. б) сумма двух первых цифр заданного четырехзначного числа равна сумме двух его последних цифр;
13. в) среди цифр заданного трехзначного числа есть одинаковые;
14. г) среди первых трех цифр из дробной части заданного положительного вещественного числа есть цифра 0.
15. Три точки на плоскости заданы своими координатами. Написать
16. программу, определяющую, лежат ли они на одной прямой.
17. Дан шестизначный номер билета. Написать программу, проверяющую, является ли билет «счастливым». (Билет будем считать «счастливым», если сумма первых трёх цифр равна сумме последних трёх цифр.)
18. Стоимость минуты разговора по телефону — 36 руб. Если продолжительность разговора превышает 5 минут, то на оставшуюся часть времени действует скидка, равная 20%. Написать программу, определяющую стоимость разговора заданной продолжительности t.
19. Написать программу, проверяющую, равно ли утроенное произведение цифр заданного двузначного числа ему самому. (Например, число 15 удовлетворяет этому условию).
20. Даны радиус окружности и катеты прямоугольного треугольника. Написать программу определяющую, можно ли вписать треугольник в окружность.
21. Написать программу, определяющую, является ли заданное четырёхзначное число палиндромом. (Число-палиндром — это число, запись которого слева направо совпадает с записью справа налево.)
22. Дано целое число n — расстояние в метрах. Написать программу, выводящую на экран это расстояние с указанием единицы измерения в нужной форме. Например: «31 метр», «52 метра», «15 метров».
23. Две окружности заданы координатами центров и радиусами. Написать программу, определяющую, пересекаются ли они в точности в одной точке.
24. Две точки заданы на плоскости своими координатами, которые могут быть как декартовыми, так и полярными. Требуется вычислить расстояние между этими двумя точками.
25. Задано число. Определить для четырехзначного числа равна ли сумма двух первых цифр сумме двух его последних цифр, а для трехзначного числа есть ли среди цифр заданного числа одинаковые.
26. Даны три стороны треугольника. Написать программу, определяющую, является ли он остроугольным, тупоугольным, прямоугольным.

**Задание 2.**

1. На промежутке от 1 до  найти все числа Армстронга. Натуральное число из цифр называется числом Армстронга, если сумма его цифр, возведенных в -ю степень, равна самому числу.
2. Дано натуральное. Вычислить:

.

1. Дано натуральное . Вычислить:

.

1. Дано действительное число, натуральное число . Вычислить:

.

1. Дано действительное число , натуральное число . Вычислить:

.

1. Дано действительное число , натуральное число . Вычислить:

.

1. Дано натуральное . Вычиcлить:

.

1. Дано натуральное . Вычиcлить:

.

1. Вычислить приближенно значение бесконечной суммы (справа от каждой суммы дается ее точное значение, с которым можно сравнить полученный ответ):



1. Вычислить приближенно значение бесконечной суммы (справа от каждой суммы дается ее точное значение, с которым можно сравнить полученный ответ):



1. Вычислить приближенно значение бесконечной суммы (справа от каждой суммы дается ее точное значение, с которым можно сравнить полученный ответ):



1. Найти все двузначные числа, сумма цифр которых не меняется при умножении числа на 2,3,4,5,6,7,8,9.
2. Найти все трехзначные числа, сумма цифр которых равна данному целому числу.
3. Найти все трехзначные числа, средняя цифра которых равна сумме первой и третей цифр.
4. Найти все трехзначные числа, которые можно представить разностью между квадратом числа, образованного первыми двумя цифрами и квадратом третьей цифры.
5. Найти все двузначные числа, сумма квадратов цифр которых делится на 17.
6. Найти все трехзначные числа, представимые в виде сумм факториалов своих цифр.
7. Найти двузначное число, обладающее тем свойством, что куб суммы его цифр равен квадрату самого числа.
8. Найти двузначное число, равное утроенному произведению его цифр.
9. В каких двузначных числах удвоенная сумма цифр равна их произведению?
10. Можно ли заданное натуральное число М представить в виде суммы квадратов двух натуральных чисел? Написать программу решения этой задачи.
11. Определить, является ли заданное число совершенным, т.е. равным сумме всех своих (положительных) делителей, кроме самого этого числа (например, число 6 совершенно: 6=1+2+3).
12. Для заданного натурального числа  и действительных чисел и  вычислить , по итерационной формуле:

.

1. Найти все трехзначные числа, в которых сумма цифр равна произведению этих же цифр.
2. Найти все двузначные числа, сумма цифр которых равна квадрату разности этих цифр.
3. Найти все трехзначные числа, которые являются палиндромами и сумма их цифр четная.
4. Найти все двузначные числа, которые можно представить в виде суммы двух квадратов.
5. Найти все трехзначные числа, для которых разность между первым и последним цифрами равна второй цифре.
6. Найти все трехзначные числа, произведение цифр которых делится на 6.
7. Найти все двузначные числа, сумма цифр которых равна числу, полученному путем их перестановки.

**Пример разработки приложения**

**Задание 1.**

Даны три стороны треугольника. Написать программу, определяющую, является ли он равнобедренным или равносторонним. (Треугольник может относиться к нескольким классам одновременно.)

**Шаг 1. Создание проекта в Visual Studio 2022**

1. Запустите **Visual Studio 2022**
2. Нажмите **"Создать новый проект"**
3. Выберите шаблон **"Консольное приложение"** (C#)
4. Укажите:
   * Имя проекта: TriangleTypeChecker
   * Расположение: выберите папку для сохранения
5. Нажмите **"Создать"**

**Шаг 2. Написание кода программы**

В файле Program.cs запрашиваем у пользователя ввод трех сторон треугольника и выполняем чтение и преобразование ввода в числа с плавающей точкой:

Console.WriteLine("Введите длины трех сторон треугольника:");

Console.Write("Сторона A: ");

double sideA = Convert.ToDouble(Console.ReadLine());

Console.Write("Сторона B: ");

double sideB = Convert.ToDouble(Console.ReadLine());

Console.Write("Сторона C: ");

double sideC = Convert.ToDouble(Console.ReadLine());

**Шаг 3. Пояснение кода**

**1. Ввод данных**

* Console.Write() – выводит текст без перевода строки.
* Convert.ToDouble() – преобразует введенную строку в число с плавающей точкой.
* Значения сохраняются в переменные sideA, sideB, sideC.

**2. Проверка типа треугольника**

**Логика проверки:**

**Равнобедренный треугольник** – хотя бы две стороны равны:

if (sideA == sideB || sideB == sideC || sideA == sideC)

**Равносторонний треугольник** – частный случай равнобедренного, где все три стороны равны:

if (sideA == sideB && sideB == sideC)

**Разносторонний треугольник** – все стороны разные (else-блок).

**3. Завершение программы**

* Console.ReadKey() – ожидает нажатия любой клавиши перед закрытием консоли.

// Проверка на равнобедренный треугольник

if (sideA == sideB || sideB == sideC || sideA == sideC)

{

Console.WriteLine("Треугольник является равнобедренным.");

// Проверка на равносторонний треугольник

if (sideA == sideB && sideB == sideC)

{

Console.WriteLine("Треугольник является равносторон-ним.");

}

}

// Если ни одно из условий не выполнено

else

{

Console.WriteLine("Треугольник является разносторонним.");

}

Ожидание ввода перед закрытием консоли:

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey();

**Шаг 4. Запуск и тестирование программы**

1. Нажмите **F5** для запуска.
2. Введите три стороны треугольника (например, 5, 5, 5).
3. Проверьте вывод:
   * Для 5, 5, 5: "равнобедренный" и "равносторонний"
   * Для 5, 5, 3: только "равнобедренный"
   * Для 3, 4, 5: "разносторонний"

**Задание 2.**

Дано натуральное  и вещественное . Вычислить:

.

Запрашиваем у пользователя ввод натурального числа k:

Console.Write("Введите натуральное число k: ");

int k = Convert.ToInt32(Console.ReadLine());

Запрашиваем у пользователя ввод вещественного числа x:

Console.Write("Введите вещественное число x: ");

double x = Convert.ToDouble(Console.ReadLine());

Запишем цикл для расчета суммы и выведем результат:

// Инициализация переменной для хранения суммы

double sum = 0;

// Вычисление суммы по формуле

for (int j = 0; j <= k; j++)

{

sum += x / Math.Pow(k, x + j);

}

// Вывод результата

Console.WriteLine($"Сумма по формуле: {sum}");

// Ожидание ввода перед закрытием консоли

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey();

**Контрольные вопросы**

1. Что такое оператор ветвления в C#? Приведите примеры.
2. Как работает оператор switch и в чем его преимущества по сравнению с цепочкой if?
3. Что происходит, если в условии if не указать блок else?
4. Как можно использовать логические операторы (например, &&, ||) в условиях ветвлений? Приведите пример.
5. Объясните различия между for, while и do-while циклами. Когда следует использовать каждый из них?
6. Как можно прервать выполнение цикла в C#?
7. Что такое бесконечный цикл?
8. Как можно реализовать вложенные циклы в C#?
9. Что такое оператор return и как он работает в контексте циклов и методов?
10. Как можно использовать цикл foreach для перебора элементов массива или коллекции?