



Синтаксис С#

(ветвления, исключения)

Андрей Голяков

Ветвления

Ветвление — это команда алгоритма, в которой делается выбор, выполнять или не выполнять какую-нибудь группу команд в зависимости от условий.

На алгоритмической схеме ветвление изображают в виде ромба, имеющего один вход и два выхода. Внутри ромба пишется утверждение. В зависимости от истинности утверждения выполняется та или иная ветка кода:



Условный оператор **if...else**

```
// Example of if...else #1
Console.WriteLine("Press any key for analysis:");
char c = Console.ReadKey(true).KeyChar;

if (char.IsLetterOrDigit(c))                // conditional statement
{                                           // block if it is true
    Console.WriteLine("You entered letter or digit!");
}
else
{                                           // block if it is false
    Console.WriteLine("You pressed a strange key...");
}

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```



Ветвления

Важно понимать, что после завершения кода веток программа может опять вернуться к линейному потоку



Условный оператор **if...else**

```
// Example of if...else #2
Console.WriteLine("Enter a number less than 100:");
string strNum = Console.ReadLine();
int num = int.Parse(strNum);
string message;

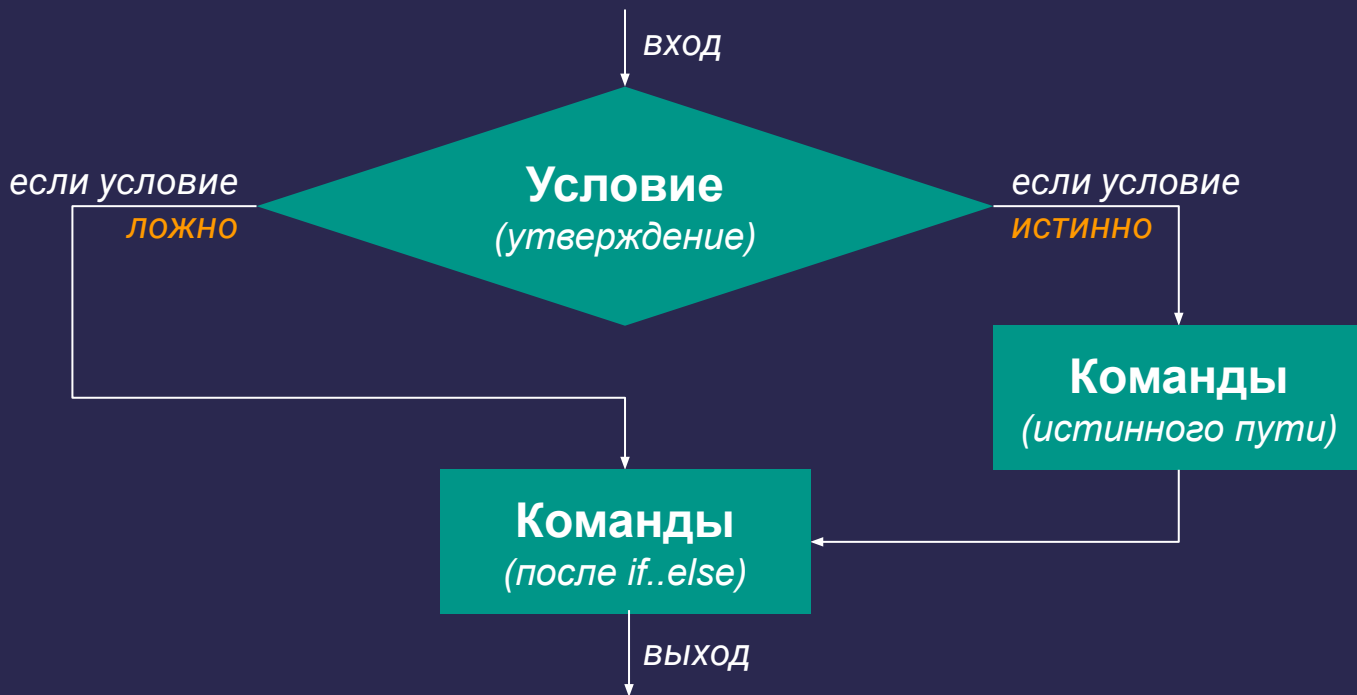
if (num < 100)
{
    message = "Correct!";
}
else
{
    message = "Error!";
}

Console.WriteLine(message + " Press any key to exit...");
Console.ReadKey();
```



Ветвления

Блока **else** может вовсе не быть, в таком случае схема будет выглядеть так:



Самостоятельная работа: **if...else**

Написать программу с блоками ветвления. Программа запрашивает у пользователя количество лет договора аренды в диапазоне от 1 до 30.

Если введено значение за этими пределами, программа выводит сообщение "Вы ввели неверное значение!" и завершается.

Если значение находится в диапазоне [1..30], на экран должна вывестись по-русски грамматически корректная фраза о длительности заключённого договора.

Пример работы программы:

- > Введите длительность договора аренды в годах: 21 /это ввод пользователя/
- > Договор аренды оформлен на период длительностью 21 год
- > Введите длительность договора аренды в годах: 5 /это ввод пользователя/
- > Договор аренды оформлен на период длительностью 5 лет
- > Введите длительность договора аренды в годах: 35 /это ввод пользователя/
- > Вы ввели неверное значение!

Тернарный условный оператор ?:

Терна́рная условная операция (от латинского *ternarius* — “тройной”):

```
// Example of operator ?:  
Console.WriteLine("Enter a number less than 100:");  
string strNum = Console.ReadLine();  
int num = int.Parse(strNum);  
string message;  
  
message = num < 100 // condition statement  
    ? "Correct!"      // value when condition is true  
    : "Error!";      // value when condition is false  
  
Console.WriteLine(message + " Press any key to exit...");  
Console.ReadKey();
```



Самостоятельная работа: ?:

Написать программу с использованием тернарного оператора ?:

Программа запрашивает число от 0 до 100. В зависимости от введенного числа будет выводиться:

- если число меньше 50: “Введенное число меньше 50”
- в если число больше либо равно 50: “Введенное число больше либо равно 50”

Пример работы программы:

> Введите число от 1 до 100: 25 /это ввод пользователя/

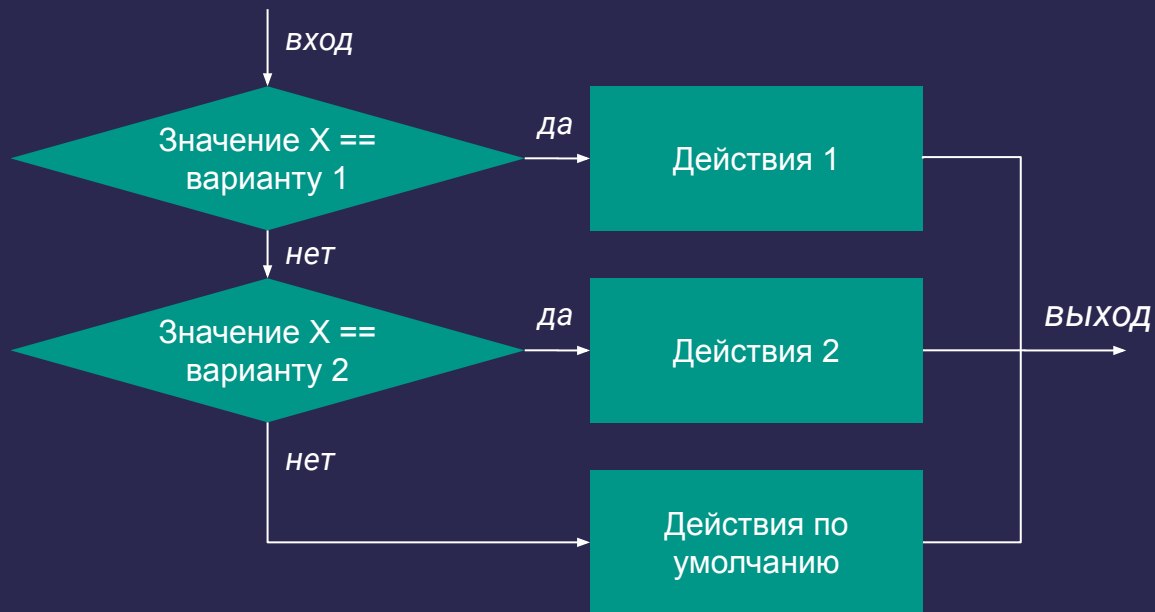
> Введенное число меньше 50

> Введите число от 1 до 100: 73 /это ввод пользователя/

> Введенное число больше либо равно 50

Конструкция **switch**

Конструкция **switch** отличается от **if...else**: она позволяет сравнить выражение с набором возможных значений:



Конструкция **switch**

```
public enum Color { Red, Green, Blue }
public static void Main()
{
    Color c = (Color)(new Random()).Next(0, 3);
    switch (c)
    {
        case Color.Red:
            Console.WriteLine("The color is red");
            break;
        case Color.Blue:
            Console.WriteLine("The color is blue");
            break;
        default:
            Console.WriteLine("The color is unknown.");
            break;
    }
}
```



Конструкция **switch**

Можно объединять команды для нескольких значений, перечисляя несколько блоков **case** подряд один под другим:

```
public enum Color { Red, Green, Blue, Yellow }
public static void Main()
{
    Color c = (Color)(new Random()).Next(0, 4);
    switch (c)
    {
        case Color.Red:
        case Color.Yellow:
            Console.WriteLine("The color is red or yellow");
            break;
        ...
    }
}
```



Самостоятельная работа: switch

Модифицировать программу задания на условие if...else (про длительность договора от 1 до 30 лет) используя для определения правильной формы числительного не if...else, а switch.

Внешне программа не должна отличаться от предыдущей реализации.

Пример работы программы:

- > Введите длительность договора аренды в годах: 21 /это ввод пользователя/
- > Договор аренды оформлен на период длительностью 21 год
- > Введите длительность договора аренды в годах: 5 /это ввод пользователя/
- > Договор аренды оформлен на период длительностью 5 лет
- > Введите длительность договора аренды в годах: 35 /это ввод пользователя/
- > Вы ввели неверное значение!

Исключения

Исключения позволяют обозначить, что во время выполнения программы произошла ошибка.

Объекты исключений, описывающие ошибку, создаются и затем вызываются с помощью ключевого слова **throw**.

Программисты должны вызывать исключения в том случае, если прогнозируется неверное поведение программы.

Объекты исключений наследуются от базового класса **System.Exception**.

Исключения **не рекомендуется** использовать для изменения потока программы в рамках обычного выполнения. Их следует использовать только для сообщения о состояниях ошибки и их обработки.

Генерация собственного исключения

```
Console.WriteLine("Enter a number less than 100:");
string strNum = Console.ReadLine();
int num = int.Parse(strNum);

if (num >= 100)
{
    // throwing new exception according to our logic
    throw new Exception("The value should be less than 100!");
}

Console.WriteLine($"You entered correct value {num}");
Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```



Самостоятельная работа: **throw exception**

Модифицировать программу из задачи на блок switch таким образом, чтобы в случае, если число выходит за диапазон от 1 до 30, генерировалось новое исключение с текстом “Введенное значение выходит за допустимые пределы от 1 до 30”.

Обработка исключений

Функции обработки исключений помогают справиться с непредвиденными или исключительными проблемами, которые возникают при выполнении программы.

Обработка исключений использует ключевые слова `try`, `catch` и `finally` для действий, которые могут оказаться неудачными.

Если исключение перехватывается, его необходимо либо обработать, либо генерировать повторно используя ключевое слово `throw`.

Обработка исключений

```
// for example, we would like to create a simple calculator
Console.WriteLine("Enter integer value A: ");
int a = int.Parse(Console.ReadLine());

Console.WriteLine("Enter integer value B: ");
int b = int.Parse(Console.ReadLine());

int result = a / b;

Console.WriteLine($"{a} divide by {b} equals to {result:##.###}");

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```



Обработка исключений

```
// for example, we would like to create a simple calculator
Console.WriteLine("Enter integer value A: ");
int a = int.Parse(Console.ReadLine());

Console.WriteLine("Enter integer value B: ");
int b = int.Parse(Console.ReadLine());

int result = a / b;

Console.WriteLine($"{a} divide by {b} equals to {result:##.###}");

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```



Обработка исключений

```
try
{
    Console.WriteLine("Enter integer value A: ");
    int a = int.Parse(Console.ReadLine());
    Console.WriteLine("Enter integer value B: ");
    int b = int.Parse(Console.ReadLine());
    int result = a / b;

    Console.WriteLine($"{a} divide by {b} equals to {result}");
}
catch (Exception e) // We can specify exception variable to use it later
{
    Console.WriteLine("Can't continue calculation:");
    Console.WriteLine($"{e.GetType()}: {e.Message}"); // here!
}
Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```



Обработка исключений

```
try
{
    Console.WriteLine("Enter integer value A: ");
    int a = int.Parse(Console.ReadLine());
    Console.WriteLine("Enter integer value B: ");
    int b = int.Parse(Console.ReadLine());
    int result = a / b;

    Console.WriteLine($"{a} divide by {b} equals to {result}");
}
catch // catching ALL exceptions is a bad practice as it may hide problems!
{
    Console.WriteLine("Can't continue calculation! Something goes wrong!");
}

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```



Обработка исключений

```
try
{
    Console.WriteLine("Enter integer value A: ");
    int a = int.Parse(Console.ReadLine());
    Console.WriteLine("Enter integer value B: ");
    int b = int.Parse(Console.ReadLine());
    int result = a / b;
    Console.WriteLine($"{a} divide by {b} equals to {result}");
}
catch (FormatException)           // Cannot parse integer value!
{
    Console.WriteLine("You entered wrong data!");
}
catch (DivideByZeroException e) // Cannot divide by zero!
{
    Console.WriteLine("Cannot divide by zero!");
}
```



Самостоятельная работа: `try...catch`

В предыдущей задаче обернуть в блок `try...catch` код, который запрашивает строку у пользователя и пытается представить его в виде числа с помощью функции `Parse()`.

В блоке `catch` выводить на экран сообщение “Введенная строка не распознаётся как число указанного типа!” и

Сначала (вариант 1):

Корректно завершать работу приложения.

Затем (вариант 2):

Аварийно завершать работу приложения (пробросив наружу изначальное исключение с помощью команды `throw`).

Домашнее задание

Написать консольное приложение, которое спросит у пользователя тип фигуры (1 - круг, 2 - равносторонний треугольник, 3 - прямоугольник), затем спросит параметры фигуры:

1. для круга - диаметр
2. для треугольника - длину стороны
3. для прямоугольника - ширину и высоту

В качестве результата программа должна вывести площадь поверхности и длину периметра соответствующей фигуры.

Тип фигур должен быть объявлен в виде перечисления.

Необходимо обработать все предсказуемые исключения.



Домашнее задание

Пример работы программы (при корректном вводе):

- > Введите тип фигуры (1 круг, 2 равносторонний треугольник, 3 прямоугольник):
- > 3 /это ввод пользователя, соответствующий выбору прямоугольника/
- > Введите длину прямоугольника:
- > 12.1 /ввод пользователем ширины/
- > Введите высоту прямоугольника:
- > 9.4 /ввод пользователя высоты/
- > Площадь поверхности: 113.74
- > Длина периметра: 43

Пример работы программы (при неверном вводе):

- > Введите тип фигуры (1 круг, 2 равносторонний треугольник, 3 прямоугольник):
- > 3 /это ввод пользователя, соответствующий выбору прямоугольника/
- > Введите длину прямоугольника:
- > Abcd /ввод пользователем нечислового значения/
- > Ошибка! Введено нечисловое значение!



Спасибо за внимание.

