



C# programming language. The beginning

Course

Programming Languages

Semester 2, FIIT

Mayer S.F.

Mikhalkovich S.S

LECTURE # 2, 3. Basic constructions

Conditional if statement

```
string answer;  
    if (answer == "yes")  
    {  
        // Block of code executes if the value of the  
        // answer variable is "yes".  
    }  
    else  
    {  
        // Block of code executes if the value of the  
        // answer variable is not "yes".  
    }
```

Else if clauses

```
string answer;
    if (answer == "yes")
    {
        // Block of code executes if the value of the
        // answer variable is "yes".
    }
    else if (answer == "I_don't_know")
    {
        // Block of code executes if the value of the
        // answer variable is "I_don't_know".
    }
    else
    {
        // Block of code executes if the value of the
        // answer variable is neither above answers.
    }
}
```

Switch Statement

```
string answer=Console.ReadLine();
switch (answer)
{
    case "yes":
        // Block of code executes if the value of answer is
        // "yes".
        break;
    case "I_don't_know":
        // Block of code executes if the value of answer is
        // "I_don't_know".
        break;
    case "no":
        // Block of code executes if the value of answer is
        // "no".
        break;
    default:
        // Block executes if none of the above conditions
        // are met.
        break;
}
```

Samples

```
if (a<b)
    min = a;
else min = b;
```

Logical and

```
if (i==0 && i!=1)
    i++;
else i--;
```

```
if (x<y)
{
    var t = x;
    x = y;
    y = t;
}
```

Logical or

```
if (i > 0 || i % 2 == 0 )
    i+=2;
```

```
switch (i)
{
    case 1:
        WriteLine("First");
        break;
    case 2:
    case 3:
        WriteLine("Second");
        break;
    default:
        WriteLine("Third");
        break;
}
```

Ternary operator

```
int five = 5;  
bool answer = five == 5 ? true : false; // answer = true
```

Ternary operator

```
var rand = new Random();  
var weather = rand.Next(-20,20);  
string s = weather > 5 ? "good" : "bad";  
Console.WriteLine($"{weather} = {s}");
```

Tasks

Lesson #2, task 2

All the rest of Lesson #2 students have to do themselves <https://labs-org.ru/c-sharp2-eng/>

Loops

For loops & Foreach loops

```
for (int i = 0; i < 10; i++)  
    {  
        // Code to execute  
    }
```

```
string s = "loops";  
    // Process each character in the s:  
foreach (char c in s)  
    {  
        // Code to execute  
    }
```

An example of how to use a foreach loop **to iterate the characters of a string**:

```
string s = "loops";  
    // process each character in the s:  
    foreach (char c in s)  
    {  
        Console.WriteLine($"{c} "); // l o o p s  
    }
```

An example of how to use a foreach loop **to iterate a string array**:

```
string[] names = new string[5] {  
    "Ivan", "Max", "Olga", "Maria", "Mike"  
};  
  
    // process each name in the array  
    foreach (string name in names)  
    {  
        Console.WriteLine($"{name} ");  
        // output: Ivan Max Olga Maria Mike  
    }
```

while

```
string answer = Console.ReadLine();  
while (answer != "Quit") // "Quit" - is a special value to  
end the loop  
    {  
        // Process the data  
        answer = Console.ReadLine();  
    }
```

Do while

```
string answer;  
do  
    {  
        // Process the data  
        answer = Console.ReadLine();  
    } while (answer != "Quit");
```

Loops samples

```
i=5; j=0;  
while (i>0)  
{  
    i--;  
    j++;  
}
```

```
i=5; j=0;  
while (i>0)  
{  
    i--;  
    j++;  
}
```

```
for (int i=0; i<10; i++)  
    Write($"{i} ");
```

```
for (var i=1; i<100; i*=2)  
    Write($"{i} ");
```

```
foreach (var x in a)  
    Write($"{x} ");
```

Lecture task

Lesson #3, task 6

All the rest of Lesson #3 students have to do themselves <https://labs-org.ru/c-sharp3-eng/>

Exceptions

Exceptions

Exceptions are the best mechanism for error handling. An exception throws by a code with run-time error, and catches by a code that can handle an error. If the exception is not handled by a program, then the program terminates with crash.

An exception in C# is an object of an Exception class or some inherited class. Name of a class defines type of exception.

Example. Bad input.

```
static void Main()
{
    string s = Console.ReadLine();
    var a = int.Parse(s);
}
```

abc

Exception type:
System.FormatException

```
C:\WINDOWS\system32\cmd.exe
Необработанное исключение: System.FormatException: Входная строка имела неверный формат.
  в System.Number.StringToNumber(String str, NumberStyles options, NumberBuffer& number, NumberFormatInfo info, Boolean
  parseDecimal)
  в System.Number.ParseInt32(String s, NumberStyles style, NumberFormatInfo info)
  в System.Int32.Parse(String s)
  в ConsoleApp25.Program.Main() в C:\Users\Станислав\source\repos\ConsoleApp25\ConsoleApp25\Program.cs:строка 14
Для продолжения нажмите любую клавишу . . .
```


Exception handling

1 case – without an Exception

```
try
{
    string s = Console.ReadLine();
    int i = int.Parse(s);

    Console.WriteLine("NextLine");
}
catch (Exception e)
{
    Console.WriteLine("We catch the exception !!!");
}

Console.WriteLine("everything is good!");
```



```
1323
NextLine
everything is good!
```


Exception handling

2 case – with an Exception

```
try
{
    string s = Console.ReadLine();
    int i = int.Parse(s);

    Console.WriteLine("NextLine");
}
catch (Exception e)
{
    Console.WriteLine("We catch the exception !!!");
}

Console.WriteLine("everything is good!");
```



```
fdgh
We catch the exception !!!
everything is good!
```

Exception handling

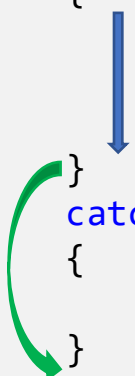
The exceptions are handled in **try ... catch** block.

When inside **try** block an exception appears, the program execution goes to the first appropriate exception handler – **catch** block.

If there are no handlers for the created exception, the program terminates with an error message.

Example. Bad input.

```
int a = 0;
bool flag = false;
do {
    try
    {
        string s = Console.ReadLine();
        a = int.Parse(s);
        flag = true;
    }
    catch (FormatException e)
    {
        Console.WriteLine(e.Message + " Repeat input");
    }
} while (!flag);
Console.WriteLine(a + " OK");
```

A blue arrow points from the 'try' block down to the 'catch' block, indicating the flow of execution when an exception is thrown. A green arrow points from the 'catch' block back to the 'while' loop, indicating the loop's continuation.

Exception handling

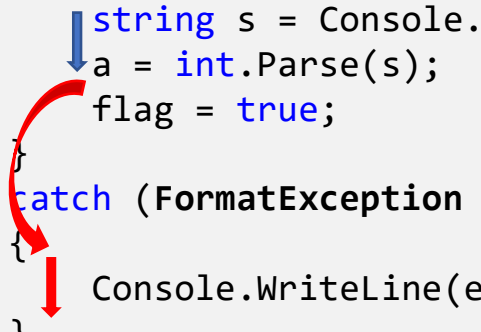
The exceptions are handled in try ... catch block.

When inside try block an exception appears, the program execution goes to the first appropriate exception handler – catch block.

If there are no handlers for the created exception, the program terminates with an error message.

Example. Bad input.

```
int a = 0;
bool flag = false;
do {
    try
    {
        string s = Console.ReadLine();
        a = int.Parse(s);
        flag = true;
    }
    catch (FormatException e)
    {
        Console.WriteLine(e.Message + " Повторите ввод");
    }
} while (!flag);
Console.WriteLine(a + " OK");
```

A diagram illustrating exception handling. A blue arrow points from the `string s = Console.ReadLine();` line to the `int.Parse(s);` line. A red arrow curves from the `int.Parse(s);` line down to the `catch (FormatException e)` block, indicating the flow of an exception. Another red arrow points from the `catch` block down to the `Console.WriteLine(e.Message + " Повторите ввод");` line.

Exception handling 2

Type value exceeding:

```
int a = int.Parse(Console.ReadLine());
int power = int.Parse(Console.ReadLine());
int result;
if (Math.Pow(a,power) > int.MaxValue)
{
    throw new InvalidOperationException("exceeds maximum Int32");
}
else
{
    result = (int)Math.Pow(a, power);
}
```

Lecture task

Lesson #4, lab 3

All the rest of Lesson #4 students have to do themselves <https://labs-org.ru/c-sharp3-eng/>