Declaring Variables

type variableName = value;

- string a="Hello";
- string b;
- b = "World!";
- int intValue1=80;
- int intValue2;
- intValue2 =20;
- const int Days = 365;
- int sum = intValue1+ intValue2;
- char myLetter = 'D';
- double myDoubleNum = 5.99D;
- double d = 1.2d;
- decimal dec = 389.5M;
- decimal m = 2m;
- bool myBool = true;
- float f = 1.2f;
- long l = 2L;

1

Conditional Operators

- For any flow control statements, we must be able to use conditional operators. These operators compare the values of two variables. These operators are:

C# Operators	Meaning
==	Equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
!=	Not equal to

2

Exercise: True or False

- int i1 = 10;
- int i2 = 5;
- (i1 == i2)
- (i1 > i2)
- (i1 < i2)
- (i1 >= i2)
- (i1 <= i2)
- (i1 != i2)

3

Exercise: True or False

- int i1 = 10;
- int i2 = 10;
- (i1 == i2) true
- (i1 > i2) false
- (i1 < i2) false
- (i1 >= i2) true
- (i1 <= i2) true
- (i1 != i2) false

4

Logical Operators

- We can make more complicated comparisons by using logical operators. These operators are:

C# Operators	Meaning
&&	And: Both must be true
||	Or: Either must be true
^	Xor : One but not both must be true
!	Not: Reverses Condition

5

Evaluating AND expressions

- An AND expression only evaluates to true if BOTH of the sub-expressions also evaluate to true. Otherwise, it evaluates to false.
- false **AND** false = false
- false **AND** true = false
- true **AND** false = false
- true **AND** true = true

6

AND – More examples

- A is true, B is false, C is true, D is true
- (A && B) && (C && D)

7

Evaluating OR expressions

- An OR expression only evaluates to true if EITHER of the sub-expressions also evaluate to true. Otherwise, it evaluates to false.
- false **OR** false = false
- false **OR** true = true
- true **OR** false = true
- true **OR** true = true

8

OR – More examples

- Assume: A is true, B is false, C is true, D is true
- What does the expression A || B evaluate to?
- What does the expression C || D evaluate to?
- What does (A || B) || (C || D) evaluate to?

9

Evaluating NOT expressions

- An NOT expression simply evaluates to the inverse value of the expression.
- **NOT** false = true
- **NOT** true = false

10

NOT - Example

- Assume: A is true and B is false
- What does the expression !A evaluate to?
- What does the expression !B evaluate to?

11

Order of Operations

- First, any expression in **parenthesis**
- Then the AND
- Then the OR
- Examples:

```
bool result;
result = true || true && false; // --> true
result = (true || true) && false; // --> false
result = true || (true && false); // --> true
```

12

Exercise: True or False

- int i1 = 10;
- int i2 = 5;
- int i3 = 5;

1. ((i1 == i2) && (i1 == i3))
2. ((i1 > i2) &&(i1 > i3))
3. ((i1 > i2) &&(i2 == i3))
4. ((i1 > i2) || (i1 > i3))
5. ((i1 > i2) ^ (i1 > i3))
6. ((i1 > i2) ^ (i2 > i3))
7. !((i1 > i2) && (i1 > i3))

13

Exercise: True or False

- int i1 = 10;
- int i2 = 5;
- int i3 = 5;

1. ((i1 == i2) && (i1 == i3)) false
2. ((i1 > i2) &&(i1 > i3)) true
3. ((i1 > i2) &&(i2 == i3)) true
4. ((i1 > i2) || (i1 > i3)) true
5. ((i1 > i2) ^ (i1 > i3)) false
6. ((i1 > i2) ^ (i2 > i3)) true
7. !((i1 > i2) && (i1 > i3)) false

14

If statements

- int i1 = 10;
- int i2 = 5;

```
if (i1 == i2)
{
    Console.Write("i1 is equal to i2!");
}
if (i1 > i2)
{
    Console.Write("i1 is greater than i2!");
}
if (i1 < i2)
{
    Console.Write("i1 is less than i2!");
}
```

- Equal? Greater? Less?

15

Why using if, else-if statement instead of multiple if statements?

```
int i1 = 10;
int i2 = 5;
if(i1 == i2) {
  Console.Write (“Equal”);
}
if(i1 > i2) {
  Console.Write (“Greater”);
}
if(i1 < i2) {
  Console.Write (“Less”);
}
```

```
int i1 = 10;
int i2 = 5;
if(i1 > i2) {
  Console.Write (“Greater”);
} else if(i1 < i2) {
  Console.Write (“Less”);
} else {
  Console.Write (“Equal”);
}
```

- Efficiency
- Readability

16

If/Else statements

```
int i1 = 10;
int i2 = 5;
if(i1 == i2) {
        Console.Write(“??”);
} else {
        Console.Write(“??”);
}
```

17

If/Elseif statements

```
int i1 = 10;
int i2 = 5;
```

```
if (i1 == i2)
{
    Console.Write("i1 is equal to i2!");
}else if (i1 > i2)
{
    Console.Write("i1 is greater than i2!");
}else
{
    Console.Write("i1 is less than i2!");
}
```

18

if-elseif-else statements

```
if(condition1) {
    Statements;
} elseIf (condition2) {
    statements;
} else {
    Statements;
}
```

19

If/Elseif statements

```
int i1 = 10;
int i2 = 5;
if(i1 > i2) {
        Console.Write (“??”);
} else if(i1 < i2) {
        Console.Write (“??”);
} else {
        Console.Write (“??”);
}
```

20

Logical Operators && Examples

- bool b1= true;
- bool b2 = false;
- if (b1 && b2) {
  Console.WriteLine(" ???")
  }else{
  Console.WriteLine(“?? ?")
  }

21

Logical Operators || Examples

- bool b1= true;
- bool b2 = false;
- if (b1 || b2) {
  Console.WriteLine("???")
  }else{
  Console.WriteLine(“?? ?")
  }

22

nest an if statement

```
bool Condition1 = true;
bool Condition2 = true;
bool Condition3 = true;
bool Condition4 = true;

if (Condition1)
{
    // Condition1 is true.
}
else if (Condition2)
{
    // Condition1 is false and Condition2 is true.
}
else if (Condition3)
{
    if (Condition4)
    {
        // Condition1 and Condition2 are false. Condition3 and Condition4 are true.
    }
    else
    {
        // Condition1, Condition2, and Condition4 are false. Condition3 is true.
    }
}
else
{
    // Condition1, Condition2, and Condition3 are false.
}
```

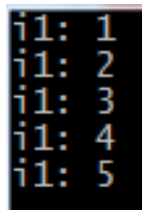
23

Which One Is Better?

Example 1	Example 2
(1)Error case. (2)Nominal case. (3)Nominal case. (4)Error case. (5)Nominal case. (6)Error case (7)Nominal case.	(1)Nominal case. (2)Nominal case. (3)Nominal case. (4)Nominal case. (5)Error case. (6)Error case. (7)Error case. (8)Error case.

24

While Loops




```
int i1 = 0;
int i2 = 5;

while (i1 < i2)
{
    i1 += 1;
    Console.WriteLine("i1: " + i1.ToString());
}
```

How many times will the loop run?
What are the outputs?

25

While Loops

```
int i1 = 0;
int i2 = 5;
while(i1 < i2) {
 i1 -= 1;
Console.WriteLine(i1.toString);
}
```

infinite:
i1 <i2
Loop
forever

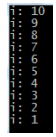
How many times will the loop run?
What are the outputs?

26

Do While

```
do {
     statements
}while (condition);
```

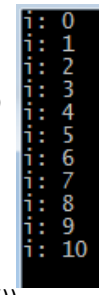
```
int i = 10;
do {
      Console.WriteLine("i: " + i.ToString());
      i = i - 1;
} while (i > 0);
```



How many times will the loop run?
What are the outputs?

27

For Loops




```
for(int i = 0; i <= 10; i++)
{
     statements
}
```

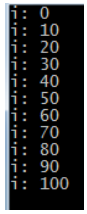
```
for (int i = 0; i <= 10; i++)
{
      Console.WriteLine("i: " + i.ToString());
}
```

How many times will the loop run?
What are the outputs?

28

For Loops

```
for(int i = 0; i <= 100; i += 10){
      Console.WriteLine("i: " + i.ToString());
}
```



How many times will the loop run?
What are the outputs?

29

Nested For loops

```
for ( init; condition; increment )
{
      for ( init; condition; increment ) {
            statement(s);
      }
      statement(s);
}
```

30

Nested For loops:

```
for (int i = 0; i <= 2; i++){
    for (int j = 0; j <=2; j ++) {
        Console.WriteLine("i: " + i.ToString());
        Console.WriteLine("j: " + j.ToString());
    }
}
```

How many times will the loop run?
What are the outputs?

31

```
public class Student {
   private string _name;
   private int _age;
   private int _id;

   public Student(string name, int age, int id = 0) {
      _name = name;
      _age  = age;
      _id   = id;
   }

   public string Name{get; set;}
   public string Age{get; set;}
   public string ID{get; set;}

   public string PrintAge() {
      return "Student " + _name + "(" + _id + ") is"+ _age + " years old";
   }

}
```

32

University

A University class that uses the Student Class can be defined as follows:

```
public class University {
  private string _uniName;
  private Dictionary<int, Student> _students;

  public University(string uniName) {
    _uniName = uniName;
    _students = new Dictionary<int,Student>();
  }
}
```

33

University

Each University will store a set of Students using a Dictionary.

Example of defining the behaviour to enrol a new Student in the University is as follows:

```
public void Enrol(Student newStudent) {
  _students.Add(newStudent.ID, newStudent);
  }
```

34

University

The behaviour for printing out all the students and their ages is as follows:

```
public string Print() {
  string uniStudentRoll;
  uniStudentRoll = _UniName + "\n"
  foreach(Student s in _Students.Values ) {
     uniStudentRoll += s.PrintAge() +  "\n";
  }
  return uniStudentRoll ;
}
```

35

University

A University object can be created and Students added to it one at a time:

```
University massey = new University("Massey");
massey.enrol(new Student("Tim", 25, 01234567);
massey.enrol(new Student("Tom", 23, 76543210);
Console.WriteLine(massey.Print());
```

36

Exception Handling

Exceptions can be caught by the application using a **try/catch** block.

```
try {

} catch(exception variable) {

} finally {

}
```