

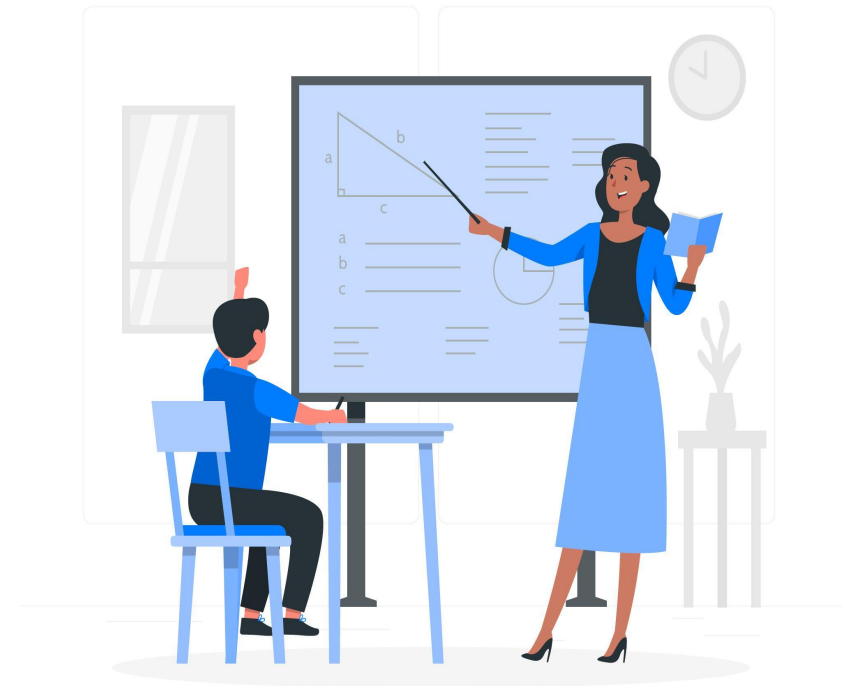
C#



Variables

- ❑ "Variables" are simply storage locations for data.
- ❑ You can place data into them and retrieve their contents as part of a *C#* expression.
- ❑ The interpretation of the data in a variable is controlled through "Types".
- ❑ In *C#*, you declare a variable in this format:
❑ [modifiers] datatype identifier;

```
public static string role = admin;
```



Control Flow Statements

- ❑ Control flow and program logic are of the most important parts of a programming language.
- ❑ Selection statements select one of a number of possible statements for execution based on the value of some expression.
 - The if statement
 - The if-else Statement
 - The if-else if-else Statement
 - The Nested if-else Statement
 - The switch statement



If Statement

Use the if statement to specify a block of *C#* code to be executed if a condition is True.

Syntax :

```
if ( boolean-expression )  
{  
    embedded-statement  
;  
}
```

```
if (role==admin)  
{ Console.WriteLine("welcome  
to admin");  
}
```

If/else Statement

Use the else statement to specify a block of code to be executed if the condition is False.

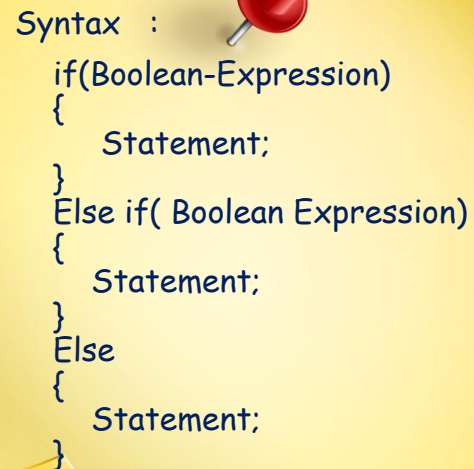
Syntax :

```
if ( boolean-expression )  
{  
    embedded-statement ;  
}  
else  
{  
    embedded-statement ;  
}
```

```
if (role==admin)  
{  
    Console.WriteLine("welcome to  
admin");  
}  
else  
{  
    Console.WriteLine("Error");  
}
```

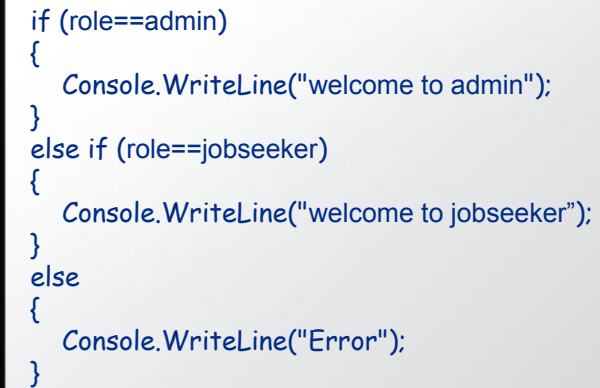
if/else if/else Statement

Use the else if statement to specify a new condition if the first condition is False.



Syntax :

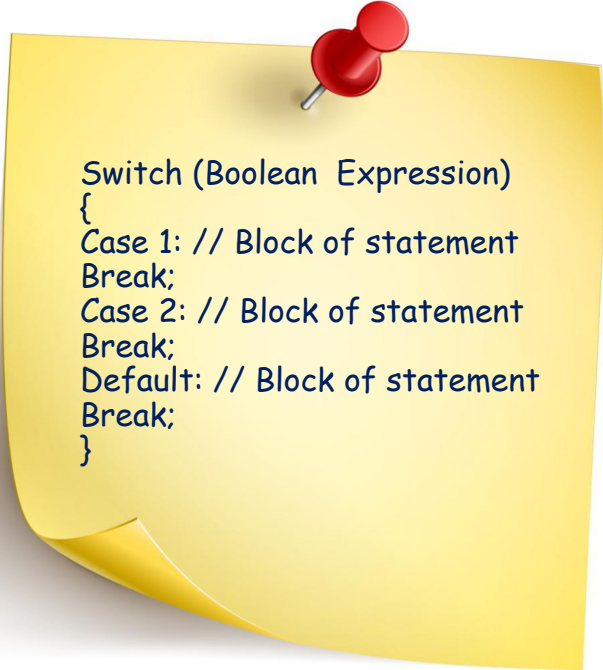
```
if(Boolean-Expression)
{
    Statement;
}
Else if( Boolean Expression)
{
    Statement;
}
Else
{
    Statement;
}
```



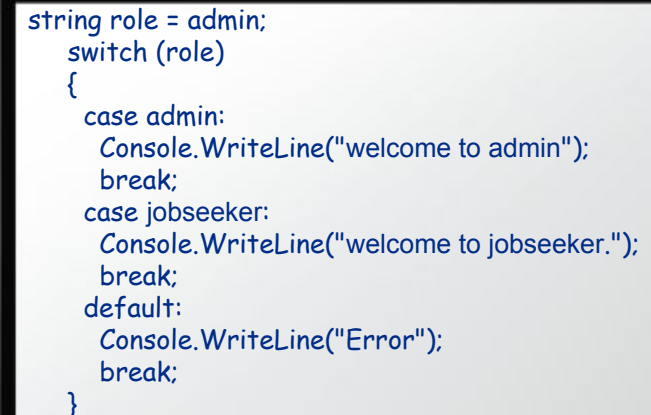
```
if (role==admin)
{
    Console.WriteLine("welcome to admin");
}
else if (role==jobseeker)
{
    Console.WriteLine("welcome to jobseeker");
}
else
{
    Console.WriteLine("Error");
}
```

Switch Statement

Use the switch statement to select one of many code blocks to be executed.



```
Switch (Boolean Expression)
{
Case 1: // Block of statement
Break;
Case 2: // Block of statement
Break;
Default: // Block of statement
Break;
}
```



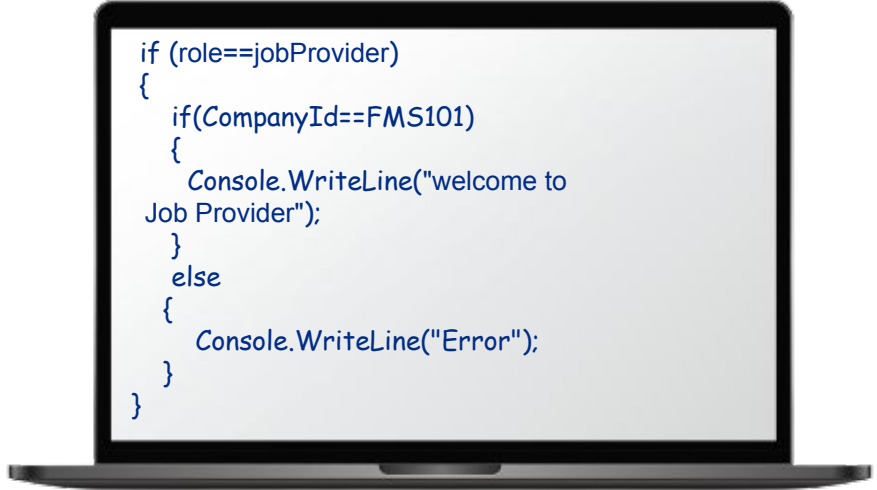
```
string role = admin;
switch (role)
{
    case admin:
        Console.WriteLine("welcome to admin");
        break;
    case jobseeker:
        Console.WriteLine("welcome to jobseeker.");
        break;
    default:
        Console.WriteLine("Error");
        break;
}
```

Nested If-else Statement

Nested IF functions, meaning one IF function inside of another, allows you to test multiple criteria and increases the number of possible outcomes.

Syntax :

```
if(condition1)
{
    if(condition2)
    {
        // execute code when condition1 and condition2 are
        true
    }
    else
    {
        // execute code when condition1 is true and
        condition2 is false
    }
}
else
{
    // execute code when conditions1 is false
}
```



```
if (role==jobProvider)
{
    if(CompanyId==FMS101)
    {
        Console.WriteLine("welcome to
Job Provider");
    }
    else
    {
        Console.WriteLine("Error");
    }
}
```


Control Statement- Loop

Iteration statements repeatedly execute an embedded statement.

- while-statement
- do-while statement
- for-statement
- foreach-statement



While-statement

The while statement conditionally executes an embedded statement zero or more times.

Syntax:

```
while( condition )  
{  
    embedded-statement;  
}
```

Example

```
int i = 0;  
string[] jobs = new string[10];  
Console.WriteLine("Enter the no of jobs posted ?");  
int count = Convert.ToInt32(Console.ReadLine());  
Console.WriteLine("Enter Jobs");  
while(count!=i)  
{  
    jobs[i] = Convert.ToString(Console.ReadLine());  
    i++;  
}  
Console.WriteLine("=====");  
for (i = 0; i <= count; i++)  
{  
    Console.WriteLine(jobs[i]);  
}  
Console.ReadLine();
```

```
Enter the no of jobs posted ?  
2  
Enter Jobs  
Jr Angular Developer  
Jr .Net Developer  
=====  
Jr Angular Developer  
Jr .Net Developer
```

do-while Statement

This statement executes its embedded statements one or more times.

Unlike the while Statement ,a do-while loop is executed once before the conditional expression evaluated.

Syntax:

```
do
{
    //code to be executed
}
while ( condition )
```

Example

```
int i = 0;
string[] jobs = new string[10];
Console.WriteLine("Enter the no of jobs posted ?");
int count = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Enter Jobs");
do
{
    jobs[i] = Convert.ToString(Console.ReadLine());
    i++;
}
while (count != i);
Console.WriteLine("=====");
Console.WriteLine("POSTED JOBS");
Console.WriteLine("=====");
for (i = 0; i <= count; i++)
{
    Console.WriteLine(jobs[i]);
}
Console.ReadLine();
```

```
Enter the no of jobs posted ?
2
Enter Jobs
Jr Angular Developer
jr .Net Developer
=====
POSTED JOBS
=====
Jr Angular Developer
jr .Net Developer
```

The for statement

This statement begins with the for keyword and is followed by parentheses. The parentheses contain an initializer, a condition, and an iterator statement, all separated by semicolons.

Syntax:

```
for ( Initialization ; condition ; Iterator )  
{  
    Statement;  
}
```

Example

```
string[] jobs = new string[10];  
Console.WriteLine("Enter the no of jobs posted ?");  
int count = Convert.ToInt32(Console.ReadLine());  
Console.WriteLine("Enter Jobs");  
for (int i = 1; i <= count; i++)  
{  
    jobs[i] = Convert.ToString(Console.ReadLine());  
}  
Console.WriteLine("-----");  
Console.WriteLine("POSTED JOBS");  
Console.WriteLine("-----");  
for (int i = 1; i <= count; i++)  
{  
    Console.WriteLine(jobs[i]);  
}  
Console.WriteLine("-----");  
Console.ReadLine();
```

```
Enter the no of jobs posted ?  
2  
Enter Jobs  
Asst Manager  
Senior .Net Developer  
-----  
POSTED JOBS  
-----  
Asst Manager  
Senior .Net Developer  
-----
```

The foreach statement

The foreach statement enumerates the elements of a collection, executing an embedded statement for each element of the collection.

Syntax :

```
foreach(local-variable-type identifier  
in expression )  
{  
    embedded-statement  
}
```

Example

```
string[] jobs = new string[] { "Manager","Tester","Developer"};  
  
foreach(var job in jobs)  
{  
    Console.WriteLine(job);  
}  
Console.ReadLine();
```

```
Manager  
Tester  
Developer
```

Thank You