



# Logols Learning

WEEKEND WEB DEVELOPMENT BOOT CAMP

TRAINING: C#



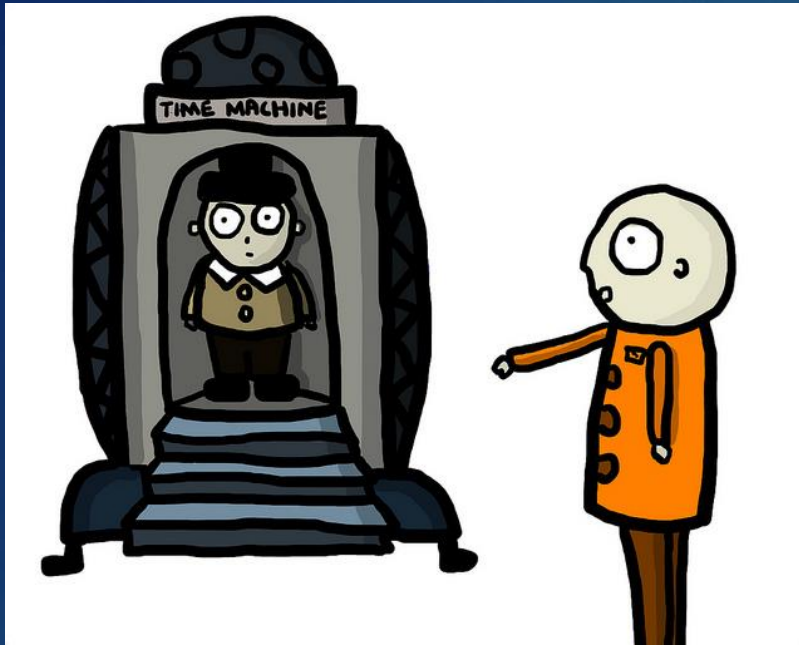
# Visual Studio Code

# .Net Command Line Interface (CLI)

- ▶ Commands within the command line
- ▶ Entered in the terminal window
- ▶ Basic Commands
  - ▶ new, restore, build, run, clean
- ▶ Project Modification Commands
  - ▶ add/remove package, add/remove reference

# CLI new Examples

- ▶ mkdir – create directory
- ▶ cd – change directory
- ▶ Console project:
  - ▶ dotnet new console
- ▶ Class Library project:
  - ▶ dotnet new classlib
- ▶ Web API project:
  - ▶ dotnet new webapi



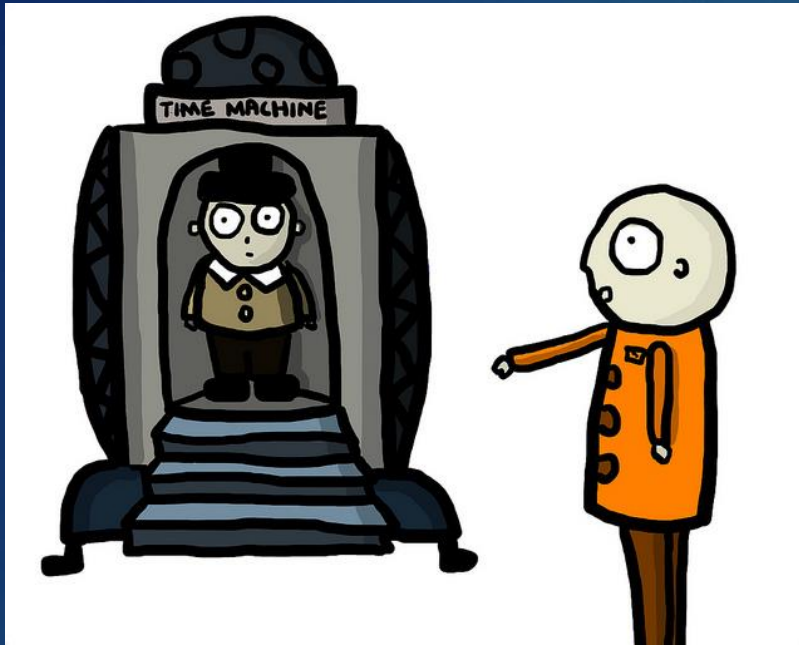
# EXAMPLE

NEW CONSOLE APPLICATION IN VISUAL STUDIO CODE

# Statements

- ▶ Made up of:
  - ▶ Keywords
  - ▶ Expressions
  - ▶ Operators
- ▶ Statements end with a Semicolon ;
- ▶ Statements can span multiple lines
- ▶ Statement blocks contain multiple statements
  - ▶ Surrounded by curly braces { }
  - ▶ Can have blocks within blocks





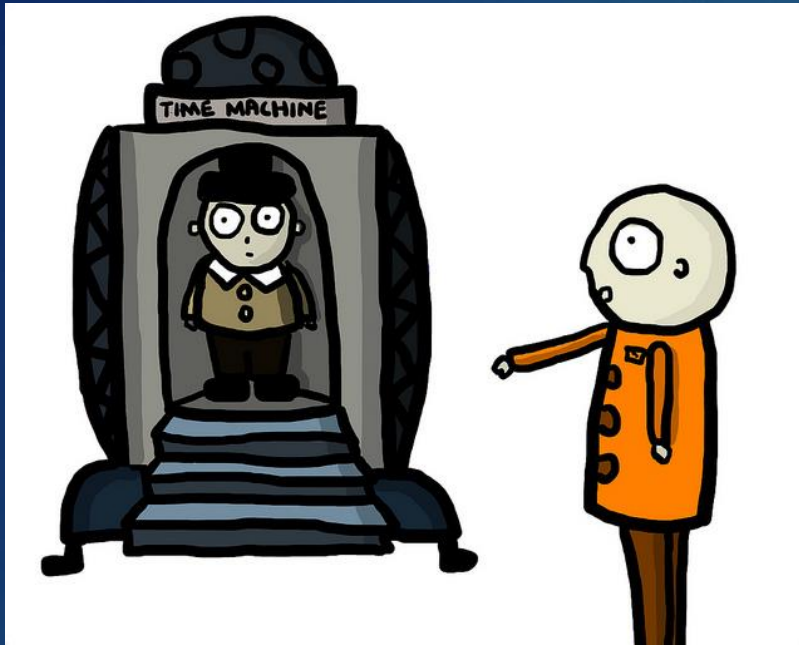
# EXAMPLE

STATEMENTS AND STATEMENT BLOCKS

# C# Comments

- ▶ `//` this is a comment
  - ▶ Single line comments
- ▶ `/*` this is a multi line  
comment `*/`
  - ▶ Multi-line comments





# EXAMPLE

COMMENTS

# Types

- ▶ Basic Built-In Types

- ▶ bool

- ▶ int

- ▶ decimal

- ▶ string

- ▶ array

# Declaring String Variables

- ▶ Declaring Variables
  - ▶ `string myString;`
  - ▶ `string myString = "test string";`
- ▶ Using Variables
  - ▶ `Console.WriteLine(myString);`

# Declaring Number Variables

- ▶ Declaring Variables

- ▶ `int myInt;`

- ▶ `int myInt = 5;`

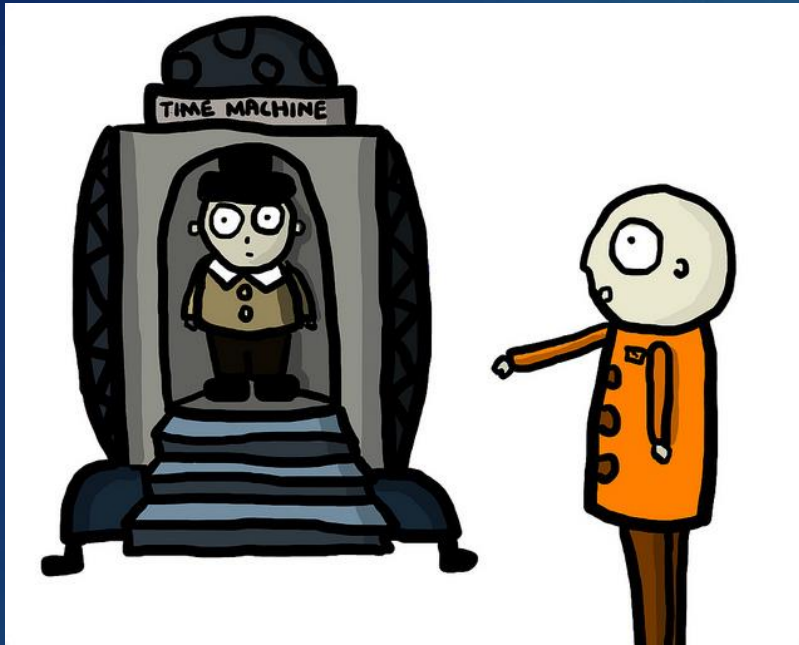
- ▶ `decimal myDecimal = 5.234;`

- ▶ Using Variables

- ▶ `Console.WriteLine(myDecimal);`

# Declaring Bool Variables

- ▶ Declaring Variables
  - ▶ `bool myBool;`
  - ▶ `bool myBool = true;`
- ▶ Using Variables
  - ▶ `Console.WriteLine(myBool);`



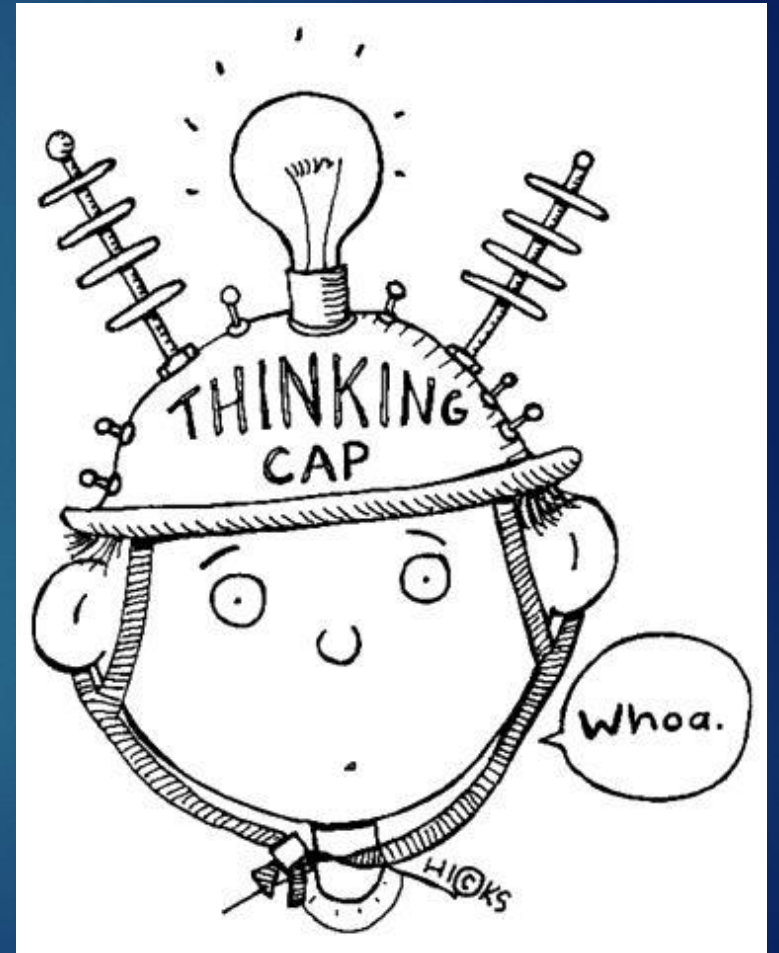
# EXAMPLE

DECLARING VARIABLES



# ASSESSMENT

CLI, STATEMENTS, BLOCKS, COMMENTS, VARIABLES



# Comparison Operators

- ▶ **Do not compare with =**
- ▶ < Less Than
- ▶ > Greater Than
- ▶ <= Less Than or Equal To
- ▶ >= Greater Than or Equal To
- ▶ == Equal To
- ▶ != Not Equal To

# Logical Operators

- ▶ & And
- ▶ | Inclusive Or
- ▶ && Conditional And
- ▶ || Conditional Or

# If Statement

## ► Example:

```
bool myVariable = true;  
If (myVariable)  
{  
    console.WriteLine("true");  
}
```

# If-Else Statement

## ► Example:

```
bool myVariable = true;  
If (myVariable)  
{  
    console.WriteLine("true");  
}  
else  
{  
    console.WriteLine("false");  
}
```

# Nested If Statement

## ► Example:

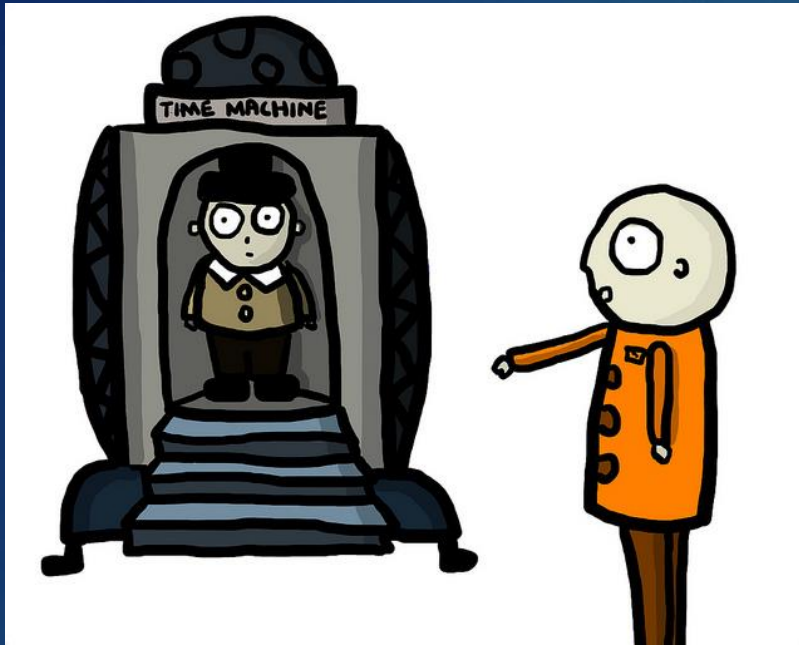
```
bool myVariable = true;  
bool myVariable2 = false;  
If (myVariable)  
{  
    if(myVariable2)  
    {  
        console.WriteLine("true");  
    }  
}
```



# If Multiple Else Statement

## ► Example:

```
bool myVariable = true;
bool myVariable2 = true;
If (myVariable)
{
    console.WriteLine("true");
}
else if(myVariable2)
{
    console.WriteLine("variable2 true");
}
else
...
```



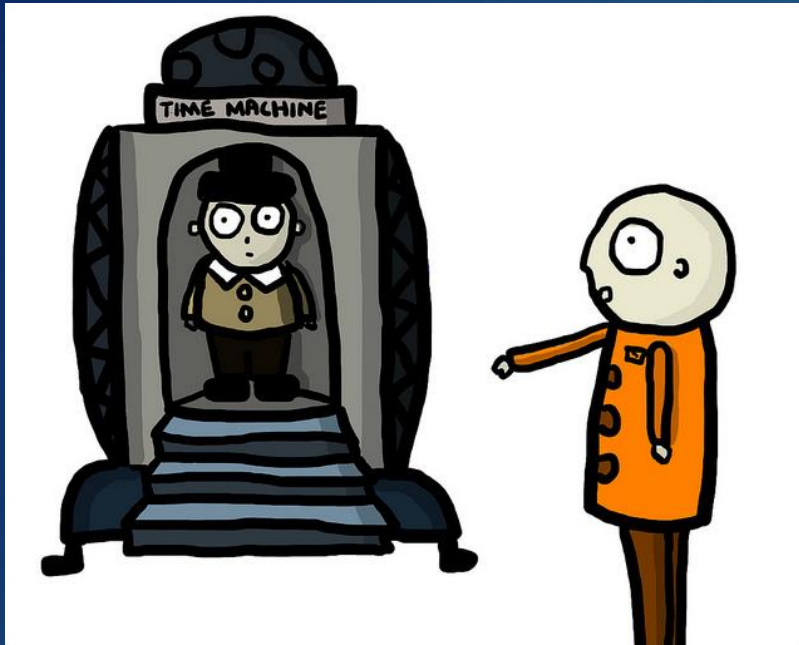
# EXAMPLE

IF ELSE STATEMENTS

# Switch Statement

## ► Example

```
int myVariable;  
switch(myVariable)  
{  
    case 1:  
        Console.WriteLine("1");  
        break;  
    case 2:  
    case 3:  
        Console.WriteLine("2 or 3");  
        break;  
    default:  
        Console.WriteLine("default");  
        break;  
}
```

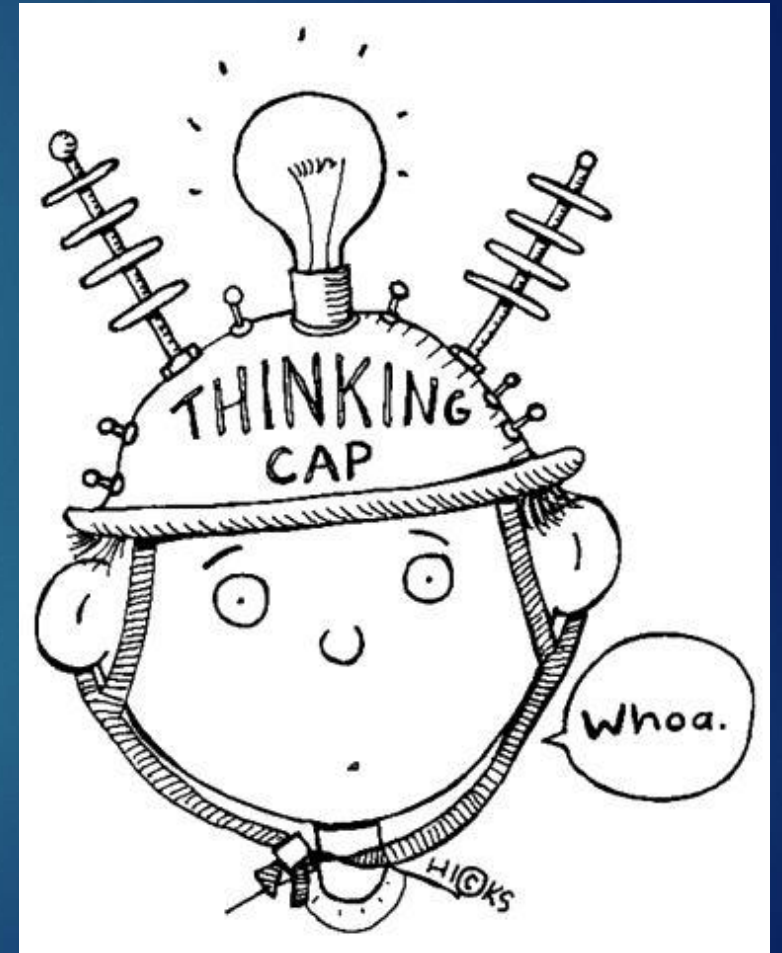


# EXAMPLE

SWITCH STATEMENTS

# ASSESSMENT

CONDITIONAL STATEMENTS





# Assignment

- ▶ A status report is needed of all government employees. Statuses are:
  - ▶ 1: Alive, 2: Zombie, 3: Dead, 4: Unknown
- ▶ Given an int variable, write if else statements and console out the persons status.
- ▶ Using the same int variable, modify your code to perform the same operation with a switch statement.





# Value and Reference Types

- ▶ Type System

- ▶ Value Types

- ▶ Contain data within it's own memory location.
    - ▶ Int, decimal, bool, struct

- ▶ Reference Types

- ▶ Contain a pointer to a memory location.
    - ▶ Require a new instance of an object.
    - ▶ Are null if no instance of an object has been provided.
    - ▶ string, array, class

# Declaring Arrays

## ▶ Declaring Variables

- ▶ `int[] myArray;`
- ▶ `myArray = new int [5];`
- ▶ `myArray = new int[] {0, 1, 2, 3};`
- ▶ `int[] myArray = new int[] {0, 1, 2, 3};`
- ▶ `int[] myArray = {0, 1, 2, 3};`

## ▶ Using Variables

- ▶ `myArray[5] = 6;`
- ▶ `Console.WriteLine(myArray[5]);`
- ▶ `myArray.Length`

# while Loop

## ► Example

```
int[] myArray = {0, 1, 2, 3};  
int counter = 0;
```

```
while (counter < myArray.Length)  
{  
    Console.WriteLine(myArray[counter].ToString());  
    counter++;  
}
```

# do-while Loop

## ► Example

```
int[] myArray = {0, 1, 2, 3};  
int counter = 0;
```

```
do  
{  
    Console.WriteLine(myArray[counter].ToString());  
    counter++;  
} while (counter < myArray.Length);
```

# for Loop

## ► Example

```
int[] myArray = {0, 1, 2, 3};
```

```
for(int counter = 0; counter < myArray.Length;  
counter++)  
{  
    Console.WriteLine(myArray[counter].ToString());  
}
```

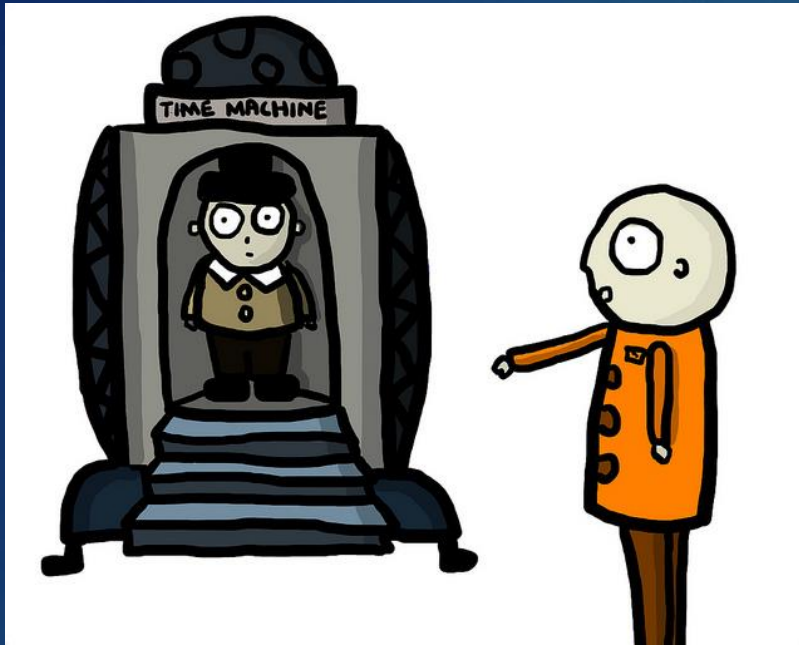
# foreach Loop

## ► Example

```
int[] myArray = {0, 1, 2, 3};
```

```
foreach(int value in myArray)
{
    Console.WriteLine(value.ToString());
}
```



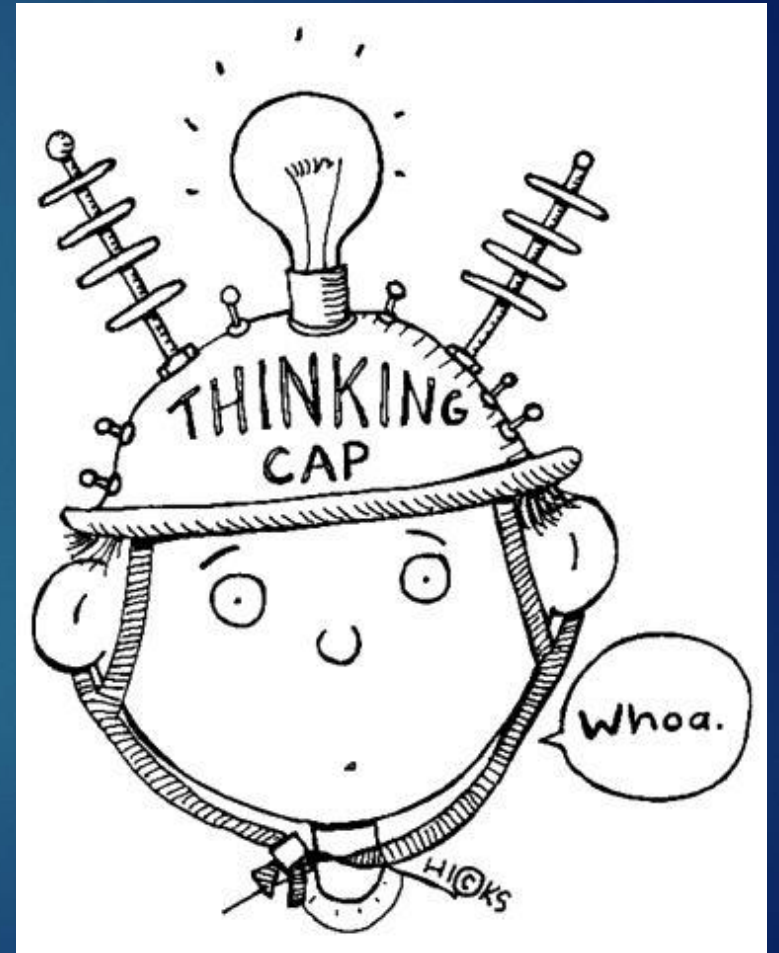


# EXAMPLE

LOOPS

# ASSESSMENT

LOOPS



# Assignment

- ▶ A status report is needed of all government employees. Statuses are:
  - ▶ 1: Alive, 2: Zombie, 3: Dead, 4: Unknown
- ▶ Given an array of int variable, write loops with if else statements and console out everyone's status.
- ▶ Use all loop types.
- ▶ Given another array of string variables with names, write out the name and their status.



# Methods

- ▶ Smaller and Manageable
- ▶ Cohesive Actions
- ▶ Reusable
- ▶ Functions Return a Value
  - ▶ Only one value can be returned
- ▶ Voids do not Return a Value
- ▶ Parameters
- ▶ Method Overloads

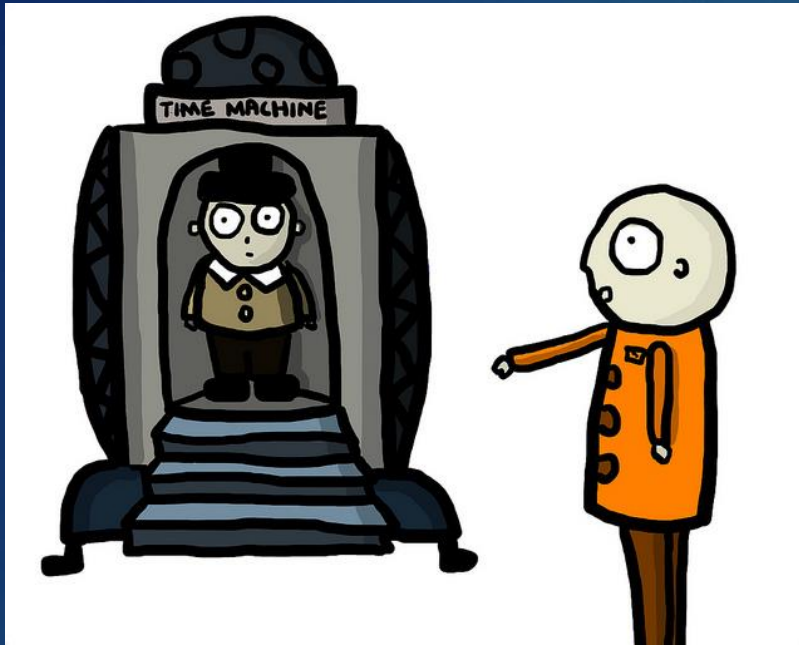
# Method Syntax

```
[access modifier] [return type] [name]([type1] [parameter1],  
[type2] [parameter2])  
{  
    Statements...;  
}
```

► Example:

```
private int AddNumbers(int num1, int num2)  
{  
    Statements...;  
}
```





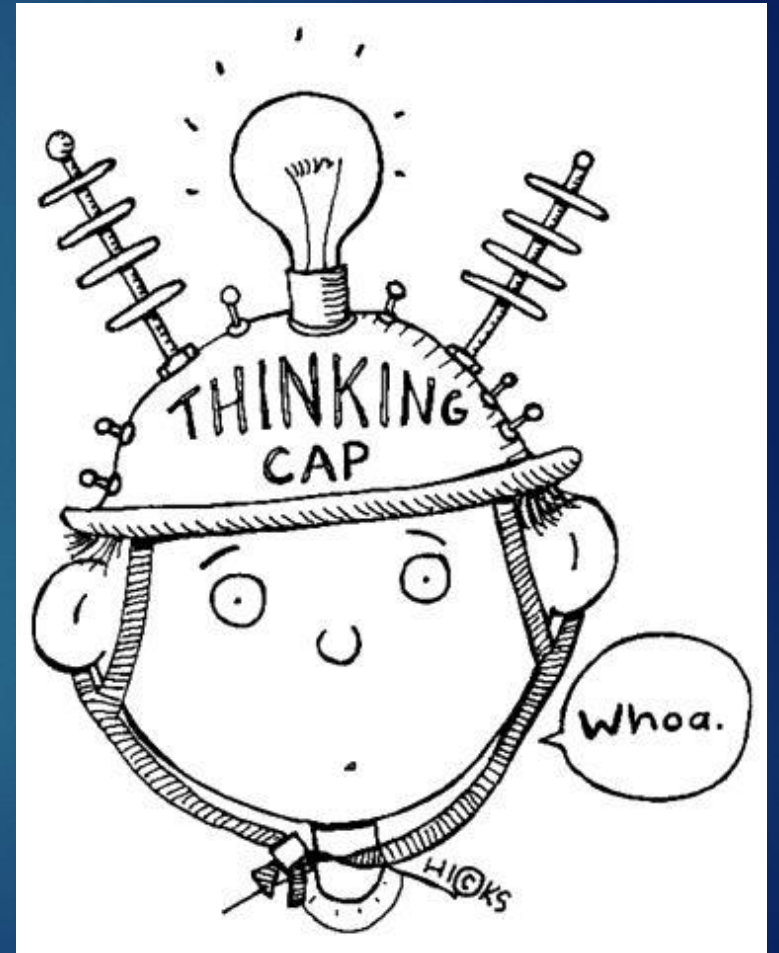
# EXAMPLE

METHODS



# ASSESSMENT

METHODS



# Assignment

- ▶ A status report is needed of all government employees. Statuses are:
  - ▶ 1: Alive, 2: Zombie, 3: Dead, 4: Unknown
- ▶ Modify your previous program to create a method that handles the condition given a parameter for status and for name that returns the concatenated string.
- ▶ Write a void method that takes a string parameter and writes it to the console.

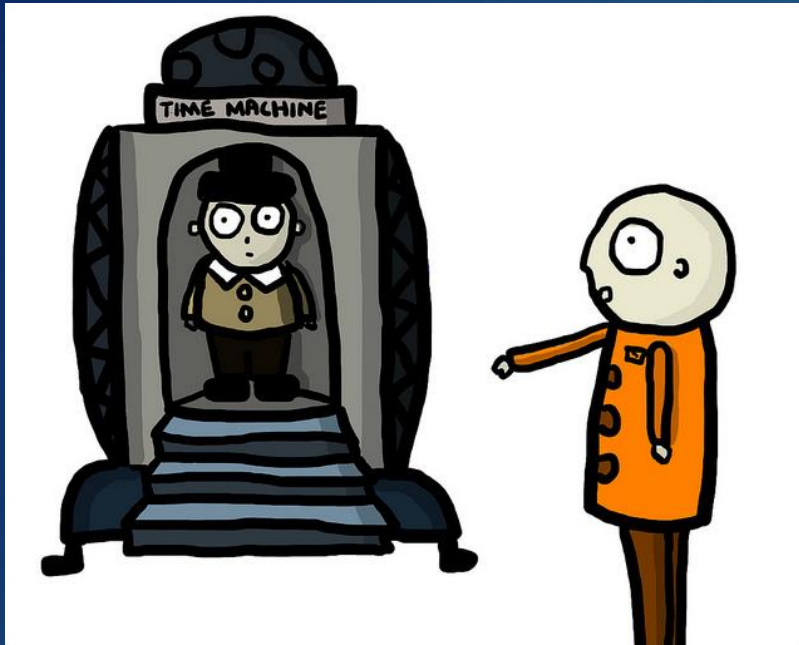


# Working with Generic Types

- ▶ Type Safety
- ▶ Re-use
- ▶ Generic Collections –  
System.Collections.Generic

- ▶ Example:

```
List<string> strings = new List<string>();  
strings.Add("test");  
List<int> ints = new List<int>();  
ints.Add(3);
```



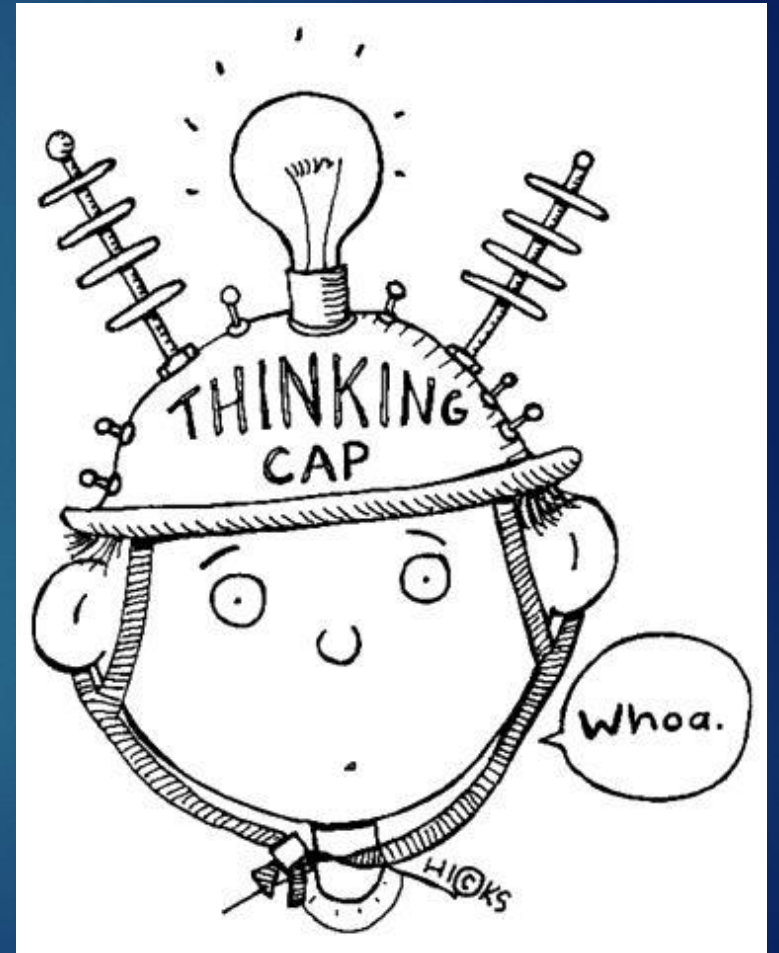
# EXAMPLE

GENERIC



# ASSESSMENT

GENERICS



# Assignment

- ▶ A status report is needed of all government employees. Statuses are:
  - ▶ 1: Alive, 2: Zombie, 3: Dead, 4: Unknown
- ▶ Modify your previous program to create a generic list of names of everyone who is alive.
- ▶ At the end of the program, list everyone still alive.





# QUICK REVIEW

C#



Not really a sign you'd want to see whilst driving through an eerily quiet neighbourhood...

# Additional Resources

- ▶ Code Katas
  - ▶ <https://www.codewars.com/>
- ▶ DotNet Fiddle
  - ▶ <https://dotnetfiddle.net/>
- ▶ Codeasy.net
  - ▶ <https://codeasy.net/welcome>
- ▶ Microsoft Virtual Academy
  - ▶ <https://mva.microsoft.com/>
- ▶ Microsoft Docs
  - ▶ <https://docs.microsoft.com/en-us/dotnet/csharp/index>

# Keep Practicing!

- ▶ Try declaring different types of variables.
- ▶ Try different combinations of if, else statements.
- ▶ Try different combinations and logic for loops.
- ▶ Try creating different methods with different parameters and return types.
- ▶ Try different ways of working with the generic list.