C# Notes

Naming Conventions

- For local variables: Camel Case (int rollNumber)
- For constants: Pascal Case (const int MaxZoom = 5)

Primitive Types

- Integral Numbers
    - Byte
    - Short
    - Int
    - Long
- Real Numbers
    - Float
    - Double
    - Decimal
- Character – char
- Boolean – bool

> Note – double is used as a default data type when using real numbers in C#.

- Declare a float `float number = 1.2f`
- Declare a decimal `decimal number = 1.2m`

Overflowing – Overflow happens when we perform an operation with a data type and the result of this operation exceeds the size of a storage for this datatype.

```
byte number = 255;
number = number + 1 //this will output 0
```

Type Conversion

1. **Implicit Type Conversion**

```
byte b = 1;
int i = b;

int i = 1;
float f = i
```

2. **Explicit Type Conversion**

```
int i = 1;
byte b = i //won't compile, so we use explicit type conversion

byte b = (byte)i;
```

3. **Non-Compatible Types**

```
string s = "1";
int i = (int)s; //won't compile

int i = convert.ToInt32(s);
or
int i = int.Parse(s);
```

Non-Primitive Types

1. String
2. Array
3. Enum
4. Class

Classes

Combines related variables (fields) and functions (methods).

Declaring a class

```
public class Person {
  public string Name;
  public void Introduce() {
    Console.WriteLine("Hi, my name is " + Name);
  }
}
```

Creating Objects

```
Person person = new Person();
//or
var person = new Person();
person.Name = "John";
person.Introduce(); //Hi, my name is John
```

Arrays

An array is a data structure to store a collection of variables of the same type.

Declaring an array

```
int[] numbers = new int[3];
numbers[0] = 1;
numbers[1] = 2;
numbers[3] = 3;

//or directly use object initialization syntax
int[] numbers = new int[3] {1, 2, 3};
```

Strings

A sequence of characters.

Creating strings

```
string name = "John";

//Using string concatenation
string name = firstName + " " + lastName;

//Using string format
string name = string.Format({0} {1}, firstName, lastName);

//Using string join
var numbers = new int[3] {1, 2, 3};
string list = string.Join(",", numbers); //1,2,3
```

String Elements

```
string name = "John";
char firstChar = name[0]; //J

name[0] = 'm'; //cannot be done as strings are immutable
```

> Strings are Immutable. Once you create them, you cannot change them.

Escape Characters

Char	Description
\n	New Line

Char	Description
\t	Tab
\	Backslash
'	Single Quotation Mark
"	Double Quotation Mark

Verbatim Strings

```
string path = "c:\\projects\\csharp\\folder";
//using verbatim strings
string path = @"c:\projects\csharp\folder";
```

Enums

A set of name/value pairs (constants).

```
public enum ShippingMethod {
 RegularShipping = 1,
 RegisteredMail = 2,
 Express = 3
}

var method = ShippingMethod.Express;

//We can also specify the type for enum
public enum ShippingMethod : byte {

}
```