C# Cheat Sheet

Primitive data types

These are basic types defined by the programming language to store simple values like integers, double, float, char, boolean.

Variables

Variables are containers that stores values. They have a type, a name, and a value.

```
int number1 = 5;
double number2 = 3.14;

// Compare two variables (primitive data
types only)
number1 == number2
```

Strings

A string is a sequence of characters, specified between quotes.

```
String name1 = "John";
String name2 = "Jane";
```

Strings cannot be compared like primitive data types.

```
Console.WriteLine(name1.Equals(name2));
```

Print to the console

It is useful to print values to the console. To show the result of an operation on the screen, to find out if our algorithm is giving the right results.

```
Console.WriteLine( "John" );
```

Keyboard (Console) input

```
Console.WriteLine("Enter a number: ");
int a = Convert.ToInt32(Console.ReadLine());
Console.WriteLine(a);
Console.WriteLine("Enter a name: ");
String name = Console.ReadLine();
Console.WriteLine(name);
```

Arrays

An array is a collection of values of the same type. We access the elements using an index. The first element is in the position 0.

```
int[] array1 = {1,2,3,4,5,6,7};
Console.WriteLine(array1[0]);
```

```
int b = array1[1] + array1[2];
Console.WriteLine(b);
//Create an array to store 5 elements
int[] array2 = new int[5];
// Asign a value to the first position
array2[0] = 15;
Console.WriteLine(array2[0]);
```

if-else

If-else structure is used to do something according to a condition.

```
if (number1>10){
    // Do something
}
else{
   //Do something else
}
```

Loops

Loops can have pre-condition or post-condition.

```
//With pre-condition
for (int i = 10; i < 5; i++) {
   //Do something
}
```

```
int index = 0;
while (index<5){
    //Do something
}
//With post-condition
do {
    //Do something
}while(index<5);
```

Classes

A class is a type. It allows you to represent the behaviours of certain objects. It has a name, attributes and methods.

```
class Person{
    private String name;
    public Person(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
}
```

Objects

Objects are instances of a class.

```
Person person1 = new Person("John");
Console.WriteLine(person1.getName());
```

Inheritance

This refers to the ability of a class to inherit the behaviour of another class. In the example below, the class Student inherited the method *getName* from the class *Person*.

```
class Student : Person{
 public Student(String name): base(name){
 }
}
```

Static and dynamic type

The static type is the type used in the object declaration. The dynamic type is the type used in object creation. In the example below, *Person* is the static type and *Student* is the dynamic type. You can only write this type of statement if the class *Student* inherited from the class *Person*.

```
Person person1 = new Student("John");
```

Sometimes, you have to ask for the dynamic type of an object from your code.

```
if (person1.GetType() == typeof(Student))
{
    Console.WriteLine("It is student");
}
```

Exceptions

Exceptions allow us to create robust software. It is a must to use them in programming.

```
try
{
   // Do something that can
   //throw an exception
}
catch (System.Exception)
{
    //Catch the exception
    //and do something
}
finally{
    //Do something independently
    //if an exception happened or not.
}
```

Files

Files allow us to store information in a hard drive.

Write plain text to a file

```
using System.IO;
…

using (StreamWriter writer = new
StreamWriter("file.txt"))
{
   try{
      writer.WriteLine("Line 1");
      writer.WriteLine("Line2");
      writer.WriteLine("Line3");
   }
   catch(System.Exception e){
       //Do something
   }
}
```

Read plain text from a file

```
using System.IO;
…
using (StreamReader reader = new
StreamReader("file.txt"))
{
   try{
      while(!reader.EndOfStream){
         Console.WriteLine(reader.ReadLine());
      }
   }
   catch(System.Exception e){
      //Do something
   }
}
```

ComputingLearner.com