C# Programming Reference Sheet

Built In Data Types & Literals

Integers

```
int(signed 32 byte), long(signed 64
byte),short(signed 16 byte), uint(unsigned 32 byte)
(signed uses the bytes to represent negative num but
unsigned uses it all for positive numbers
```

Floating Point Numbers

```
double(8bytes, precision ~15-17 digits),
float(4bytes, precision ~6-9 digits),
decimal(16bytes, precision ~28-29 digits)
```

Strings and Characters

```
char(represents a character), string(represents a
bunch of char-characters)
```

Boolean

```
bool (true or false)
```

Working with Strings

Assignment (giving a string a value)

```
string x = "Hello";
```

Concatenation (joining strings)

```
string y = x + " World!";
```

Comparison

```
x == y, x == "Hello", "Hello" != "World"
```

Construction from other types:

```
string x = a.ToString();
string y = Convert.ToString(4);
```

value.toString() will cause an error if value is null. String(value) should not because obj.ToString() gets a string representation of obj

Simple Programming Statements

Constant declaration

```
const int;
```

Variable declaration

```
string message;
```

Assignment

```
message = "Hello";
```

Method call

```
Console.ReadLine();
```

Sequence of statements - grouped

```
{          }
```

Structured Programming Statements

If statement

```
If( condition ){ then } else { then }
```

Case statement

```
switch( variable ) {default: break;}
```

While loop:

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

Repeat loop

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

For loop

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

Declaring Methods

Declare a method with parameters:

```
public void HelloWorld(string message) { }
```

Declare a method that returns data:

```
public int Addition(int x, int y) { }
```

Pass by reference:

```
public void MyFunc(ref int x) {x = 20;}
(value passed is the memory location of argument so
the parameter passed may be modified by the method)
```

Boolean Operators and Other Statements

Comparison: equal, less, larger, not equal, less eq

```
==, <, >, !=, <=
```

Boolean: And, Or and Not

```
&&, ||, !
```

Skip an iteration of a loop

```
continue;
```

End a loop early

```
break;
```

End a method:

```
return;
```

Custom Types – create own data type

Classes

```
public class Message(string message) { }
```

Enumerations

```
enum Season {Summer, Autumn, Winer, Spring}
```

Structs

```
struct Car{
      public string model;
      public int year;
}
```

Arrays

Declaration

```
String[] words = new String[10];
Message[] messages = new Message[4];
```

Access

```
words[0]; messages[10];
```

Loop with index i

```
for(int i=0;i< word.length; i++) { }
```

For each loop

```
foreach (string word in words) { }
foreach (Message message in messages) { }
```

Programs and Modules

Creating a program

```
namespace program {
      class mainprogram {

      }
}
```

Using a class from a library

```
#include Swin;
```

Other Things

Reading from Terminal

```
Console.ReadLine();
```

Writing to Terminal

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

Comments

```
//
/*
  sdfsadfasdf
*/
```