C# complete course

```
Console.WriteLine (" welcome to c# ");
```

welcome to c#

```
class Program
{
    Public Static void Main (string[] args)
    {
        system.console.WriteLine ("Hello");
    }
}
```

Hello

class : is a keyword which is used to define class

Program : is the class name. A class is a blueprint or template from which objects are created.

Static : is a keyword which means object is not required to access static members. So it saves memory.

void :- is a return type of the method. It does not return any type of value.

Main :- is a method name. It is the entry point of c# program.

String[] args :- is used for command line arguments in c#.

System.Console.WriteLine("Hello");

Here System is the nameSpace. Console is the class define in System nameSpace. the WriteLine() is static method of console class which is used to write text on the console.

```
using System; namespace consoleApplication{
public class program
{
    public static void Main(String[] args)
    {
        Console.WriteLine("Hello World");
    }
}
}
```

Variable type :-

```
int i, j;
double d;
char ch;
float f;
```

Value Data Type

Pointer Data Type ⟶ int *a; // pointer to int

Reference Data Type ⟷ object, string, classes, Interface

& → Address operator (Determine the address of variable)

* ⟶ Indirection operator (Access the value of a Address)

"\t" → use for tab

"\n" → " " newline

Syntax c#

```
using System;
namespace firstproject
{
    class Program
    {
        static void Main(string[] args)
        {
            console.Write("hello");
            console.Writeline("I am good");
            console.Readkey();
        }
    }
}
```

3. Read & Readline

- Read
- Readline

```
console.Writeline("what is your name");
string name = Console.Readline();
Console.Writeline(name + "welcome here");
console.Readkey();
```

1. Console Application

```
Write       // write in same line   Console.Write("hello");
WriteLine   // Write in 2nd line
Read        // take a character
ReadLine    // Read a String
Readkey     // terminate the console by a key, it hold the console
```

2. Write, WriteLine and Readkey

variable (data holders)

- int
- String

] type of variable

```
String name;
int StudentNumber;
name = "kuntal"
StudentNumber = 12
```

```
Console.Write("hello")
Console.WriteLine("I am good")
Readkey();
```

4. Variables :-

variables types → [Datetime. 8byte, int- 4byte, String 2byte/char, double 16byte, float 4byte, Char 2byte, bool 1byte]

Syntax:
type name;
type name = value;

Example :-
char letter = 'A';
String name = "kuntal";
constant variable [const double Pi = 3.14159;

5. Naming Rules for creating variable :-

// same like c programming.

6. Variable conversions :-

Casting → Implicit casting

[byte → short → int → long → float → double
(small type to big type →)]

Explicit casting

```
double x = 24;
int y;
y = (int) x;   // casting double to int-
```

```
double x = 24;
String str;
str = Convert.ToString(x);
Console.WriteLine(str.GetType());
```

(A) Goto page (A)

operators ⇒

(Binary operators)
1. Arithmetic (+, -, *, /, %)
2. Relational (<, <=, >, >=, ==, !=)
3. Logical (&&, ||, !)
4. Bitwise (&, |, <<, >>, ~, ^)
5. Assignment (=, +=, -=, *=, /=)

6. unary (++, --)
7. Ternary (? :)

Control Statement ⇒

Syntax if-else

```
if (condition) {
    // code if condition True
}
else {
    // code if condition false
}
```

Input from user :=

```
string name = Console.ReadLine();      // by default take string input
int name = Convert.ToInt32(Console.ReadLine());   // string to int convert
```

Syntax if-else if-else

```
if (condition1) {
    // code be executed if condition1 true
}
else if (condition2) {
    // code be execute if condition2 true
}
```