ملخص .. C#

1)

- Input : (ReadLine();)
- Output : (Console.WriteLine("Hello"))

2) Data Type :- [int ⇒ 1,2,3], [float ⇒ 13f] [double ⇒ 1,5], [bool ⇒ true or false], [char ⇒ 'A', 'B'], [String ⇒ hello]

3) Operators

- Logical
  - [&& ⇒ and]
  - [|| ⇒ or]
  - [! ⇒ not]
- Arithmetic
  - [+] ⇒ Addition, [-] ⇒ Subtraction
  - [*] ⇒ Multiplication, [/] ⇒ Division
  - [%] ⇒ Modulus, [++] ⇒ Increment, x++
  - [--] ⇒ Decrement, x--

4) Conditional Statments

- if
- else if
- else
- ∴ Switch

- if ⇒ is specify a block of code to be executed, if a specified condition is True.

- else ⇒ is specify a block of code to be excuted, if a specfied condition is False.

- else if ⇒ to sepecify a new condition to test, if the first condition is False

- Switch ⇒ to specify many alternative blocks of code to be executed.

- if ⇒ if (20>18) { Console.WriteLine (" 20 is greater than 18 ");

- else ⇒ int time= 20; if (time<18) { Console. WriteLine (" Good day "); } else { Console. WriteLine (" Good evening "); }

- else if ⇒ int time= 22; if (time<10) { Console. WriteLine (" Good moring "); } else if (time<20) { Console.WriteLine (" Good day "); } else { Console. WriteLine (" Good evening "); }

▲ Variables ⇒ type veriblename = Value;

String name = "Riham";

▲ Constants ⇒ add Const keyword infornt of Variable type. which means unchangeable and read-only. Const int mynum = 15;

mynum = 20; //errors.

▲ C# for Loop : use when you know exactly how many times you want to Loop.

for (int i = 0; i<5; i++) {Console.WriteLine (i)}

▲ C# While Loop: loops through ablock of code as long as a specified Conditron is True.

int i=0 while(i<5) { Console.WriteLine(i); i++; }

▲ Do while loop s[Do while loop is variant of the While loop, this Loop will execute the Code block before checking if the Condition is true, then it repeats the Loop as long as the Condition is true.

int i=0 do { Console.WriteLine(i); i++}

while (i>5);

Nested operations

A C# Nested Loops :-

1] for Loop ⇒ You can use for Loop inside another for Loop. for (int i=0; i<8; i++)

for (int j=5; j<i; j++) {

Console.WriteLine("Good"); }

2] While Loop ⇒ You Can use While loop inside another while Loop. { it is not recommended to use because it is difficult to maintain and debug.

{ int x=1, y=2

while (x<4) { Console.WriteLine("hello", x);

x++;

wile (y<4) { Console.WriteLine("not", y);

y++; }}}

3] Do-While-Loop ⇒ You can use do while loop inside another do while loop.

int x = 1; do { Console.WriteLine("outloop", x);

int y=x; x++; do { Console.WriteLine("

Inloop ", x); y++; } while (y<4);

}while (x<4); }}

C# Arrays:-

- Array ⟶ used to store multiple values in single variable, it come with the Squarebackets.

String[] cars;

Example:- ⟶ int[] mynum = {10, 20, 30, 40};

String[] cars = {"BMW", "Camry", "Ford"};

⇒ Array start indexes with 0.

Example:- String[] cars = {"BMW", "Camry", "ford"};

cars[0] = "Opel";

Console.WriteLine(cars[0]); //output- BMW

- String[] cars = {"BMW", "Camry", "ford"};

Console.WriteLine(cars.Langth); //output 3

Functions C#

- Built-in Functions
  - Math Function
  - String Function
  - Date and Time Function
  - Array Function
  - Console Function
  - Type Conversion Function
  - Random Number Function
- User defined Functions
  - No Parameters, No Return V
  - Parameters, No Return Value
  - Parameters with Return V
  - Overloading Function
  - Optional Parameters
  - Named Parameters
  - Passing Parameters by V, Ref (Value, Reference)
  - Local Function.

Built-in Function

▲ Math Function :- [Math.Abs], [Math.Pow], [Math.Sqrt],
[Math.Max], [Math.Min], [Math.Round];

▲ Math.Max(x,y) To find the highest value of x and y.

```
Math.Max(5,10); //output 10
```

▲ Math.Min(x,y) To Find the lowest value of x and y.

```
Math.Min(5,10); //output 5
```

▲ Math.Sqrt(x) ⇒ To returns the Square root of x.

```
Math.Sqrt(64); //output 8
```

▲ Math.Abs(x) ⇒ To returns the absolute (positive) value of x:-

```
Math.Abs(-4.7); //out-put 4.7
```

▲ Math.Round() ⇒ rounds a number to nearest whole num.

```
Math.Round(9.99) //output 10
```

▲ Math.Pow ⇒ Returns the Value of x to power of y. [x^y]

```
Math.Pow(2,2) // output 4
```

▲ String Function :- [String.length], [String.Substring],
[String.ToUpper], [String.ToLower], [String.Trim],
[String.Replace], [String.Contains], [String.Split],

▲ String.Length ⇒ To get the length of String.

```
String txt = "ABCDEF";
Console.String = txt.Length //output- Length=6
```

▲ String.Substring :: Extracts SubString from string.

String abc = "ABCDEF";

Console.WriteLine("String : " + abc);

Console.WriteLine("SubString 1 :- " + abc.Substring(0,3));

//output Substring1 :- ABCD

▲ String.ToUpper :: Convert string to upperCase.

String upper = tex.ToUpper(); output "RIHAM";

▲ String.ToLower :: Convert string to lower.

string tow = tex.ToLower(); //output "riham";

▲ String Trim :: Removes leading and trailing white-space characters from a string.

String txt = "Riham ".Trim(); //output "Riham"

▲ String Replace :: Replaces all occurrences of a specified string with another string.

String su = "hello, Oman";

StringB Sb = new StringB(su); ~~Sb.Replace~~

Sb.Replace("hello", "B"); //output hellop, B

▲ String Contains :: Determins whether a string contains a specified substring.

bool contains = message.Contains("world");

// Contains= true.

▲ String.Split :: Splits a string into array of substring based on a delimiter.

String[] words = message.Split(' ');

A

A Date and Time Functions :- [DateTime.Now], [DateTime.Today], [DateTime.AddDays], [DateTime.ToString], [DateTime.Parse and DateTime.TryParse].

- DateTime.Now ⇒ To get the current date & Time.

DateTime Now = DateTime.Now;

- DateTime.Today ⇒ To get current date with the time components set to 00:00:00.

- DateTime.AddDays ⇒ Adds a specified number of date to DateTime object.

~~DateTime.AddDays~~ DateTime tomorrow = today.AddDays(1);

- DateTime.ToString ⇒ Converts the date and time to a string in specified format.

String formattedDate = new ToString("yyyy-MM-dd HH:mm:ss");

2025-03-3 15:30:00

- DateTime.Parse and DateTime.TryParse ⇒ Parse a string representation of a date and time into DateTime object. DateTime parsedDate = DateTime.Parse("2024-08-

bool success = DateTime.TryParse("2024-08-18", out DateTime result);

A Ar

▲ Array Functions:- [Array.Sort], [Array.Reverse], [Array.IndexOf], [Array.Resize].

- Array.Sort ⇒ Sort elements of array.

```
int[] numbers = {3, 2, 1, 1, 5};
Array.Sort(numbers); output {1, 1, 2, 3, 5}
```

- Array.Reverse ⇒ To reverses the sequence of the elements in array. int[] numbers = {3, 1, 5, 2};

```
Array.Reverse(numbers); // {5, 3, 2, 1}
```

- Array.IndexOf ⇒ Searches of specified object and returns the index of its first occurrence in array.

```
int index = Array.IndexOf(numbers, 4); // 1.
```

- Array.Resize : Changes the size of one-dimensional array.

```
Array.Resize(ref numbers, 10);
```

▲ Console Functions:- [Console.WriteLine], [Console.Write], [Console.ReadLine].

- Console.WriteLine ⇒ To add new line after write.

```
Console.WriteLine("hello, word");
```

- Console.Write ⇒ without new line

```
Console.Write("enter the name");
```

- Console.ReadLine :- Read the next Line

```
String name = Console.ReadLine();
```

△ Type Conversion functions:- [Convert.ToInt32], [Convert.ToDouble], [Convert.ToString], & [int.Parse and int.TryParse].

-Convert.ToInt32 :- to convert String or double to int integer.

```
String numberS = "123";
int numberS = Convert.ToInt32(numberS); // 123
```

-Convert.ToDouble ⇒ Convert value to double

```
String dString = "123,1";
double dstring = Convert.ToDouble(dString); 123,1
```

-Convert.ToString ⇒ Convert value to String.

```
int hum = 123;
string numString = Convert.ToString(hum); "123"
```

-int.Parse and int.TryParse ⇒ Convert a String repersentation of number to it integer equivalent.

```
int ParsedNumber = int.Parse("456"); 456.
bool Success = int.TryParse("456"); true.
```

△ Random Number Functions ⇒ [Random.Next], [Random.NextDouble].

-Random.Next ⇒ Returns random integer.

```
Random random = new Random();
int randomNumber = random.Next(); // any integer
int randomRange = random.Next(1,10); // (1,9)
```

-Random.NextDouble ⇒ return random floating 0.0 and 1.0

```
double randomDouble = random.NextDouble();
```

⇒

⇒ User Defined Functions: Funetions defined by users ~~wh~~ ~~wich~~ witch is not done by system.

1) Defining a function 2) Function Example

⇒

1) Defining Function ⇒

[access_modifier][return_type][Function_name]

([parameters])

{

// Function body

}

▲ access_modiffier : This determines which parts of the code can call the function.

▲ return_type : Specifies the type of value the function (int, String, void)

▲ funetion_name : The funetion name, it should be descriptive and follow C# naming conventions.

▲ parameters : A List of input parameters.

⇒ Funetion Example : [1] No Parameters, No Return value.

Example of function that donot take any parameters and do not return a value :

public void PrintHello()

{Console.WriteLine("Hello, world!"); } ⇒

This funetion print 'Hello, World!"

Errors in Programming:

① Syntax Errors

- when the code violates the Languages grammatical rules.
- detected by Compiler befor the program runs.

Ex:

```
int number = 10 ;
Console.WriteLine(number);
```

② Runtime Errors

- when all program is running
- Ex:

```
int divi = 0;
int res = 10 / divi;
Console.WriteLine(res);
```

- How to handle:

```
try { }
catch ( )
{Console.WriteLine ("errors:" + ex.Message);}
```

③ Logical Errors

- when all programe compiles and runs. but the output is not what you expected.

or

validations: