Tips for Logic programming

Working with Characters

1) Ascii Range

- 0 - 9 --> 48 to 57
- A - Z --> 65 to 90
- a - z --> 97 to 122
- Space --> 32

2) Extract Ascii value from a character

```
char c = 'A';
int ascii = c;
```

3) Convert Ascii to Character

```
int ascii = 65;
char c = (char)ascii;
```

4) Convert alphabet to upper or lower case

```
char c = 'a';
char ch = char.ToUpper(c);
char ch = char.ToLower(c);
```

5) To check whether the character is in upper case or lower case

```
char c = 'a';
if(char.IsUpper(c))
{
    // your logic
}
else
{
    // your logic
}
```

6) To check whether the given character is a letter or number

```
char c = '4';
// To check if c is alphabet
if(char.IsLetter(c))
{
    // your logic
}
// To check if c is number
else if (char.IsNumber(c))
{
    // your logic
}
// To check if c is either alphabet or number
if(char.IsLetterOrDigit(c))
{
    // your logic
}
```

Working with Strings

1) CompareTo

Compares this instance with a specified System.String object and indicates whether this instance precedes, follows, or appears in the same position in the sort order as the specified string.

```
Eg: string str = "Wipro Technologies";
int output =  str.CompareTo("XYZ"); // output --> - 1
int output = str.CompareTo("ABC"); // output --> 1
int output = str.CompareTo("Wipro Technologies"); // output --> 0
```

2) Contains

Returns a value indicating whether a specified substring occurs within this string. Returns true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

```
Eg: string str = "Wipro Technologies India Limited";
bool b1 = str.Contains(""); // output --> true
bool b2 = str.Contains("Tech"); // output --> true
bool b3 = str.Contains("abc"); // output --> false
```

3) CopyTo

CopyTo takes string characters and puts them into an array. It copies a group of characters from one source string into a character array of a certain size.

```
Eg: string value1 = "Wipro Technologies";
char[] array1 = new char[3];
// Copy the fifth, sixth, and seventh characters to the array.
value1.CopyTo(4, array1, 0, 3);
Console.WriteLine(array1); // output --> Tec
```

4) ElementAt

Returns the character at the specified index

```
Eg: string value1 = "Wipro Technologies";

value1.ElementAt(2); // output → p
```

5) EndsWith

Determines whether the end of this string instance matches the specified string.

```
string value1 = "Wipro Technologies";

value1.EndsWith("gies");// output → true
```

6) Equals

Determines whether this instance and a specified object, which must also be a System.String object, have the same value.

```
string value1 = "Wipro Technologies";

value1.Equals("Wipro"); // output → false
```

7) IndexOf

Reports the zero-based index of the first occurrence of the specified string in this instance.

```
string value1 = "Wipro Technologies";
value1.IndexOf("chno"); // output --> 8
```

```
value1.IndexOf('o'); // output --> 4 (first occurance of the
character index)
```

8) IndexOfAny

Reports the zero-based index of the first occurrence in this instance of any character in a specified array of Unicode characters.

```
string value1 = "Wipro Technologies";

char[] cArray = new char[] { 'd', 's' };

value1.IndexOfAny(cArray); // output --> 17 ('d' is not present in value1
so it checks for the index of 's')
```

9) Insert

Returns a new string in which a specified string is inserted at a specified index position in this instance.

```
string value1 = "Wipro Technologies";

value1.Insert(18, " India Pvt Ltd"); // output --> Wipro Technologies
India Pvt Ltd
```

10) LastIndex

Reports the zero-based index position of the last occurrence of a specified string within this instance.

```
string value1 = "Wipro Technologies";

value1.LastIndexOf("chno"); // output --> 8
```

Reports the zero-based index position of the last occurrence of a specified Unicode character within this instance.

```
string value1 = "Wipro Technologies";

value1.LastIndexOf('o'); // output --> 13
```

11) LastIndexOfAny

Reports the zero-based index position of the last occurrence in this instance of one or more characters specified in a Unicode array.

```
string value1 = "Wipro Technologies";

char[] cArray = new char[] { 'd', 's' };
value1.LastIndexOfAny(cArray); // output --> 17
```

12) Remove

```
string value1 = "Wipro Technologies";
```

Returns a new string in which all the characters in the current instance, beginning at a specified position and continuing through the last position, have been deleted.

```
string str3 = value1.Remove(3); // output --> "Wip"
```

Returns a new string in which a specified number of characters in the current instance beginning at a specified position have been deleted.

```
string str4 = value1.Remove(3, 7);

// output --> "Wipnologies"
```

13) Replace

```
string value1 = "Wipro Technologies";
```

Returns a new string in which all occurrences of a specified Unicode character in this instance are replaced with another specified Unicode character.

```
string str5 = value1.Replace('p', 'P');
  // output --> "WiPro Technologies"
```

Returns a new string in which all occurrences of a specified string in the current instance are replaced with another specified string.

```
string str6 = value1.Replace("Technologies", ""); // output --> "Wipro "
```

14) Split

Splits a string into substrings that are based on the characters in an array.

```
string value1 = "Wipro Technologies";

string[] arr = value1.Split(' ');
// output --> string array. arr[0] = "Wipro", arr[1] = "Technologies"
```

15) StartsWith

Determines whether the beginning of this string instance matches the specified string.

```
string value1 = "Wipro Technologies";

bool b = value1.StartsWith("Wip");

// output --> true
```

16) Substring

```
string value1 = "Wipro Technologies";
```

Retrieves a substring from this instance. The substring starts at a specified character position and continues to the end of the string.

```
string str7 = value1.Substring(8); // output --> "chnologies"
```

Retrieves a substring from this instance. The substring starts at a specified character position and has a specified length.

```
string str8 = value1.Substring(8, 3); // output --> "chn"
```

17) ToCharArray

Copies the characters in this instance to a Unicode character array.

```
            string value1 = "Wipro Technologies";
            char[] cArray2 = value1.ToCharArray();
// output --> cArray2[0] = 'W', cArray2[1] = 'i', cArray2[2] = p',......cArray2[17]
's'
```

18) ToLower

Returns a copy of this string converted to lowercase.

```
            string value1 = "Wipro Technologies";
            string str10 = value1.ToLower(); // output --> "wipro technologies"
```

19) ToUpper

Returns a copy of this string converted to uppercase.

```
            string value1 = "Wipro Technologies";
            string str9 = value1.ToUpper();
            // output --> "WIPRO TECHNOLOGIES";
```

20) Trim

Removes all leading and trailing white-space characters from the current System.String object.

```
            string value1 = "         Wipro Technologies         ";
            string str11 = value1.Trim(); // output --> "Wipro Technologies"
```

21) TrimEnd

Removes all trailing occurrences of a set of characters specified in an array from the current System.String object.

```
            string value1 = "         Wipro Technologies         ";
            string str12 = value1.TrimEnd();
            // output --> "       Wipro Technologies"
```

22) TrimStart

Removes all leading occurrences of a set of characters specified in an array from the current System.String object.

```
            string value1 = "         Wipro Technologies         ";
            string str13 = value1.TrimStart(); // output --> "Wipro Technologies      "
```

Working with Arrays

1) Declaring an array

```
int[] myarray = new int[5];
```

2) Initializing values to an array

```
int[] myarray = new int[5] { 100, 200, 300, 400, 500 };
       or
myarray[0] = 100;
myarray[1] = 200;
myarray[2] = 300;
myarray[3] = 400;
myarray[4] = 500;
```

3) Finding Number of elements in an array

```
int[] myarray = new int[5];

int length = myarray.Length;
```

4) Looping through an array

```
int[] myarray = new int[5] { 100, 200, 300, 400, 500 };

for(int i=0; i<myarray.Length;i++)
{
  Console.WriteLine(myarray[i]);
}

OR

foreach(int i in myarray)
{
  Console.WriteLine(i);
}
```

5) Sorting elements in an array

```
int[] myarray = new int[5] { 300, 100, 400, 200, 500 };

Array.Sort(myarray);
```

Note1: Always elements in an array will be sorted only in ascending order. To sort in descending order, after sorting, reverse the array.

Note2: By default only premitive datatypes like int, string, double, decimal… etc can be sorted using sort mentod. To sort Complex data types like classes we have to implement Icomparable interface.

6) Reversing an array

```
int[] myarray = new int[5] { 300, 100, 400, 200, 500 };
Array.Reverse(myarray);
```

7) To find the index of an element in an array

```
int[] myarray = new int[5] { 300, 100, 400, 200, 500 };

int index = Array.IndexOf(myarray, 200);
```

Working with Numbers

1) Extracting each digit from a number and sum them

```
int i = 12345;
int sum = 0;
while(i>0)
{
    int temp = i % 10;
    sum += temp;
    i = i / 10;
}
```

2) Finding the length of a number

```
int i = 12345;
int length = i.ToString().Length;
```