String Reverse-
I/p- Geeks.For.Geek | O/P- Geek.For.Geeks

```
String inp=”Geeks.For.Geeks”;
String[] str= inp.Split(“.”);
String res= “ ”;
for(int i= str.Length-1; i>=0; i--){
res += str[i]+”.”; }
Console.Write(res.Substring(0,res.Length-1));
```

I/P- GeeksForGeek; | O/P- keeG roF skeeG

```
String inp= “”;
Char[] arr= str.ToCharArray();
Array.Reverse(arr);
String reversedStr= new string(arr);
Console.Write(reversedStr);

//Or for loop
Public static string Reverse (string input){
char[] charArray= Input.ToCharArray();
string reversedString =String.Empty;
Int length, index;
while(index > -1){
reversedString = reversedString + charArray[index];
index--; }
return reversedString;
}
```

I/P- “I Like GFG; | O/P- GFG like I

```
String txt= “I Like GFG”;
Console.Write(string.Join(“ ”,text.Split(‘ ’).Reverse());
```

Reverse string words- Hey Ram-> Ram Hey

```
Void ReverseWords(string str){
Int i;
StringBuilder revSentence= new StringBuilder();
Int start= str.Length-1, end=str.Length-1;
while(start >0){
if(str[start] ==’ ‘){
i= start +1;
while(i <= end){
```

```
reverseSentence.Append(str[i]);
i++; }
reverseSentence.Append(‘ ‘);
end= start-1; }
Start--; }
for(i =0; i <= end; i++){
reverseSentence.Append(str[i]); }
Console.Write(reverseSentence.ToString();
}
```

I/P- Hello | O/P- olleH

```
Static void ReverseStr(string str){
Char[] charArray= str.ToCharArray();
for (int i=0; int j=str.Length-1; i<j; i++, j--){
charArray[ i ]= str[ j ];
charArray[ j ]= str[ i ]; }
String reversedStr= new string(charArray);
Console.Write(reversedStr);
```

Remove Duplicate- Hello -> Helo

```
Static void removeDuplicate(string str){
String result = string.Empty;
for( int i=0; i< str.Length; i++){
if(!result.Contains(str[ i ]){
Result += str[i];              }       }
Console.WriteLine(result);
```

Character Occurrence- Hello -> H-1, e-1, l-2,0-1

```
string message= “Hello”;
message = message.Replace(“ ”, string.Empty);
while(message.Length >0){
Console.Write(message[0] + “ : ”);
Int count =0;
for(int j=0; j<message.Length; j++)         {
if(message[0] == message[j]){ count++; }    }
Console.WrireLine(count)’
message= message.Replace(message[0].ToString(), string.Empty);
```

Possible substring- abcd=> a, ab, abc, abcd…

```
Void FindSubStrstring (str){
for(int i=0; i<str.Length; ++i){
StringBuilder subStr= new StringBuilder(str.Length- i);
for(int j= i; j < str.Length; ++j){
substring.Append(str[ j ]);
Console.Write(subString + “ “);
} } }
```

Make reverse= my name is -> ym eman si

```
class Program {
    static void Main(string[] args) {
        Console.Write("Enter a String : ");
        string originalString = Console.ReadLine();
        StringBuilder reverseWordString = new StringBuilder();
        List<char> charlist = new List<char>();
        for (int i = 0; i < originalString.Length; i++) {
            if (originalString[i] == ' ' || i == originalString.Length - 1){
                if (i == originalString.Length - 1)
                    charlist.Add(originalString[i]);
                for (int j = charlist.Count - 1; j >= 0; j--) {
                    reverseWordString.Append(charlist[j]); }
                reverseWordString.Append(' ');
                charlist = new List<char>();
        }
            else    {  charlist.Add(originalString[i]);   }
        }
        Console.WriteLine($"Reverse Word String : {reverseWordString.ToString()}");
        Console.ReadKey();
    }
}
```

SHORT Solution

```
Static string MakeReverse( string str){
char[ ] charArray = str.ToCharArray();
Array.Reverse(charArray);
str= new String(charArray);
String[ ] rev = str.Split(‘ ‘);
Array.Reverse(rev);
Return string.Join(“ ”, rev);
}
```

Vowels, const count-

```
String str= “Avengers”;
String upstr= str.ToUpper();
Int vowelsCount=0, constCount=0;
for(int i=0; i< upStr.Length; i++){
if(upStr == ‘A’ || upStr ==’E’ || upStr==’I’ || upStr==’O’ || upStr==’U’ ){
```

vowelsCount++; }
Else constCount++; }
Console.WriteLine("Vowels are {0}", vowelsCount);

Swapping - 3rd var
Int temp=0, n1 = , n2=
Temp = n1 ; n1 = n2; n2 = temp;
2 variable - A = A+ B; B= A-B; A= A-B;

----------\

replaces all vowels in a string with a specified character

```
char str[50], ch, i;
   Console.WriteLine("Enter str:");
 string str = Console.ReadLine();
Console.WriteLine("Enter special char");
string ch= Console.ReadLine();
   for(i=0; str[i]!='\0'; i++)
    {   if(str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o'
         || str[i]=='u' || str[i]=='A' || str[i]=='E' || str[i]=='I'  ||
str[i]=='O' || str[i]=='U')
        {  str[i] = ch;
        } }
Console.WriteLine("New String (after replacing vowel with", str);
```

Shift All The Zeros To The End Of An Array

Input : arr[] = {1, 2, 0, 4, 3, 0, 5, 0};
Output : arr[] = {1, 2, 4, 3, 5, 0, 0, 0};

```
class Program  {
      static void Main(string[] args){
        int[] array = new int[] { 3, 0, 0, 1, 2, 0, 4, 0 };
        int[] result = Approach1(array);
        //result is [ 3, 1, 2, 4, 0, 0, 0, 0 ]        }
static int[] Approach1(int[] array)      {
 for (int i = 0; i < array.Length; i++) {
            //skip non-zero elements
            if (array[i] != 0) continue;
   //look for the nearest non-zero
   for (int j = i + 1; j < array.Length; j++) {
       if (array[j] == 0)   continue;
        //swap it with our zero
```

```
            array[i] = array[j]; array[j] = 0;
                break;    }
}      return array;   }}
```

find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string "".

Example 1: Input: strs = ["flower","flow","flight"] **Output:** "fl"

```
public string LongestCommonPrefix(string[] strs) {
        if (strs == null || strs.Length == 0)
        {    return ""; }
        var prefix = strs[0];
        foreach (var s in strs)
        { while (s.IndexOf(prefix) != 0)
            {prefix = prefix.Substring(0,prefix.Length - 1-0);
            } }
        return prefix;     }
```

find an element such that sum of right side element is equal to sum of left side

Input: 1 4 2 5 0 Output: 2

Explanation: If 2 is the partition, subarrays are : [1, 4] and [5]

```
class GFG{
// Function to compute partition
static int findElement(int []arr, int size)
{
      int right_sum = 0, left_sum = 0;
      left_sum = 0;    // Maintains left cumulative sum
      right_sum = 0; // Maintains right cumulative sum
      int i = -1, j = -1;
for(i = 0, j = size - 1; i < j; i++, j--)
{  left_sum += arr[i];
   right_sum += arr[j];
 while (left_sum < right_sum && i < j)
  {  i++;    // Keep moving i towards center until  // left_sum is found lesser than right_sum
   left_sum += arr[i]; }
 while (right_sum < left_sum && i < j)
 {   j--;
  right_sum += arr[j]; }
}
if (left_sum == right_sum && i == j)
return arr[i];
else { return -1; }
```

```
}
// Driver code
public static void Main(String []args)  {
      int []arr = { 2, 3, 4, 1, 4, 5 };
      int size = arr.Length;
      Console.WriteLine(findElement(arr, size));  }
}
```