**Exercise 6**

**String, StringBuilder and Regular Expressions**

**Objective**

To create a C# program to demonstrate various string handling methods and to write various patterns to validate the given input through regular expression.

**String**

System.String is a class specifically designed to store a string and allow a large number of operations on the string. A String object is immutable, i.e. a String cannot be changed once created. Every time when you use any of the methods of the System.String class, then you create a new string object in memory.

**Creating a String Object**

You can create string object using one of the following methods:

* By assigning a string literal to a String variable
* By using a String class constructor
* By using the string concatenation operator (+)
* By retrieving a property or calling a method that returns a string
* By calling a formatting method to convert a value or object to its string representation

Some of the C# String functions are as follows:

|  |  |
| --- | --- |
| String Functions | Definitions |
| CompareTo() | Compare two strings and returns integer value as output. It returns 0 for true and 1 for false. |
| Clone() | Make clone of string |
| Contains() | The C# Contains method checks whether specified character or string is exists or not in the string value. |
| EndsWith() | This EndsWith Method checks whether specified character is the last character of string or not. |
| Equals() | The Equals Method in C# compares two string and returns Boolean value as output. |
| GetHashCode() | This method returns HashValue of specified string. |
| GetType() | It returns the System.Type of current instance. |
| GetTypeCode() | It returns the Stystem.TypeCode for class System.String. |
| IndexOf() | Returns the index position of first occurrence of specified character. |
| ToLower() | Converts String into lower case based on rules of the current culture. |
| ToUpper() | Converts String into Upper case based on rules of the current culture. |
| Insert() | Insert the string or character in the string at the specified position. |
| LastIndexOf() | Returns the index position of last occurrence of specified character. |
| Length | It is a string property that returns length of string |
| Remove() | This method deletes all the characters from beginning to specified index position. |
| Replace() | This method replaces the character. |
| Split() | This method splits the string based on specified value. |
| StartsWith() | It checks whether the first character of string is same as specified character. |
| Substring() | This method returns substring. |
| ToCharArray() | Converts string into char array |
| Trim() | It removes extra whitespaces from beginning and ending of string. |

**String Builder**

StringBuilder is a dynamic object. It doesn’t create a new object in the memory but dynamically expands the needed memory to accommodate the modified or new string. StringBuilder can be declared and initialized the same way as class.

**Example**

StringBuilder sb = new StringBuilder(); //string will be appended later

StringBuilder sb =new StringBuilder("Hello World!");

StringBuilder sb =new StringBuilder(20); //string will be appended later

StringBuilder s = new StringBuilder(“Hello World”, 20);

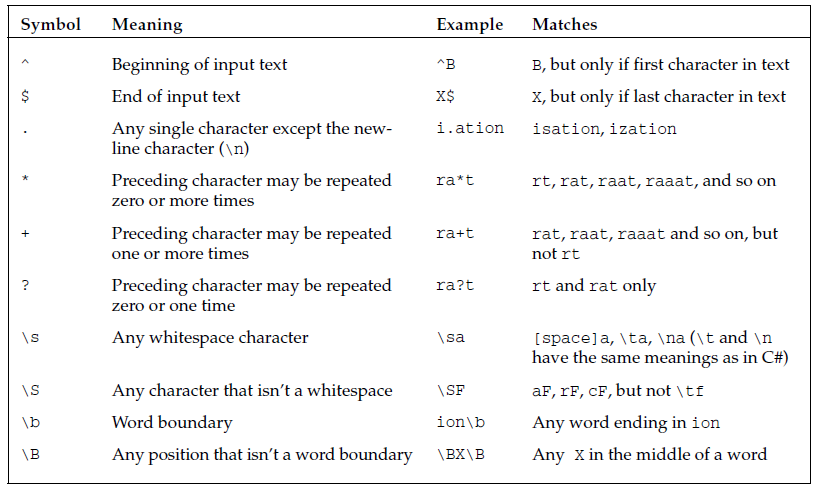
Some of the C# StringBuilder functions are as follows:

|  |  |
| --- | --- |
| Append(string value) | The method appends the string with a newline at the end. |
| AppendLine() | The method appends the string with a newline at the end. |
| AppendFormat() | The method uses to format the input string into the specified format and then append it. This method also appends text to the end of the StringBuilder object. |
| Insert(int index, string value) | The method inserts the string at the specified index in StringBuilder object. |
| Length | It is also a StringBuilder property that returns the length of the string |
| Remove(int start, int length) | This method removes the specified number of characters from the current StringBuilder object. The removing process beginning at a specified index and extends up to another specified index. |
| Replace(old\_val, new|\_val) | This method is used to replace characters within the StringBuilder object with another specified character. |

**Regular Expression:**

* Regular expressions are Patterns that can be used to match strings.
* To locate substrings within a large string expression
* The namespace is System.Text.RegularExpressions
* The regular expressions language is designed specifically for string processing
* With regular expressions, you can perform quite sophisticated and high - level operations on strings. For example, you can:
  + Identify (and perhaps either flag or remove) all repeated words in a string (for example, “ The computer books books ” to “ The computer books ” )
  + Convert all words to title case (for example, “ this is a Title ” to “ This Is A Title ” )
  + Convert all words longer than three characters to title case (for example, “ this is a Title ” to “ This is a Title ” )
  + Ensure that sentences are properly capitalized
  + Separate the various elements of a URI (for example, given http://www.wrox.com , extract the protocol, computer name, file name, and so on)
* Of course, all of these tasks can be performed in C# using the various methods on System.String and System.Text.StringBuilder . However, in some cases, this would require writing a fair amount of C# code. If you use regular expressions, this code can normally be compressed to just a couple of lines.
* You instantiate a System.Text.RegularExpressions.RegEx object
* Pass it the string to be processed, and pass in a regular expression

Some of the Regular Expression symbols are as follows



**Example Program**

Using System;

using System.Text.RegularExpressions;

class myRegEx

{

static void Main()

{

string text = "Here is the text!";

Regex regex = new Regex("\b[a-z]+\b");

if(regex.isMatch(text))

Console.WriteLine(“Matched”);

else

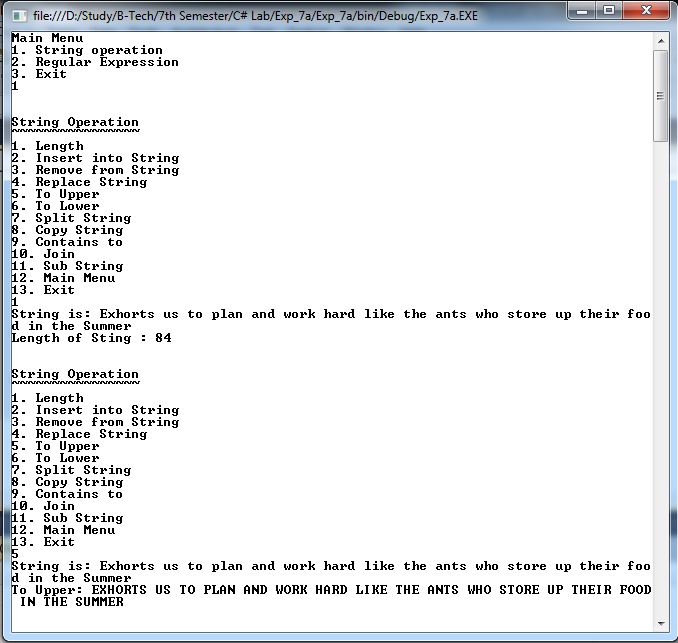
Console.WriteLine(“Not Matched”)

}

}

**Questions**

**Section1:** Create a C# console application to get the necessary strings from the user and perform all the string operations. A sample screenshot is given below.

****

**Section 2:** Create a C# console application to get the necessary strings from the user store it in the StringBuilder class and perform the following operations.

**String Builder Operations**

**1. Length**

**2.** **Append**

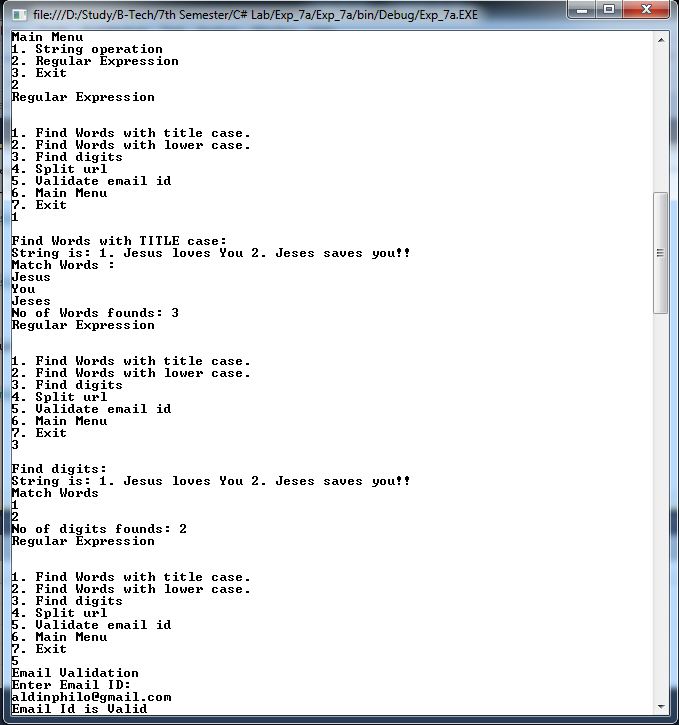
**3. AppendLine**

**4. AppendFormat**

**5. Insert**

**6. Remove**

**Section 3:** Create a C# console application to get a string from the user and check for the following conditions using regular expression. A sample screenshot is given below.



**Section 4:** Create a program using String or StringBuilder class to read a sentence as input and do the following operations

* + Display the number of alphabetical letters and digits.
  + Display the number of vowels and consonants separately
  + Display the number whitespaces
  + Display the number of special characters limited to this (#!@$&\*)

**For example,**

Input

hello\* world& 1abc 34$

Output

Number of Alphabets: 13

Number of digits: 3

Number of Vowels: 4

Number of Consonants: 9

Number of White Spaces: 3

Number of Special Characters: