**COMSATS UNIVERSITY ISLAMABAD**

**Logo, company name

Description automatically generated**

**ATTOCK CAMPUS**

**Submitted By**

Nadeem Mumtaz

**Registration No**

FA20-BCS-060

**Submitted To**

Sir Bilal Bukhari

**Course Title**

Compiler Construction

**Date**

**25-10-2023**

**Q1:Briefly describe the regex library of C#**

**Ans:**

The regex library in C# is part of the .NET Framework and provides support for working with regular expressions, which are powerful tools for pattern matching and manipulation of strings. The primary class for regex operations in C# is System.Text.RegularExpressions.Regex.

Here's a brief overview and an example of using the regex library in C#:

Regular Expression Patterns: Regular expressions are patterns used to match character combinations in strings. For example:

\d+: Matches one or more digits.

[A-Z]+: Matches one or more uppercase letters.

\w{3,5}: Matches word characters (letters, digits, or underscores) with a length of 3 to 5 characters.

@\w+\.\w+: Matches email addresses.

Matching: You can use the Regex.IsMatch method to check if a string matches a pattern:

string pattern = @"\d+";

string input = "12345";

bool isMatch = Regex.IsMatch(input, pattern);

// isMatch will be true in this case

**Search and Capture:** The Regex.Match method searches for the first occurrence of a pattern and captures the matched text:

csharp

Copy code

string pattern = @"\w+";

string input = "Hello, world!";

Match match = Regex.Match(input, pattern);

if (match.Success)

{

string capturedText = match.Value;

// capturedText will be "Hello" in this case

}

**Replacing:** The Regex.Replace method allows you to replace matched patterns in a string:

csharp

Copy code

string pattern = @"\d+";

string input = "The price is $10.50";

string replacedText = Regex.Replace(input, pattern, "[NUMBER]");

// replacedText will be "The price is $[NUMBER].[NUMBER]"

Splitting: You can use the Regex.Split method to split a string based on a pattern:

csharp

Copy code

string pattern = @"\s+"; // Matches one or more whitespace characters

string input = "This is a sentence.";

string[] parts = Regex.Split(input, pattern);

// parts will be ["This", "is", "a", "sentence."]

Options: You can specify options to control the matching behavior, such as case-insensitivity and multi-line mode.

csharp

Copy code

string pattern = "hello";

string input = "Hello, world!";

bool isMatch = Regex.IsMatch(input, pattern, RegexOptions.IgnoreCase);

// isMatch will be true in this case

The regex library in C# is powerful and flexible, allowing you to perform various string manipulation tasks based on complex patterns. It's essential for tasks like data validation, text parsing, and data extraction from text.