Q1. What is the benefit of regular expressions?

*Ans*:

1. Flexible pattern matching: Regex allows you to define complex patterns for matching and searching text data. You can use regex to match specific characters, words, patterns, or groups of characters in a text string.

2. Time saving: With regex, you can perform complex text operations with just a few lines of code. This can save a lot of time and effort compared to manually searching or manipulating text data using basic string operations.

3. Text validation: Regular expressions can be used to validate user input, such as email addresses or phone numbers, and ensure that it meets certain criteria or patterns

4. Text cleaning: Regex can be used to clean or pre-process text data, such as removing unnecessary characters, converting text to lowercase, or extracting specific information from a text string

Q2. Describe the difference between the effects of "(ab)c+" and "a(bc)+." Which of these, if any, is the unqualified pattern "abc+"?

***Ans***: 1. "(ab)c+" matches strings starting with "ab" followed by one or more "c".

2. "a(bc)+" matches strings starting with "a" followed by one or more "bc".

3. "abc+" matches strings starting with "ab" followed by one or more "c".

The unqualified pattern "abc+" matches strings that start with the characters "a" and "b" followed by one or more occurrences of the character "c". For example, it would match the strings "abc", "abcc", "abccc", and so on

Q3. How much do you need to use the following sentence while using regular expressions?

import re

***Ans***: You need to use the ‘import re’ statement at the beginning of your Python script or program whenever you want to use regular expressions in your code. This statement imports the Python re module, which provides functions for working with regular expressions.

Q4. Which characters have special significance in square brackets when expressing a range, and under what circumstances?

***Ans***:

the ^, -, and \ characters have special significance in square brackets when expressing a range of characters, and they are used for negation, range, and escaping metacharacters, respectively.

Q5. How does compiling a regular-expression object benefit you?

***Ans***:

compiling a regular-expression object in Python provides benefits such as improved performance, easier maintenance, increased flexibility, and improved error handling.

Q6. What are some examples of how to use the match object returned by re.match and re.search?

***Ans***:

1. import re

# Find the first occurrence of "cat" in the string

match\_obj = re.search('cat', 'The quick brown cat jumps over the lazy dog')

# Retrieve the matched text

matched\_text = match\_obj.group()

print(matched\_text)

# Output: "cat"

2. import re

# Match a string that starts with "Hello" and ends with "world"

match\_obj = re.match('Hello (.\*?) world', 'Hello my friend world')

# Retrieve the first captured group

captured\_text = match\_obj.group(1)

print(captured\_text)

# Output: "my friend"

Q7. What is the difference between using a vertical bar (|) as an alteration and using square brackets as a character set?

***Ans***:

The vertical bar is used to specify alternation between different patterns, while square brackets are used to specify a character set that matches any one character from a set of possible characters.

Q8. In regular-expression search patterns, why is it necessary to use the raw-string indicator (r)? In   replacement strings?

***Ans***:

In regular-expression search patterns, the raw-string indicator (r) is used to treat the backslash (\) character as a literal character, rather than as an escape character. This is because the backslash is a special character in regular expressions and is used to specify special characters and character classes.