Q1. What is the benefit of regular expressions?

A Regular Expression is used for identifying a search pattern in a text string. It also helps in finding out the correctness of the data and even operations such as finding, replacing and formatting the data is possible using Regular Expressions.

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

Plus (+) symbol matches one or more occurrences of the regex preceding the + symbol.

(ab)c+ will match for: abc, abcc, abcccc ……..and so on

a(bc)+ will match for: abc, abcbc, abcbcbc ……. and so on

The first pattern “(ab)c+” is the unqualified patten “abc+”

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

import re

A Regular Expressions (RegEx) is a special sequence of characters that uses a search pattern to find a string or set of strings. It can detect the presence or absence of a text by matching it with a particular pattern, and also can split a pattern into one or more sub-patterns. Python provides a re module that supports the use of regex in Python. Its primary function is to offer a search, where it takes a regular expression and a string. Here, it either returns the first match or else none.

So we need to import re module when we need to use it.

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

The range is an immutable sequence, so that values can be accessed by passing indexes in the square brackets [].

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

**Python Regex Sets & Ranges:**

Sets:

For example, [abc] means any of three characters. 'a', 'b', or 'c'. The [abc] is called a set. And you can use the set with regular characters to construct a search pattern.

For example, the following program uses the pattern licen[cs]e that matches both license and licence:

import re

s = 'A licence or license'

pattern = 'licen[cs]e'

matches = re.finditer(pattern, s)

for match in matches:

print(match.group())

The pattern licen[cs]e searches for:

* licen
* then one of the letters [cs]
* then e.

Therefore, it matches license and licence.

Ranges:

When a set consists of many characters in e.g., from a to z or 1 to 9, it’ll tedious to list them in a set. Instead, you can use character ranges in square brackets. For example, [a-z] is a character in the range from a to z and [0-9] is a digit from 0 to 9.

Also, you can use multiple ranges within the same square brackets. For example, [a-z0-9] has two ranges that match for a character that is either from a to z or a digit from 0 to 9.

Similarly, you can use one or more character sets inside the square brackets like [\d\s] means a digit or a space character.

Likewise, you can mix the character with character sets. For example, [\d\_] matches for a digit or an underscore.

Excluding sets & ranges:

To negate a set or a range, you use the caret character (^) at the beginning of the set and range. For example, the range [^0-9] matches any character except a digit. It is the same as the character set \D.

Notice that regex also uses the caret (^) as an anchor that matches at the beginning of a string. However, if you use the caret (^) inside the square brackets, the regex will treat it as a negation operator, not an anchor.

The following example uses the caret (^) to negate the set [aeoiu] to match the consonants in the string 'Python':

import re

s = 'Python'

pattern = '[^aeoiu]'

matches = re.finditer(pattern, s)

for match in matches:

print(match.group())

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

The re.search() and re.match() both are functions of re module in python. These functions are very efficient and fast for searching in strings. The function searches for some substring in a string and returns a match object if found, else it returns none.

There is a difference between the use of both functions. Both return the first match of a substring found in the string, but re.match() searches only from the beginning of the string and return match object if found. But if a match of substring is found somewhere in the middle of the string, it returns none.

While re.search() searches for the whole string even if the string contains multi-lines and tries to find a match of the substring in all the lines of string.

# import re module

import re

Substring ='string'

String1 ='''We are learning regex with geeksforgeeks

regex is very useful for string matching.

It is fast too.'''

String2 ='''string We are learning regex with geeksforgeeks

regex is very useful for string matching.

It is fast too.'''

# Use of re.search() Method

print(re.search(Substring, String1, re.IGNORECASE))

# Use of re.match() Method

print(re.match(Substring, String1, re.IGNORECASE))

# Use of re.search() Method

print(re.search(Substring, String2, re.IGNORECASE))

# Use of re.match() Method

print(re.match(Substring, String2, re.IGNORECASE))

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

The vertical bar symbol acts as an OR operator and matches the values to the left and right of the vertical bar. For example, the regular expression Jack|Jill matches Jack and Jill . The backslash symbol acts as an escape sequence. Use it when you want search for a regular expression symbol.

[abc] means any of three characters. 'a', 'b', or 'c'. The [abc] is called a set. And you can use the set with regular characters to construct a search pattern.

For example, the following program uses the pattern licen[cs]e that matches both license and licence:

import re

s = 'A licence or license'

pattern = 'licen[cs]e'

matches = re.finditer(pattern, s)

for match in matches:

print(match.group())

The pattern licen[cs]e searches for:

• licen

• then one of the letters [cs]

• then e.

Therefore, it matches license and licence

The vertical bar is a regex "or" means "a or b". Square brackets are a character class meaning "any character from a or b.

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

For regular expression patterns, the answer is to utilize Python's raw string notation; backslashes are not treated differently in a string literal prefixed with "r." Therefore, r"\n" is a two-character string made up of the letters "" and "n," whereas "\n" is a one-character string made up of the letter "n."