## Lecture 4-string in java

1. What is the purpose of the length() method in Java's String class?

- A) To return the number of characters in a String

- B) To split a String into an array

- C) To find the index of a character in a String

- D) To convert a String to uppercase

Answer: A

2. Which method converts a String into an array of characters?

- A) charAt()

- B) toCharArray()

- C) split()

- D) indexOf()

Answer: B

3. What does the regex ^hello$ match?

- A) Strings that start with "hello"

- B) Strings that end with "hello"

- C) Strings that are exactly "hello"

- D) Any string containing "hello"

Answer: C

4. What does the regex \bcat\b match?

- A) The word "cat" as a standalone word

- B) "cat" at the beginning of a word

- C) "cat" at the end of a word

- D) Any occurrence of "cat"

Answer: A

5. In Java, what does the indexOf(String str) method do?

- A) Splits a String into an array using the given delimiter

- B) Finds the first occurrence of a substring within a String

- C) Converts a String to lowercase

- D) Replaces characters in a String

Answer: B

6. What does the regex [a-zA-Z0-9] match?

- A) Only lowercase letters

- B) Only uppercase letters

- C) Any letter or digit

- D) Only digits

Answer: C

7. What is returned by split(" +") when applied to "Can you hear me?Hello, hello?"?

- A) An array containing words with extra spaces included

- B) An array containing exactly 6 words without extra spaces

- C) An array containing characters split by space

- D) An error due to invalid regex

Answer: B

8. Which quantifier matches zero or more occurrences of a character in regex?

- A) +

- B) \*

- C) {min,max}

- D) ?

Answer: B

9. Which quantifier matches one or more occurrences of a character in regex?

- A) +

- B) \*

- C) {min,max}

- D) ?

Answer: A

10. What does the regex [0-9]{3}[ -]?[0-9]{3}[ -]?[0-9]{4} match?

- A) Any string with exactly 10 digits in sequence

- B) Phone numbers with optional spaces or hyphens as separators (e.g., 1234567890, 123 456 7890, 123-456-7890)

- C) Strings with three digits followed by spaces only

- D) Strings with special characters only

Answer: B

11. Which built-in character class matches any digit?

- A) \D

- B) \w

- C) \d

- D) \s

*Answer: C*

12. How can you match the literal dot (.) character in regex?

- A) Use just . directly in the pattern.

- B) Use \. to escape it.

- C) Use [.].

- D) Both B and C are correct.

Answer: D

13. Which quantifier matches between 2 and 4 occurrences of a pattern?

- A) {2,4}

- B) {2,}

- C) {,4}

- D) {4}

Answer: A

14. What is the output of calling getTokens("[a-z]+") on "Splitting strings is fun!"?

Assume it extracts sequences of lowercase letters.

- A) ["Splitting", "strings", "is", "fun"]

- B) ["splitting", "strings", "is", "fun"]

- C) ["strings", "is", "fun"]

- D) ["splittingstringsisfun"]

Answer: B

15. What does the regex (abc|def|ghi) match?

- A: Only "abc"

- B: Only "def"

- C: Any one of "abc," "def," or "ghi"

- D: All three combined in sequence.

Answer: C

16. Which of the following lines of code correctly assign a String containing the text "My String" to the variable 'text'? ( ) (Select all correct options.)

A.  
String text = new String("My ");  
text += new String("String");

B.  
String s1 = "My String";  
String text = s1;

C.  
String text = "My ";  
String s2 = "String";  
text = text + s2;

D.  
String text = "My ";  
text.concat("String");

ANS: ABC

Wrong choice D: The concat() method does not modify the original string because strings in Java are immutable. It creates a new string, but since the result is not assigned back to text, the value of text remains "My ".

17. Assume String variable d contains the text:

"lalalaaaa! lala, la!"

What array does the following call return?

d.getTokens("la+");

A. ["lalala", "lala", "la"]

B. ["la", "la", "la", "la", "la", "la"]

C. ["laaaaaa", "lala", "la"]

D. ["la", "la", "laaaa", "la", "la", "la"]

ANS: D

The call d.getTokens("la+") uses the regular expression la+ to find all matches in the text "lalalaaaa! lala, la!". The regex la+ matches the letter "l" followed by **one or more** "a" characters, with greedy matching (capturing the longest possible sequence each time). This results in the array **["la", "la", "laaaa", "la", "la", "la"]**.

18. Assume String variable d contains the text:

"lalalaaaa! lala, la!"

What array does the following call return?

d.getTokens("(la)+");

A. ["lalala", "lala", "la"]

B. ["la", "la", "la", "la", "la", "la"]

C. ["laaaaaa", "lala", "la"]

D. ["la", "la", "laaaa", "la", "la", "la"]

ANS: A

The call d.getTokens("(la)+"); produces three tokens, each being the longest possible match from that segment of the string,

19. Assume String variable s contains the text

"%one%%two%%%three%%%%"

Which of the following calls will return the String array as follows:

["%", "%%", "%%%", "%%%%"]

A. s.getTokens("%+");  
B. s.getTokens("[a-z]+");  
C. s.getTokens("one|two|three");  
D. s.getTokens("[one,two,three]");

ANS: A

20. Assume String variable s contains the text

"%one%%two%%%three%%%%"

Which of the following calls will return the String array as follows:

["%", "%%", "%%%", "%%%%"]

A. s.split("%+");  
B. s.split("[a-z]+");  
C. s.split("one|two|three");  
D. s.split("[one,two,three]");

ANS: B C

The string s = "%one%%two%%%three%%%%" contains alphabetic words (one, two, three) separated by sequences of % characters. To split the string into ["%", "%%", "%%%", "%%%%"], we need to **split at the alphabetic words**, preserving the % sequences.

**Option B: s.split("[a-z]+")**

* **Regex**: [a-z]+ matches any sequence of lowercase letters (e.g., one, two, three).
* **Effect**: Splits the string at the alphabetic words, leaving the % sequences between them as array elements.
* **Result**: ["%", "%%", "%%%", "%%%%"].

**Option C: s.split("one|two|three")**

* **Regex**: one|two|three matches the exact words one, two, or three using the OR operator.
* **Effect**: Directly splits the string at these specific words, preserving the % sequences.
* **Result**: ["%", "%%", "%%%", "%%%%"].

**Incorrect Options**

**Option A: s.split("%+")**

* **Regex**: %+ matches sequences of % characters.
* **Effect**: Splits the string at % sequences, leaving the alphabetic words (one, two, three) as array elements.
* **Result**: ["", "one", "two", "three", ""].

**Option D: s.split("[one,two,three]")**

* **Regex**: [one,two,three] matches individual characters (o, n, e, ,, t, w, etc.), not the full words.

21. Assume String variable d contains the text:  
"one (1), two (2), three (3)"

Which of the following calls to getTokens will return the list of Strings as follows:  
["one", "(1)", "two", "(2)", "three", "(3)"]

A. d.getTokens("[^, ]+")  
B. d.getTokens("[^,]+")  
C. d.getTokens("[a-z()0-9]+")  
D. d.getTokens("[a-z]+|[(][0-9]+[)]")

ANS: A C D

**Option A: [^, ]+**

* Matches any sequence of characters **except commas or spaces**.
* Splits the text at commas and spaces, capturing "one", "(1)", "two", "(2)", "three", and "(3)" as tokens.

**Option C: [a-z()0-9]+**

* Matches lowercase letters, digits, parentheses, or combinations of these.
* Captures "one" (letters), "(1)" (parentheses + digits), "two", etc., splitting at commas and spaces since they are excluded from the pattern.

**Option D: [a-z]+|[(][0-9]+[)]**

* Uses alternation to match either lowercase letters **or** parenthetical numbers (e.g., (1)).
* Directly isolates "one", "(1)", "two", etc., by treating letters and parenthetical numbers as separate token types.

**Why Option B Fails**

* [^,]+ splits at commas but **includes spaces**, resulting in tokens like "one (1)" and " two (2)" (with leading spaces).