CodInClub

Powered By BridgeLabz

Best Programming Practice

1. Use Variables including for Fixed, User Inputs, and Results
2. Use Methods instead of writing code in the main() function
3. Proper naming conventions for all variables and methods
4. Proper Program Name and Class Name
5. Handle Checked and Unchecked Exceptions wherever possible
6. Proper Method Name which indicates action taking inputs and providing result

Sample Program 1: Create a program to find all the occurrences of a character in a string using charAt() method

a. Take user input for the String and occurrences of the Character to find
b. Write a method to find all the occurrences of the characters.
    i. The logic used is to first find the number of occurrences of the character and
    ii. then create an array to store the indexes of the character
c. Call the method in the main and display the result

Java

```java
// Program to find all the occurrences of a character in a string
import java.util.Scanner;
class StringAnalyzer {
  // Method to find all the index of a character in a string using charAt()
  // method and return them in an array
  public static int[] findAllIndexes(String text, char ch) {
    // The count is used to find the number of occurrences of the character
    int count = 0;
    for (int i = 0; i < text.length(); i++) {
        if (text.charAt(i) == ch) {
            count++;
        }
    }

    // Create an array to store the indexes of the character
    int[] indexes = new int[count];
    int j = 0;
    for (int i = 0; i < text.length(); i++) {
        if (text.charAt(i) == ch) {
            indexes[j] = i;
            j++;
        }
    }
    return indexes;
```

```
  }
  public static void main(String[] args) {
    // Take user input for Text and Character to check Occurrences
    Scanner sc = new Scanner(System.in);
    System.out.print(Enter a text: ");
    String text = sc.nextLine();
    System.out.print("Enter a character to find the occurrences: ");
    char ch = sc.next().charAt(0);

    // Find the occurrences of the character
    int[] indexes = findAllIndexes(text, ch);

    // Display the occurrences of the character
    System.out.println("Indexes of the character '" + ch + "': ");
    for (int i = 0; i < indexes.length; i++) {
        System.out.print(indexes[i] + " ");
    }
  }
}
```

Level 1 Practice Programs

1. Write a program to compare two strings using the ***charAt()*** method and check the result with the built-in String ***equals()*** method

   Hint =>

   a. Take user input using the ***Scanner*** ***next()*** method for 2 String variables
   b. Write a method to compare two strings using the ***charAt()*** method and return a boolean result
   c. Use the String Built-In method to check if the results are the same and display the result

2. Write a program to create a substring from a String using the ***charAt()*** method. Also, use the String built-in method ***substring()*** to find the substring of the text. Finally Compare the the two strings and display the results

   Hint =>

   a. Take user input using the ***Scanner*** ***next()*** method to take the String variable and also the start and the end index to get the substring from the given text
   b. Write a method to create a substring from a string using the ***charAt()*** method with the string, start, and end index as the parameters
   c. Write a method to compare two strings using the charAt() method and return a boolean result
   d. Use the String built-in method substring() to get the substring and compare the two strings. And finally display the result

3. Write a program to return all the characters in a string using the user-defined method, compare the result with the String built-in toCharArray() method, and display the result

   Hint =>

   a. Take user input using the ***Scanner*** ***next()*** method to take the text into a String variable
   b. Write a method to return the characters in a string without using the ***toCharArray()***
   c. Write a method to compare two string arrays and return a boolean result
   d. In the main() call the user-defined method and the String built-in toCharArray() method, compare the 2 arrays, and finally display the result

4. Write a program to demonstrate NullPointerException.

   Hint =>

   a. Write a Method to generate the Exception. Here define the variable text and initialize it to null. Then call one of the String Method to generate the exception
   e. Write the Method to demonstrate ***NullPointerException***. Here define the variable text and initialize it to null. Then write try catch block for handling the Exception while accessing one of the ***String*** method
   b. From the main Firstly call the method to generate the Exception then refactor the code to call the method to handle the RuntimeException

5. Write a program to demonstrate ***StringIndexOutOfBoundsException***

   Hint =>

   a. Define a variable of type String and take user input to assign a value
   b. Write a Method to generate the Exception. Access the index using charAt() beyond the length of the String. This will generate a runtime exception and abruptly stop the program.
   c. Write the Method to demonstrate ***StringIndexOutOfBoundsException***. Access the index using ***charAt()*** beyond the length of the String. Then write try catch block for Exception while accessing the String method
   d. From the main Firstly call the method to generate the Exception then call the method to handle the RuntimeException

6. Write a program to demonstrate ***IllegalArgumentException***

   Hint =>

   a. Define a variable of type String and take user input to assign a value
   b. Write a Method to generate the Exception. Here use the ***subString()*** and set the start index to be greater than the end index. This will generate a runtime exception and abruptly stop the program.
   c. Write the Method to demonstrate ***IllegalArgumentException***. Here use the ***subString()*** and set the start index to be greater than the end index. This will generate a runtime exception. Use the try-catch block to handle the ***IllegalArgumentException*** and the generic runtime exception
   d. From the main Firstly call the method to generate the Exception then call the method to handle the RuntimeException

7. Write a program to demonstrate ***NumberFormatException***

   Hint =>

   a. Define a variable to take user input as a String
   b. Use Integer.parseInt() to generate this exception. ***Integer.parseInt()*** is a built-in function in java.lang.Integer class to extract the number from text. In case the text does not contain numbers the method will throw NumberFormatException which is a runtime exception
   c. Write a Method to generate the Exception. Use ***Integer.parseInt(text)*** to extract number from the text. This will generate a runtime exception and abruptly stop the program.
   d. Write the Method to demonstrate ***NumberFormatException***. Use ***Integer.parseInt(text)*** to extract number from the text. This will generate a runtime exception. Use the try-catch block to handle the ***NumberFormatException*** as well as the generic runtime exception
   e. From the main Firstly call the method to generate the Exception then call the method to handle the RuntimeException

8. Write a program to demonstrate ***ArrayIndexOutOfBoundsException***

   Hint =>

   a. Define a variable of array of names and take input from the user
   b. Write a Method to generate the Exception. Here access index larger then the length of the array. This will generate a runtime exception and abruptly stop the program.
   c. Write the Method to demonstrate ***ArrayIndexOutOfBoundsException***. Here access index larger then the length of the array. This will generate a runtime exception. Use the try-catch block to handle the ***ArrayIndexOutOfBoundsException*** and the generic runtime exception
   d. From the main Firstly call the method to generate the Exception then call the method to handle the RuntimeException

9. Write a program to convert the complete text to uppercase and compare the results

   Hint =>

   a. Take user input using the ***Scanner*** ***nextLine()*** method to take the complete text into a String variable
   b. Write a method using the String built-in ***charAt()*** method to convert each character if it is lowercase to the uppercase. Use the logic ASCII value of 'a' is 97 and 'A' is 65 so the difference is 32, similarly ASCII value of 'b' is 98 and 'B' is 66 so the difference is 32, and so on
   c. Write a method to compare two strings using the charAt() method and return a boolean result
   d. In the main() use the String built-in method ***toUpperCase()*** to get the uppercase text and compare the two strings using the user-defined method. And finally display the result

10. Write a program to convert the complete text to lowercase and compare the results

    Hint =>

    a. Take user input using the ***Scanner*** ***nextLine()*** method to take the complete text into a String variable
    b. Write a method using the String built-in ***charAt()*** method to convert each character if it is uppercase to the lowercase. Use the logic ASCII value of 'a' is 97 and 'A' is 65 so the difference is 32, similarly ASCII value of 'b' is 98 and 'B' is 66 so the difference is 32, and so on
    c. Write a method to compare two strings using the charAt() method and return a boolean result
    d. In the main() use the String built-in method ***toLowerCase()*** to get the lowercase text and compare the two strings using the user-defined method. And finally display the result