Java Developer – Fresher

Candidate-ID	**0012500043**
Candidate Name	**Kranthi Chekka**
Email	**kranthi.chekka@gsspec.com**
Mobile	**7013210545**
Source	**Naukari**

Screening Test

Completed on : 15-Feb-2025 02:28 PM

#	Question	Response	Marks
1	Have you taken any long term training course on Java Technologies	Yes	1
2	Can you write Java Programs on your own	Yes	1
3	Did you complete any kind of Developer Certification in Java	Yes	1
4	Would you be open to relocating to Hyderabad?	Yes	1
5	Have you fully completed your education and prepared to work	Yes	1
6	Will you be flexible any role in Software Engineering	Yes	1
7	Explain about yourself why you are the right candidate for this Job	video response	1
8	Have you recently completed your education and training, and there no is major Gap (more then one year after the education)	Yes	1
9	Are you, willing to work on-premises (there is no work from home)	Yes	1
10	Are you, available immediately to join us, if we hire you	Yes	1
11	Do you know how to write unit testcases using Junit	Yes	1
12	if we hire you, will you be prepared to stay and work with us at least for two years	Yes	1

Overall Score : 12/13 (Promoted)

Coding Test

Completed on : 15-Feb-2025 03:37 PM

1. Question: Implement an EMI Calculator in Java. The objective is to create a program that calculates the Equated Monthly Installment (EMI) for a given loan. The EMI is calculated based on the principal loan amount, annual rate of interest, and loan tenure. Sample Input : Principal amount: 500000 Annual rate of interest: 7.5 Loan tenure: 20 Expected Output : The calculated EMI is: 3982.71
Marks Scored : 2

```
public class EMI_Calculator {

  // Method to calculate EMI
   public double calculateEMI(double principal, double annualRateOfInterest, int loanTenure) {
      if (principal < 0) {
         throw new IllegalArgumentException("Principal amount cannot be negative");
      }
      if (annualRateOfInterest < 0) {
         throw new IllegalArgumentException("Interest rate cannot be negative");
      }
      if (loanTenure <= 0) {
         throw new ArithmeticException("Loan tenure cannot be zero or negative");
      }

      //implement Your code here
      double rateOfInterest = annualRateOfInterest / 12 / 100;
      int numberOfInstallments = loanTenure * 12;

      double emi = (principal * rateOfInterest * Math.pow(1+rateOfInterest, numberOfInstallments)) /
                  (Math.pow(1 + rateOfInterest, numberOfInstallments) - 1);
      return emi;
  }

  public static void main(String[] args) {
      EMI_Calculator calculator = new EMI_Calculator();

      // Test data
      double principal = 500000;
      double annualRateOfInterest = 10;
      int loanTenure = 20;

      // Calculate EMI
      try {
         double emi = calculator.calculateEMI(principal, annualRateOfInterest, loanTenure);
         System.out.printf("The EMI for a principal of %.2f, interest rate of %.2f%%, and tenure of %d years is: %.2f%n",
               principal, annualRateOfInterest, loanTenure, emi);
      } catch (Exception e) {
         System.err.println("Error: " + e.getMessage());
      }
  }
```

```
}
```

2. Write a Java program that counts the occurrences of words in a given input string that are of a specified minimum length or longer. The program should return a map of words sorted in descending order of their frequency. If two words have the same frequency, they should be sorted alphabetically in ascending order. Sample Input : Ignore case sensitivity (e.g., "Test" and "test" should be treated as the same word). Words are sequences of alphanumeric characters. Punctuation and other symbols should not be considered part of words. Words shorter than the specified minimum length should be excluded from the count. Expected Output : Words with minimum length 8: exceptional: 2 wonderful: 1
Marks Scored : 4

```
import java.util.*;
import java.util.stream.Collectors;

public class WordCounter {

  public static Map getWordsWithMinLength(String input, int minLength) {
    if (input == null || input.isEmpty()) {
      return Collections.emptyMap();
    }

    input = input.toLowerCase();
    String[] words = input.split("\\W+");
    Map wordCount = new HashMap<>();

    for (String word : words) {
      if (word.length() >= minLength) {
        wordCount.put(word, wordCount.getOrDefault(word, 0) + 1);
      }
    }

    return wordCount.entrySet()
        .stream()
        .sorted((a, b) -> b.getValue().equals(a.getValue())
            ? a.getKey().compareTo(b.getKey())
            : b.getValue() - a.getValue())
        .collect(Collectors.toMap(
            Map.Entry::getKey,
            Map.Entry::getValue,
            (e1, e2) -> e1,
            LinkedHashMap::new
        ));
  }

  public static void main(String[] args) {
    String input = "this is exceptional";
    int minLength = 8;
```

```
        getWordsWithMinLength(input, minLength)
            .forEach((word, count) -> System.out.println(word + ": " + count));
    }
}
```

3. Question Write a program to demonstrate most simple usage, application of design pattern prototype. Expected Output: Original: Shape[type=Circle, color=Red] Cloned: Shape[type=Circle, color=Red] After modifying cloned object: Original: Shape[type=Circle, color=Red] Cloned: Shape[type=Circle, color=Blue]
Marks Scored : 0

```
// PrototypeDemo.java
class Shape implements Cloneable {
    private String type;
    private int[] dimensions;

    public Shape(String type, int[] dimensions) {
        // CODE HERE
    }

    public String getType() {
        // CODE HERE
    }

    public int[] getDimensions() {
        // CODE HERE
    }

    @Override
    protected Shape clone() {
        try {
                    // CODE HERE
            return clone;
        } catch (CloneNotSupportedException e) {
            return null;
        }
    }
}

public class PrototypeDemo {
    public static void main(String[] args) {
        Shape original = new Shape("Circle", new int[]{7});
        Shape cloned = original.clone();
        cloned.getDimensions()[0] =14;
        System.out.println("Original Shape: " + original.getType() + " with dimensions " + java.util.Arrays.toString(original.getDimensions()));
        System.out.println("Cloned Shape: " + cloned.getType() + " with dimensions " + java.util.Arrays.toString(cloned.getDimensions()));
    }
}
```

Overall Score : 6/12 (Promoted)