Java Full Stack Developer

Candidate-ID	**0012500069**
Candidate Name	**Sushama Walunjkar**
Email	**sushamawalunjkar78@gmail.com**
Mobile	
Source	**Careers**

Screening Test

Completed on : 14-Nov-2025 08:29 PM

#	Question	Response	Marks
1*	Do you have professional experience between 1 to 5 years?	Yes	1
2*	Would you be open to relocating to Hyderabad?	Yes	1
3	Are you familiar with Agile methodologies, including Scrum or Kanban?	Yes	1
4	Are you open to traveling for work when necessary, either domestically or internationally?	Yes	1
5	Will you be flexible any role in Software Engineering	Yes	1
6	Explain about yourself why you are the right candidate for this Job	video response	1
7	Have you recently completed your education and training, and there no is major Gap (more then one year after the education)	Yes	1
8	Are you, willing to work on-premises (there is no work from home)	Yes	1
9	Are you, available immediately to join us, if we hire you	Yes	1
10	if we hire you, will you be prepared to stay and work with us at least for two years	Yes	1

Overall Score : 10/10 (Promoted)

Coding Test

Completed on : 15-Nov-2025 01:07 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

```
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 monthlyRate = (annualRateOfInterest/12.0)/100.0;
        int numberOfMonths = loanTenure*12;
        if (monthlyRate==0.0){
            return principal/numberOfMonths;
        }
        double pow =Math.pow(1+monthlyRate,numberOfMonths);
        double emi=(principal*monthlyRate*pow)/(pow-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());
        }
    }
}
```

Marks Scored : 2/2

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

```
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();
      }

      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)->{
         int freqCompare = b.getValue().compareTo(a.getValue());
         if(freqCompare==0){
            return a.getKey().compareTo(b.getKey());
         }
         return freqCompare;
      }).collect(Collectors.toMap(Map.Entry::getKey,Map.Entry::getValue,(e1,e2)->e1,LinkedHashMap::new));


   }
   public static void main(String[] args){
      String input = "java is grate and java is powerfull.java code is fun!";
      int minLength = 3;

      Map result = getWordsWithMinLength(input,minLength);
      System.out.println(result);
   }
}
```

Marks Scored : 4/4

3. Question: Write Java Program to use and apply the concept of Design Pattern Singleton with simplest program. Expected Output: Instance 1: Singleton@15db9742 Instance 2: Singleton@15db9742 Both instances are the same (Singleton verified). Message from Singleton: Hello from Singleton!

```
public class Singleton {

    // Private static instance of the class
    private static Singleton instance;

    // Private constructor to prevent instantiation
    private Singleton() {}

    // Public method to provide access to the instance
    public static Singleton getInstance() {
        // Implement your code here
        if(instance==null){
            instance=new Singleton();
        }
        return instance;
    }

    // Example method to demonstrate functionality
    public String getMessage() {
        // Implement your code here
        return "Hello from Singleton!";
    }

    // Main method to test the Singleton functionality
    public static void main(String[] args) {
        // Implement your code here

        Singleton s1=Singleton.getInstance();
        System.out.println("instance 1 : "+s1);

        Singleton s2= Singleton.getInstance();
        System.out.println("instance 2 : "+s2);

        // Implement your code here
        System.out.println("Are both instances the same? ");
        System.out.println(s1.getMessage());
    }
}
```

Marks Scored : 6/6

Overall Score : 12/12 (Promoted)