JAVA PRACTICE QUESTIONS

1.Create one superclass HillStations and three subclasses Manali, Mussoorie, Gulmarg. Subclasses extend the superclass and override its location() and famousFor() method. i.call the location() and famousFor() method by the Parent class', i.e. Hillstations class. As it refers to the base class object and the base class method overrides the superclass method; the base class method is invoked at runtime. ii.call the location() and famousFor() method by the all subclass',and print accordingly.

Sol:

```
class HillStations
{
   void location()
{
      System.out.println("Location: Generic Hill Station");
   }

   void famousFor()
{
      System.out.println("Famous for: Natural beauty and pleasant weather");
   }
}

class Manali extends HillStations
{

   void location()
 {
      System.out.println("Location: Manali, Himachal Pradesh");
   }

   void famousFor()
{
      System.out.println("Famous for: Adventure sports and scenic landscapes");
   }
}

class Mussoorie extends HillStations
{
   void location()
{
```

```
        System.out.println("Location: Mussoorie, Uttarakhand");
    }
    void famousFor()
 {
        System.out.println("Famous for: Hilltop views and colonial architecture");
    }
 }

 class Gulmarg extends HillStations
 {
    void location()
 {
        System.out.println("Location: Gulmarg, Jammu and Kashmir");
    }
void famousFor()
{
        System.out.println("Famous for: Skiing and snow-covered landscapes");
    }
 }

 public class HillStationDemo
 {
    public static void main(String[] args)
 {
        HillStations genericHillStation = new HillStations();
        genericHillStation.location();
        genericHillStation.famousFor();
        Manali manali = new Manali();
        Mussoorie mussoorie = new Mussoorie();
        Gulmarg gulmarg = new Gulmarg();

        manali.location();
        manali.famousFor();

        mussoorie.location();
        mussoorie.famousFor();

        gulmarg.location();
        gulmarg.famousFor();
    }
 }
```

2. Write a Java program that demonstrates method overriding by creating a superclass called Animal and two subclasses called Dog and Cat. ● The Animal class should have a method called makeSound(), which simply prints "The animal makes a sound." ● The Dog and Cat classes should override this method to print "TheCat/The dog meows/barks" respectively. ● The program should allow the user to create and display objects of each class.

Sol:

```
class Animal {

   void makeSound()

 {

      System.out.println("The animal makes a sound.");

   }

}



class Dog extends Animal {

   void makeSound()

{

      System.out.println("The dog barks.");

   }

}



class Cat extends Animal {

   void makeSound()

 {

      System.out.println("The cat meows.");
```

```
    }
}


public class AnimalDemo {
    public static void main(String[] args)
{
        Animal genericAnimal = new Animal();
        Dog myDog = new Dog();
        Cat myCat = new Cat();
        genericAnimal.makeSound();
        myDog.makeSound();
        myCat.makeSound();
    }
}
```

3. Write code to determine if the string is a palindrome.

Input String: Madam

Output: Madam is a Palindrome

Sol:

```
public class PalindromeChecker {
    public static void main(String[] args) {
        String inputString = "Madam";
        boolean isPalindrome = checkPalindrome(inputString);
```

```
        System.out.println("Input String: " + inputString);

        System.out.println(isPalindrome ? "Output: " + inputString + " is a Palindrome" : "Output: " +
inputString + " is not a Palindrome");

    }



    public static boolean checkPalindrome(String input) {

        String cleanInput = input.replaceAll("[^a-zA-Z]", "").toLowerCase();

        int length = cleanInput.length();

        for (int i = 0; i < length / 2; i++) {

            if (cleanInput.charAt(i) != cleanInput.charAt(length - 1 - i)) {

                return false;

            }

        }

        return true;

    }

}
```

4. You need to find and print all the unique characters in a given string.

Input string: java

Output: jv

Sol:

```
public class UniqueCharacters {

    public static void main(String[] args) {

        String inputString = "java";
```

```
        String uniqueCharacters = findUniqueCharacters(inputString);
        System.out.println("Input string: " + inputString);
        System.out.println("Output: " + uniqueCharacters);
    }

    public static String findUniqueCharacters(String input) {
        StringBuilder result = new StringBuilder();
        for (int i = 0; i < input.length(); i++) {
            char currentChar = input.charAt(i);
            if (result.indexOf(String.valueOf(currentChar)) == -1) {
                result.append(currentChar);
            }
        }
        return result.toString();
    }
}
```