Q1. Write a java program to demonstrate the scope of private access modifier.

class Person {

    private String name = "Srushti"; *// Private variable*

    private void displayName() {   *// Private method*

        System.out.println("Name: " + name);

    }

    public void show() {

        System.out.println("Accessing private member from within the class:");

        displayName();

    }

}

public class Private {

    public static void main(String[] args) {

        Person p = new Person();

*// p.name = "* Srushti*";         // !Error: 'name' has private access*

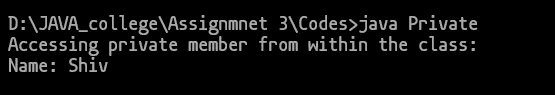
*// p.displayName();        // !Error: 'displayName()' has private access*

        p.show(); *// ?Allowed: public method accessing private members internally*

    }

}

OUTPUT:



Q2. Write a java program to demonstrate the scope of protected access modifier.

class Animal {

    protected String type = "Mammal"; *// Protected variable*

    protected void sound() {          *// Protected method*

        System.out.println("Animal makes a sound");

    }

}

class Dog extends Animal {

    public void showDetails() {

        System.out.println("Accessing protected member from subclass:");

        System.out.println("Type: " + type);

        sound();

    }

}

public class Protected {

    public static void main(String[] args) {

        Dog d = new Dog();

        d.showDetails(); *// Accessing via subclass method*

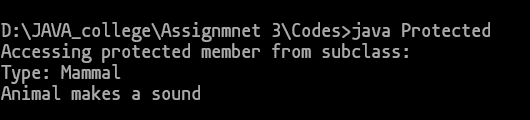
*// d.type = "Bird";  // !Not allowed outside subclass directly (though same package allows it)*

*// d.sound();        // !Not recommended: should access via method*

    }

}

OUTPUT:



Q3. Write a java program to demonstrate the scope of default access modifier.

*// File: Default1.java*

public class Default1 {

    String text = "Hello, this is default access"; *// Default variable*

    void display() { *// Default method*

        System.out.println(text);

    }}

*// File: Default2.java (same package as Default1.java)*

public class Default2 {

    public static void main(String[] args) {

        Default1  msg = new Default1();

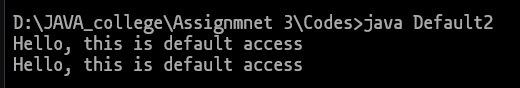
        msg.display();           *// ?Allowed: same package*

        System.out.println(msg.text); *// ?Allowed: same package*

    }

}

OUTPUT:



Q4. Write a java program to demonstrate the scope of public access modifier.

package test;

public class Public1{

    public String message = "Hello from a public class!"; *// Public variable*

    public void sayHello() { *// Public method*

        System.out.println(message);

    }

}

import test.\*;

public class Public2 {

    public static void main(String[] args) {

        Public1 g = new Public1();    *// ?Accessible anywhere*

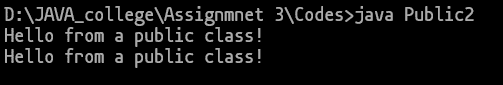
        g.sayHello();                   *// ?Public method accessible*

        System.out.println(g.message);  *// ?Public variable accessible*

    }

}

OUTPUT:



Q5. Write a java program such that if we create object of class it will print “SGSITS”.

class temp{

    temp(){

            System.out.println("SGSITS");

        }

}

public class Sgsits {

    public static void main(String[] args) {

        temp obj = new temp();

    }

}

OUTPUT:



Q6. Write a java program where we create object it takes 3 parameters i.e. name , age , city and print it.

class temp{

    temp(String name, int age, String city){

        System.out.println("Name: " + name);

        System.out.println("Age: " + age);

        System.out.println("City: " + city);

    }

}

public class Constructor {

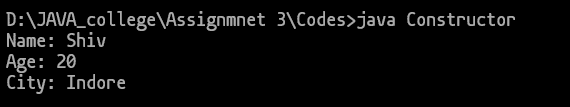
    public static void main(String[] args) {

        temp obj = new temp("Srushti ", 20, "Indore");

    }

}

OUTPUT:



Q7. Write a java program for such that it will be unalterable i.e can not change the value of  variable , methods should not be overridden and class will not inherited.

final class Unalterable {

    private final String data = "This value cannot change"; *// Final variable*

    public final void display() { *// Final method*

        System.out.println(data);

    }

}

public class Final {

    public static void main(String[] args) {

        Unalterable obj = new Unalterable();

        obj.display();

    }

}

OUTPUT:



Q8. Write a java program to create a static method and static variables in class A and call it.

class A{

    static int count = 5; *// Static variable*

    static void display() { *// Static method*

        System.out.println("Static count value: " + count);

    }

}

public class Static {

    public static void main(String[] args) {

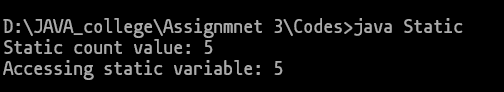
        A.display();

        System.out.println("Accessing static variable: " + A.count);

    }

}

OUTPUT:



Q9. Write a java program to make static method in class A and call it class B.

class A {

    static void greet() {

        System.out.println("Class A says Hello!");

    }

}

public class B {

    public static void main(String[] args) {

        A.greet();

    }

}

OUTPUT:



Q10. WAP to merge two arrays.

import java.util.Arrays;

public class MergeArrays {

    public static void main(String[] args) {

        int[] arr1 = {1, 3, 5};

        int[] arr2 = {2, 4, 6};

        int[] merged = new int[arr1.length + arr2.length];

        for (int i = 0; i < arr1.length; i++) {

            merged[i] = arr1[i];

        }

        for (int i = 0; i < arr2.length; i++) {

            merged[arr1.length + i] = arr2[i];

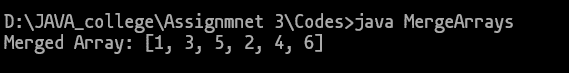
        }

        System.out.println("Merged Array: " + Arrays.toString(merged));

    }

}

OUTPUT:



Q11. WAP to sort an array.

import java.util.Arrays;

public class SortArray {

    public static void main(String[] args) {

        int[] arr = {5, 2, 8, 1, 3};

        System.out.println("Original Array: " + Arrays.toString(arr));

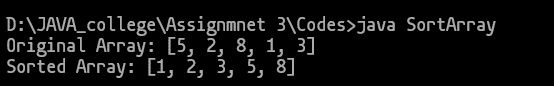
        Arrays.sort(arr);

        System.out.println("Sorted Array: " + Arrays.toString(arr));

    }

}

OUTPUT:



Q12. WAP to search an element in given array.

import java.util.Scanner;

public class SearchElement {

    public static void main(String[] args) {

        int[] arr = {10, 25, 30, 45, 50, 12, 34, 13, 90, 100, 200, 323, 32, 34, 45, 56, 78, 90, 109};

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter element to search: ");

        int key = sc.nextInt();

        boolean found = false;

        for (int i = 0; i < arr.length; i++) {

            if (arr[i] == key) {

                System.out.println(key + " found at index " + i);

                found = true;

                break;

            }

        }

        if (!found) {

            System.out.println(key + " not found in the array.");

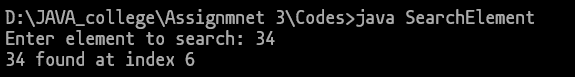
        }

        sc.close();

    }

}

OUTPUT:



Q13. WAP to find median of two sorted array.

import java.util.Arrays;

public class Median{

    public static void main(String[] args) {

        int[] arr1 = {1, 3, 5, 7, 1, 13, 15};

        int[] arr2 = {2, 4, 6, 8, 10, 12, 14};

        int n1 = arr1.length;

        int n2 = arr2.length;

        int[] merged = new int[n1 + n2];

        System.arraycopy(arr1, 0, merged, 0, n1);

        System.arraycopy(arr2, 0, merged, n1, n2);

        Arrays.sort(merged);

        double median;

        int n = merged.length;

        if (n % 2 == 0) {

            median = (merged[n/2 - 1] + merged[n/2]) / 2.0;

        } else {

            median = merged[n/2];

        }

        System.out.println("Merged Array: " + Arrays.toString(merged));

        System.out.println("Median: " + median);

    }

}

OUTPUT:

