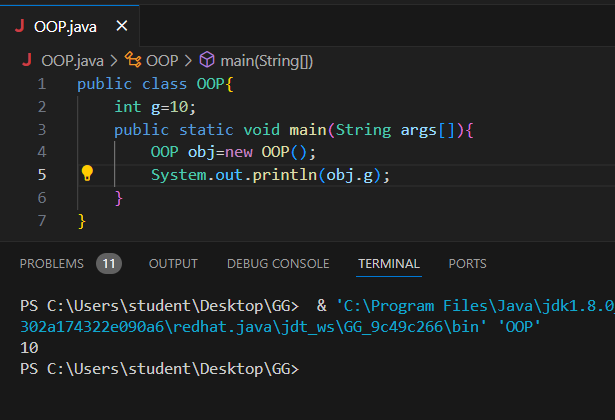
**OOP IN JAVA**

**1.**

public class OOP{

    int g=10;

    public static void main(String args[]){

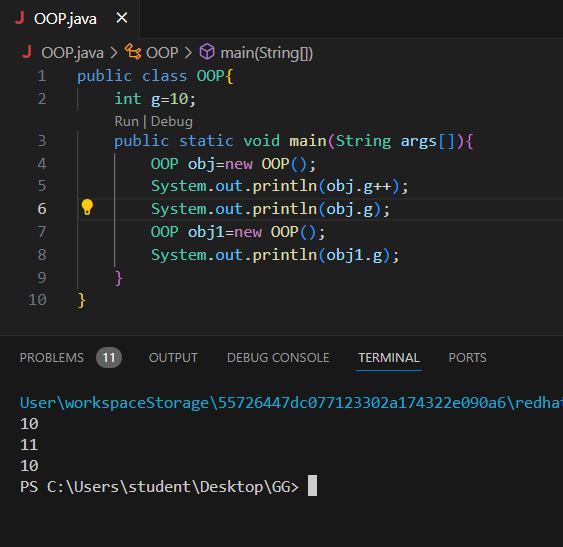
        OOP obj=new OOP();

        System.out.println(obj.g);

    }

}

2.



public class OOP{

    int g=10;

    public static void main(String args[]){

        OOP obj=new OOP();

        System.out.println(obj.g++);

        System.out.println(obj.g);

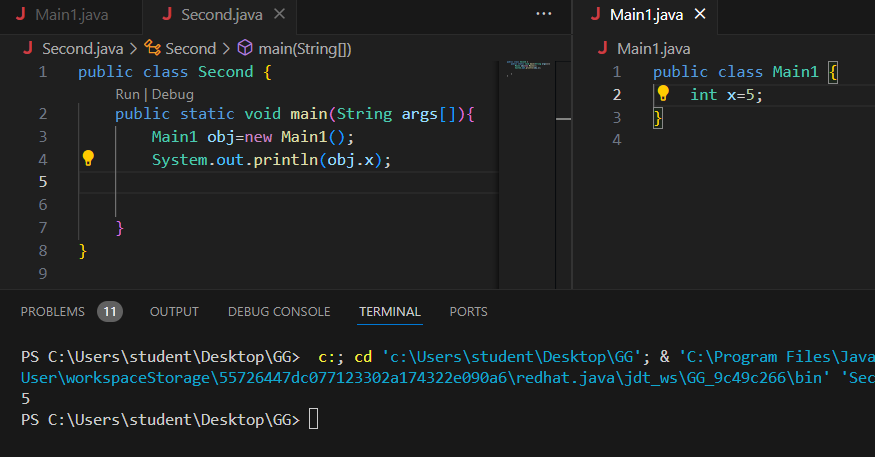
        OOP obj1=new OOP();

        System.out.println(obj1.g);

    }

}

3.



Main1.java

public class Main1 {

int x=5;

}

Second.java

public class Second {

public static void main(String args[]){

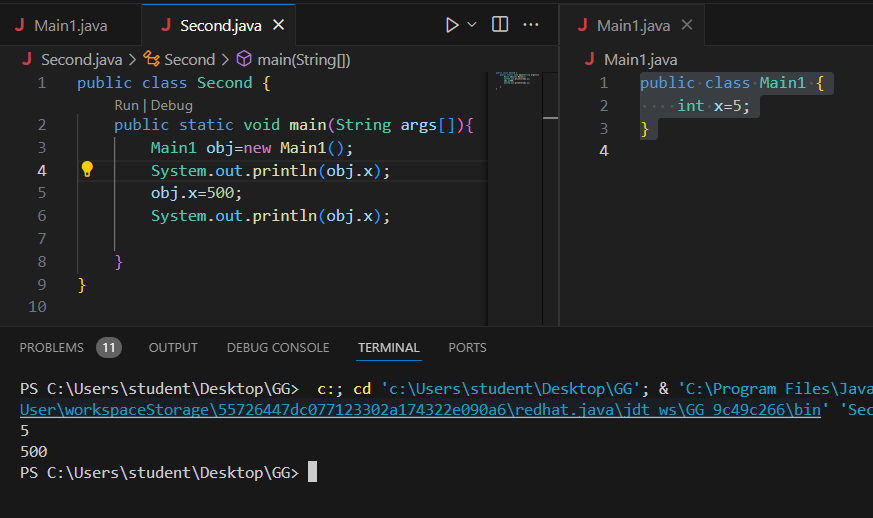
Main1 obj=new Main1();

System.out.println(obj.x);

}

}

4.reassigning



public class Main1 {

int x=5;

}

public class Second {

public static void main(String args[]){

Main1 obj=new Main1();

System.out.println(obj.x);

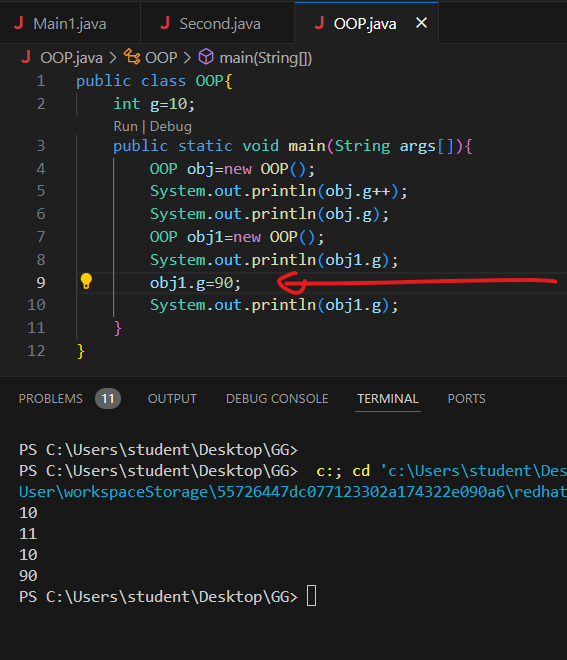
obj.x=500;

System.out.println(obj.x);

}

}

5.overridding



public class OOP{

    final int g=10;

    public static void main(String args[]){

        OOP obj=new OOP();

        System.out.println(obj.g++);

        System.out.println(obj.g);

        OOP obj1=new OOP();

        System.out.println(obj1.g);

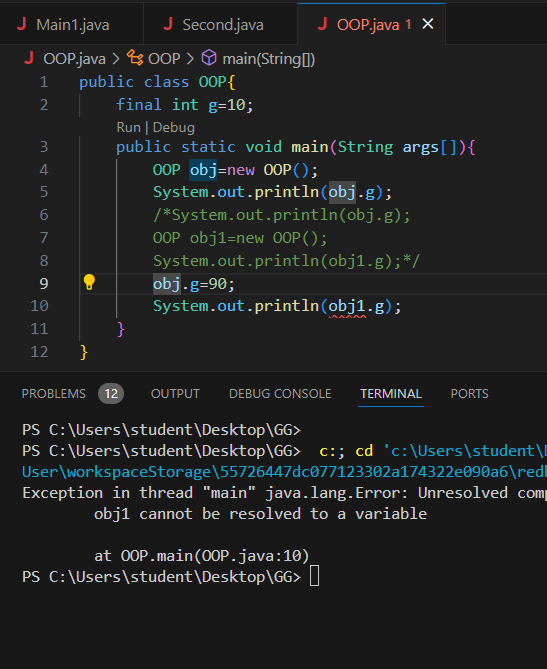
        obj1.g=90;

        System.out.println(obj1.g);

    }

}

6.to avoid overrididng (use final keyword)



public class OOP{

    final int g=10;

    public static void main(String args[]){

        OOP obj=new OOP();

        System.out.println(obj.g);

        /\*System.out.println(obj.g);

        OOP obj1=new OOP();

        System.out.println(obj1.g);\*/

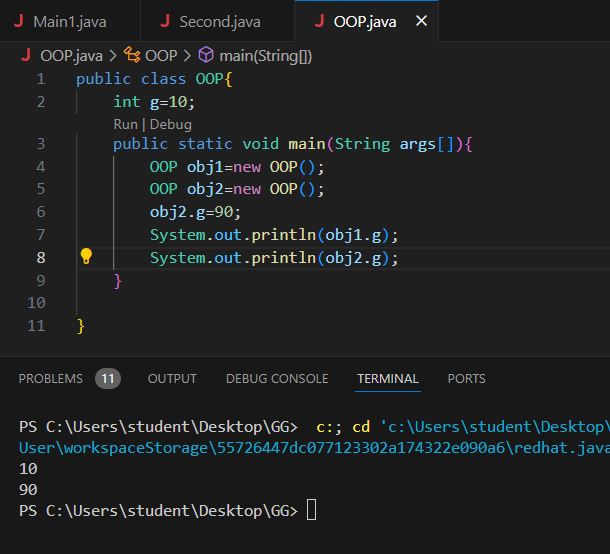
        obj.g=90;

        System.out.println(obj1.g);

    }

}

7. multiple obj with one attribute and make changes in attribute



public class OOP{

    int g=10;

    public static void main(String args[]){

        OOP obj1=new OOP();

        OOP obj2=new OOP();

        obj2.g=90;

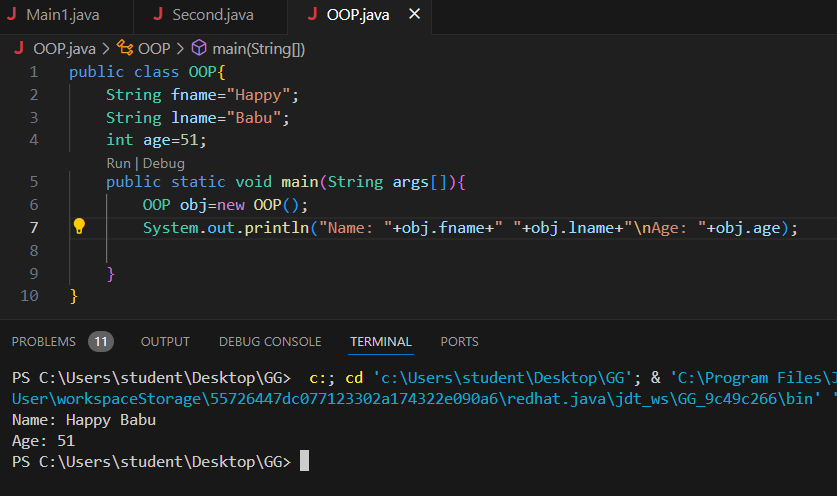
        System.out.println(obj1.g);

        System.out.println(obj2.g);

    }

}

8.



public class OOP{

    String fname="Happy";

    String lname="Babu";

    int age=51;

    public static void main(String args[]){

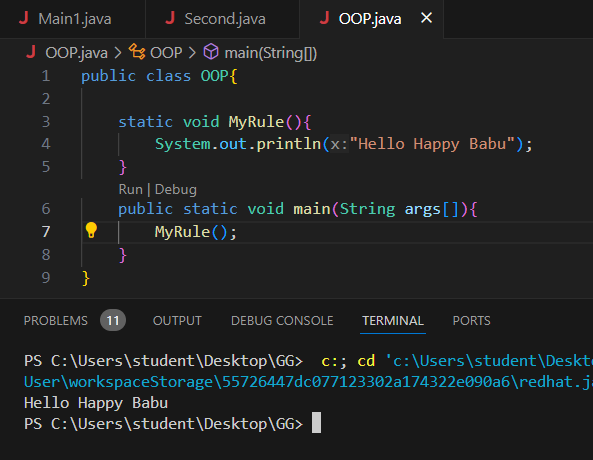
        OOP obj=new OOP();

        System.out.println("Name: "+obj.fname+" "+obj.lname+"\nAge: "+obj.age);

    }

}

9.



public class OOP{

    static void MyRule(){

        System.out.println("Hello Happy Babu");

    }

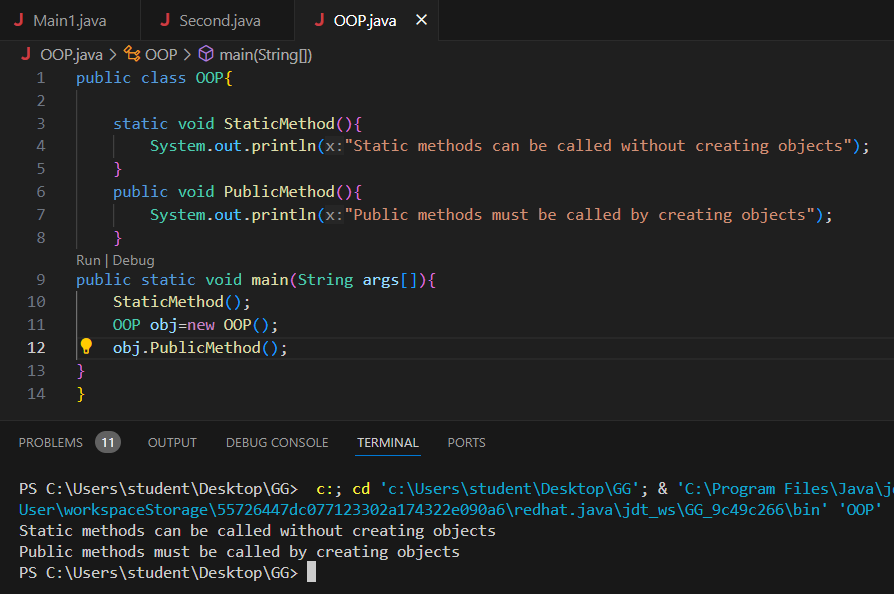
    public static void main(String args[]){

        MyRule();

    }

}

10.



public class OOP{

    static void StaticMethod(){

        System.out.println("Static methods can be called without creating objects");

    }

    public void PublicMethod(){

        System.out.println("Public methods must be called by creating objects");

    }

public static void main(String args[]){

    StaticMethod();

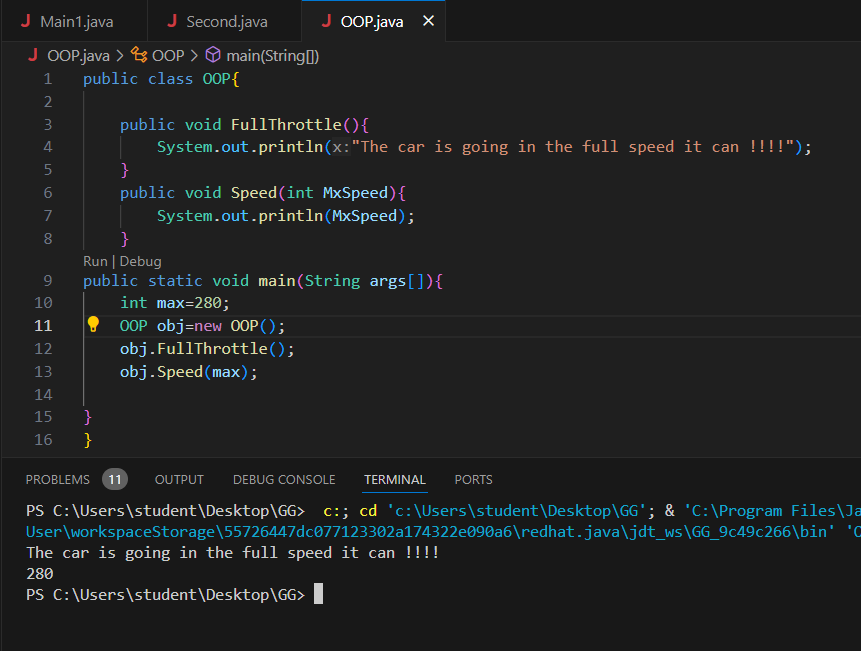
    OOP obj=new OOP();

    obj.PublicMethod();

}

}

11.



public class OOP{

    public void FullThrottle(){

        System.out.println("The car is going in the full speed it can !!!!");

    }

    public void Speed(int MxSpeed){

        System.out.println(MxSpeed);

    }

public static void main(String args[]){

    int max=280;

    OOP obj=new OOP();

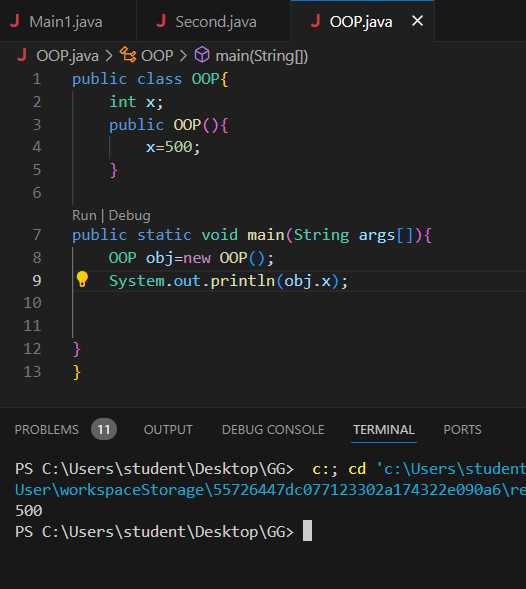
    obj.FullThrottle();

    obj.Speed(max);

}

}

12.



public class OOP{

    int x;

    public OOP(){

        x=500;

    }

public static void main(String args[]){

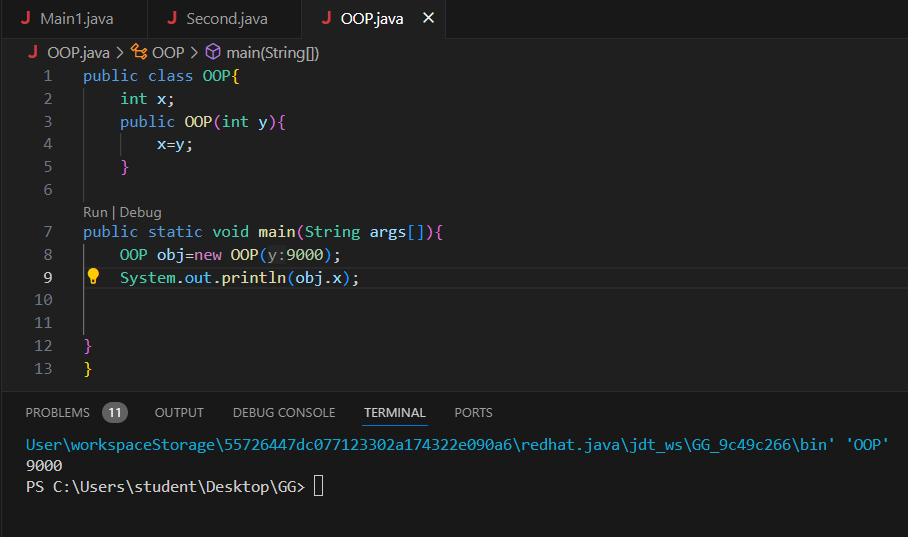
    OOP obj=new OOP();

    System.out.println(obj.x);

}

}

13.



public class OOP{

    int x;

    public OOP(int y){

        x=y;

    }

public static void main(String args[]){

    OOP obj=new OOP(9000);

    System.out.println(obj.x);

}

}