**Assignment**

1.

class bobo{

int id;

String name;

}

class ats{

public static void main(String args[]){

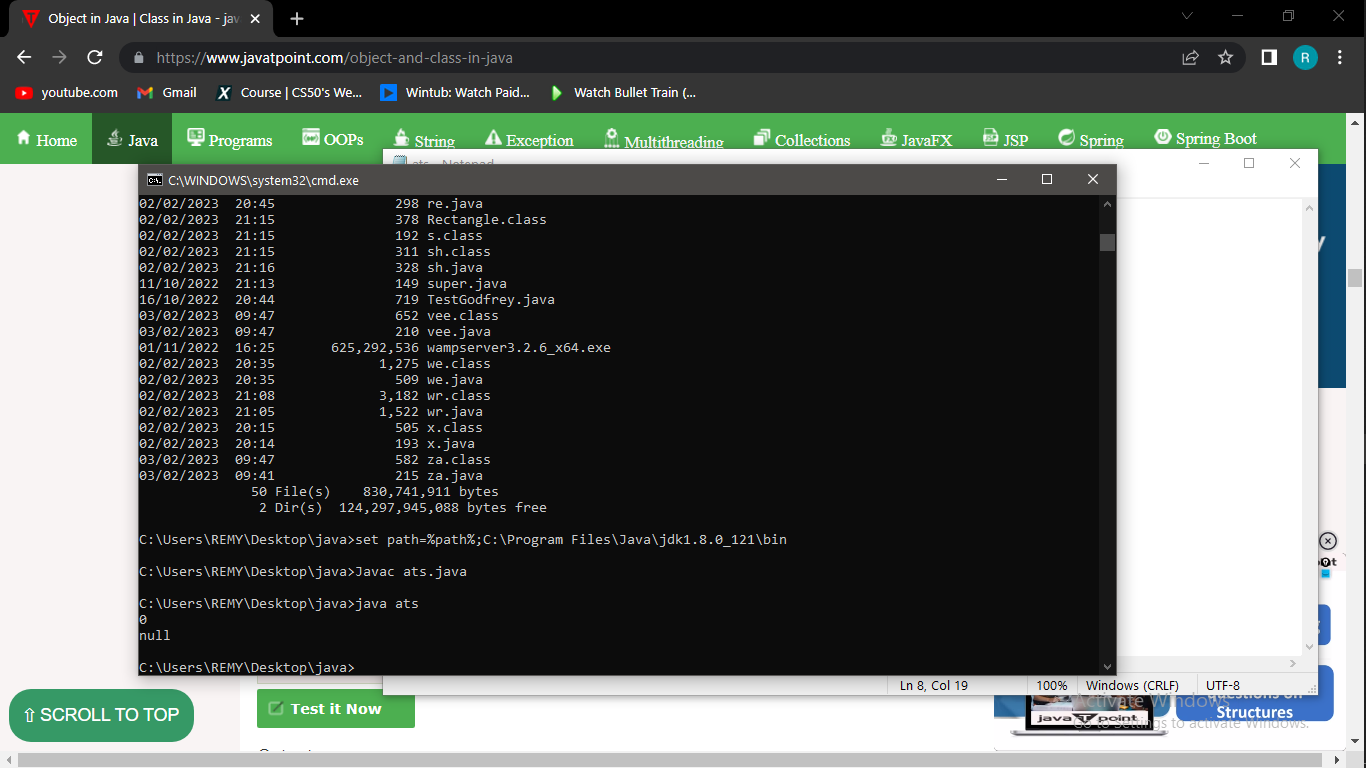
bobo s1=new bobo();

System.out.println(s1.id);

System.out.println(s1.name);

}

}



2

public class linka

{

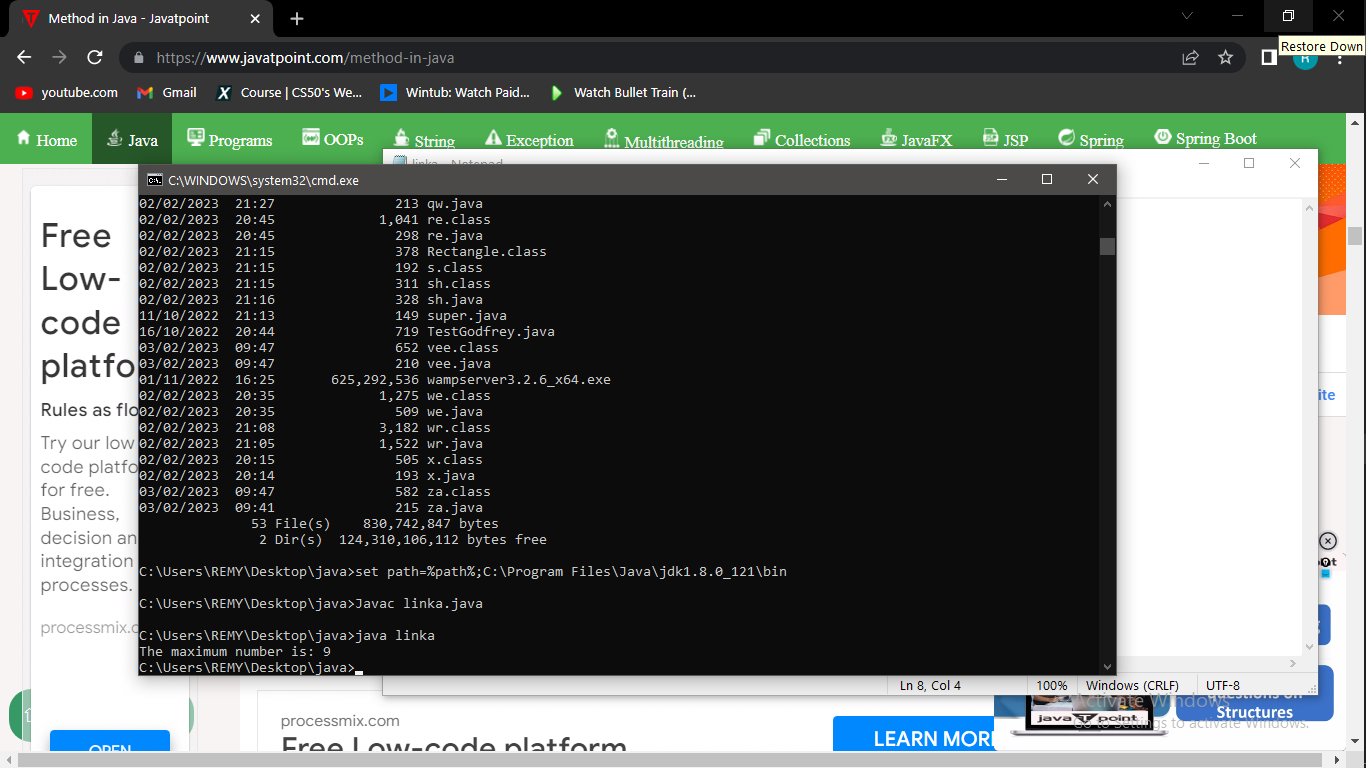
public static void main(String[] args)

{

System.out.print("The maximum number is: " + Math.max(9,7));

}

}



3.

public class Display

{

public static void main(String[] args)

{

show();

}

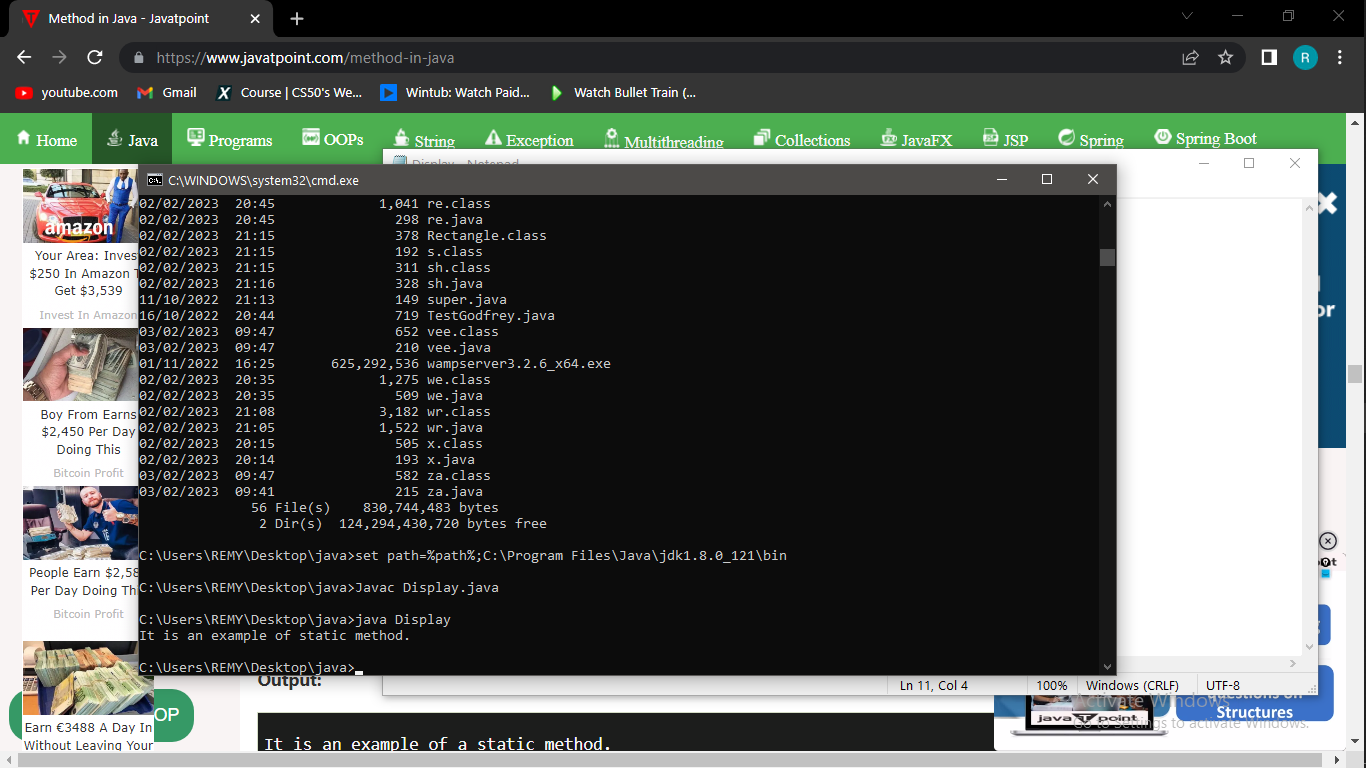
static void show()

{

System.out.println("It is an example of static method.");

}

}



4.

class Employee{

float salary=40000;

}

class Programmer extends Employee{

int bonus=10000;

public static void main(String args[]){

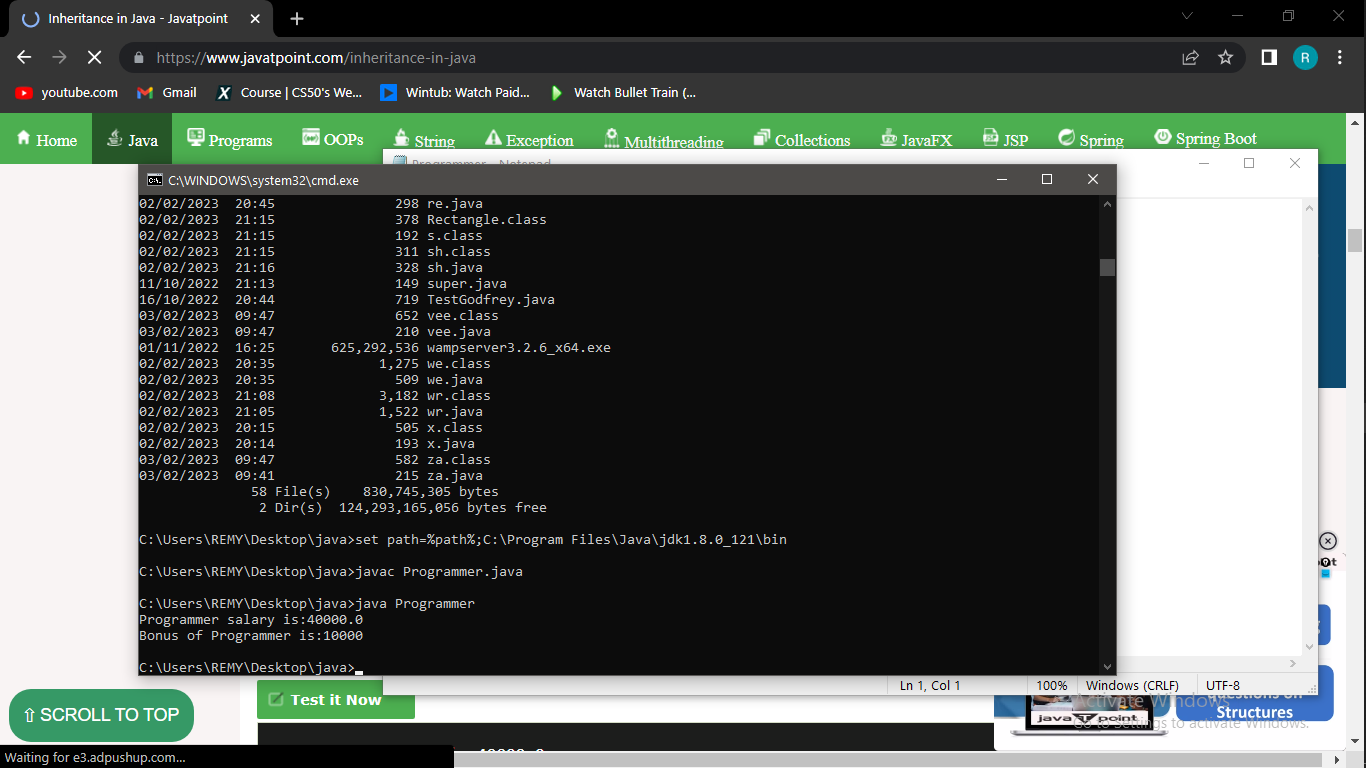
Programmer p=new Programmer();

System.out.println("Programmer salary is:"+p.salary);

System.out.println("Bonus of Programmer is:"+p.bonus);

}

}



5.

class Animal{

void eat(){System.out.println("eating...");}

}

class Dog extends Animal{

void bark(){System.out.println("barking...");}

}

class TestInheritance{

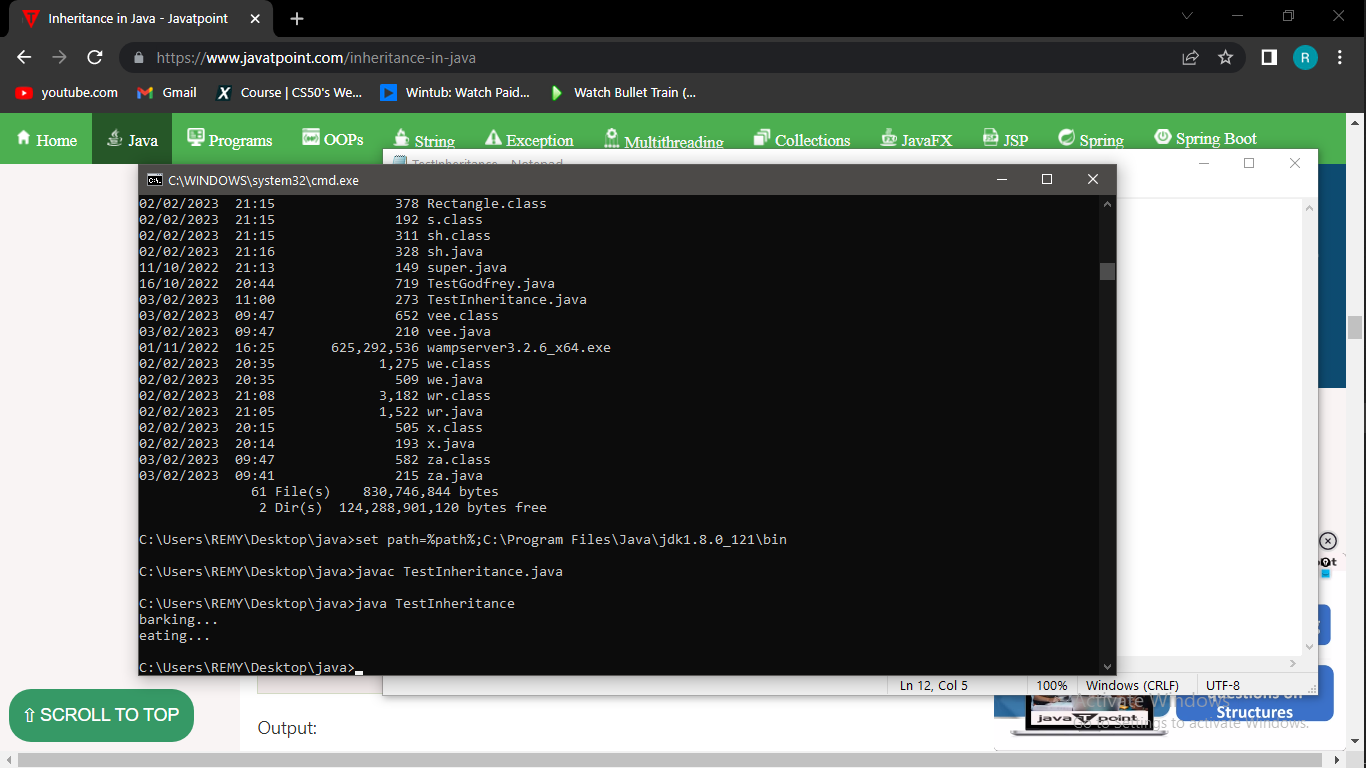
public static void main(String args[]){

Dog d=new Dog();

d.bark();

d.eat();

}}



6.

class Animal{

void eat(){System.out.println("eating...");}

}

class Dog extends Animal{

void bark(){System.out.println("barking...");}

}

class BabyDog extends Dog{

void weep(){System.out.println("weeping...");}

}

class TestInheritance2{

public static void main(String args[]){

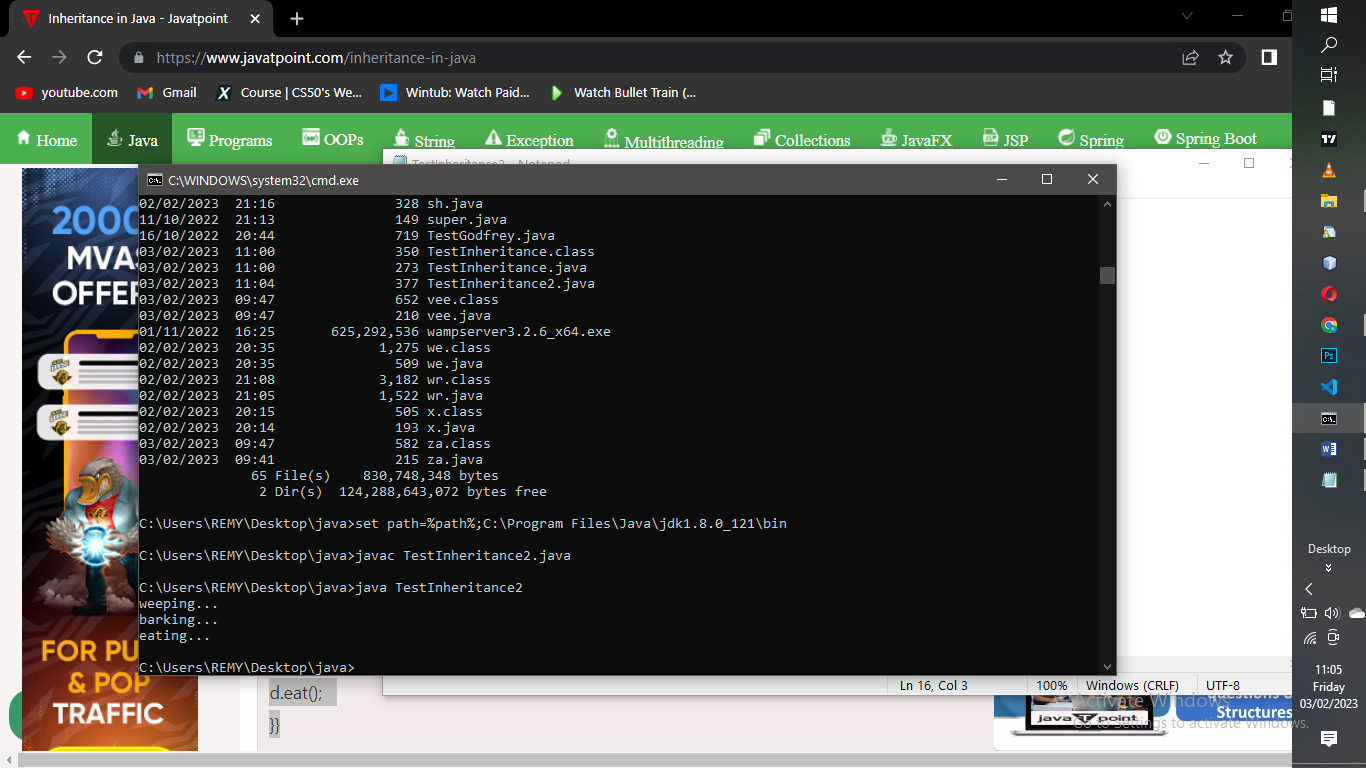
BabyDog d=new BabyDog();

d.weep();

d.bark();

d.eat();

}}



7.

class Adder{

static int add(int a,int b){return a+b;}

static int add(int a,int b,int c){return a+b+c;}

}

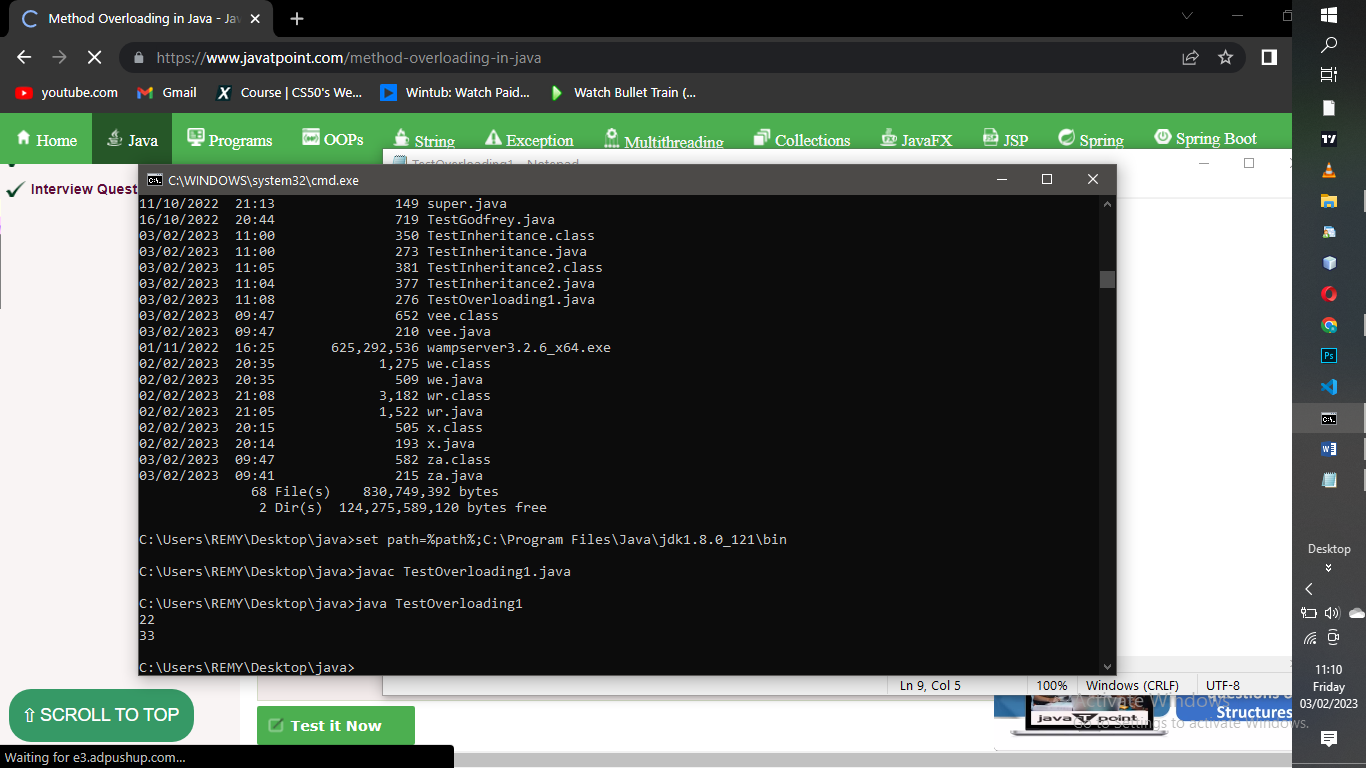
class TestOverloading1{

public static void main(String[] args){

System.out.println(Adder.add(11,11));

System.out.println(Adder.add(11,11,11));

}}



8.

class Adder{

static int add(int a, int b){return a+b;}

static double add(double a, double b){return a+b;}

}

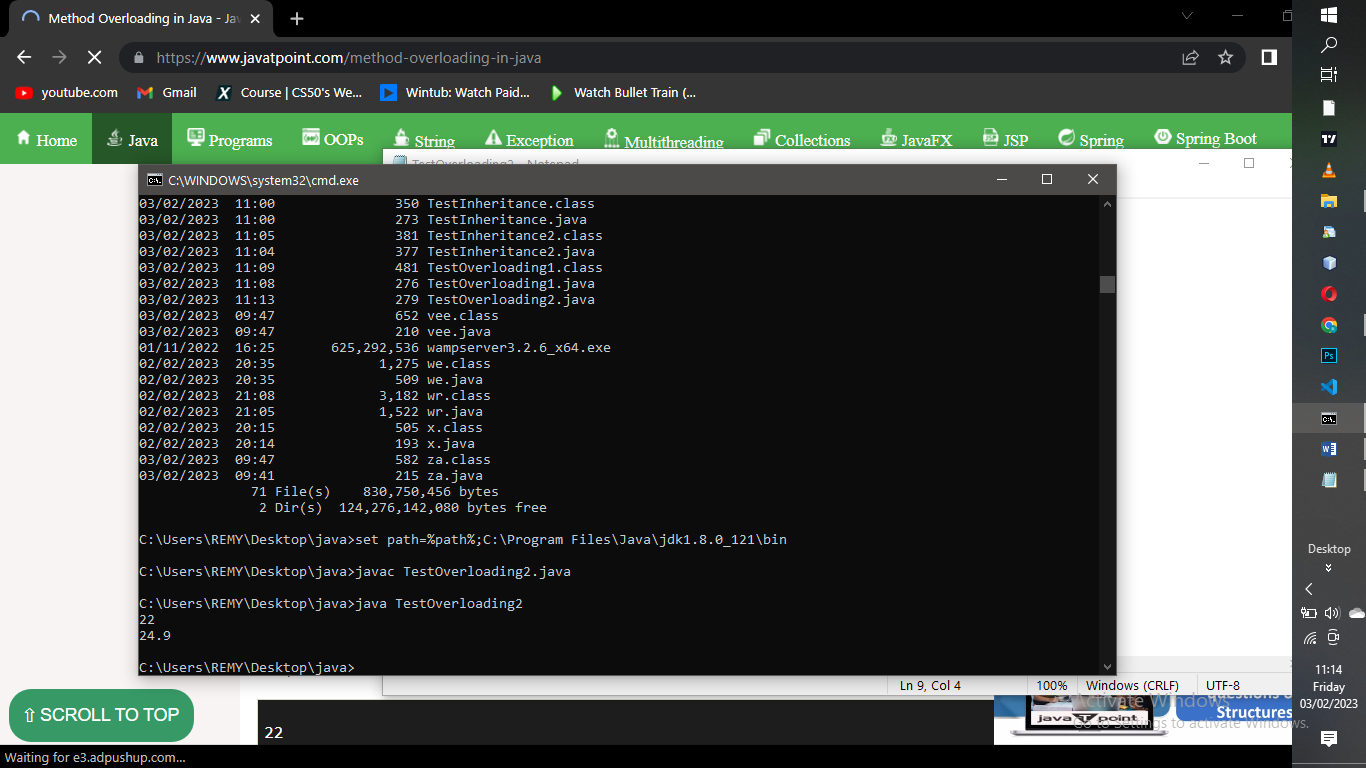
class TestOverloading2{

public static void main(String[] args){

System.out.println(Adder.add(11,11));

System.out.println(Adder.add(12.3,12.6));

}}



9.

class Adder{

static int add(int a,int b){return a+b;}

static double adder(int a,int b){return a+b;}

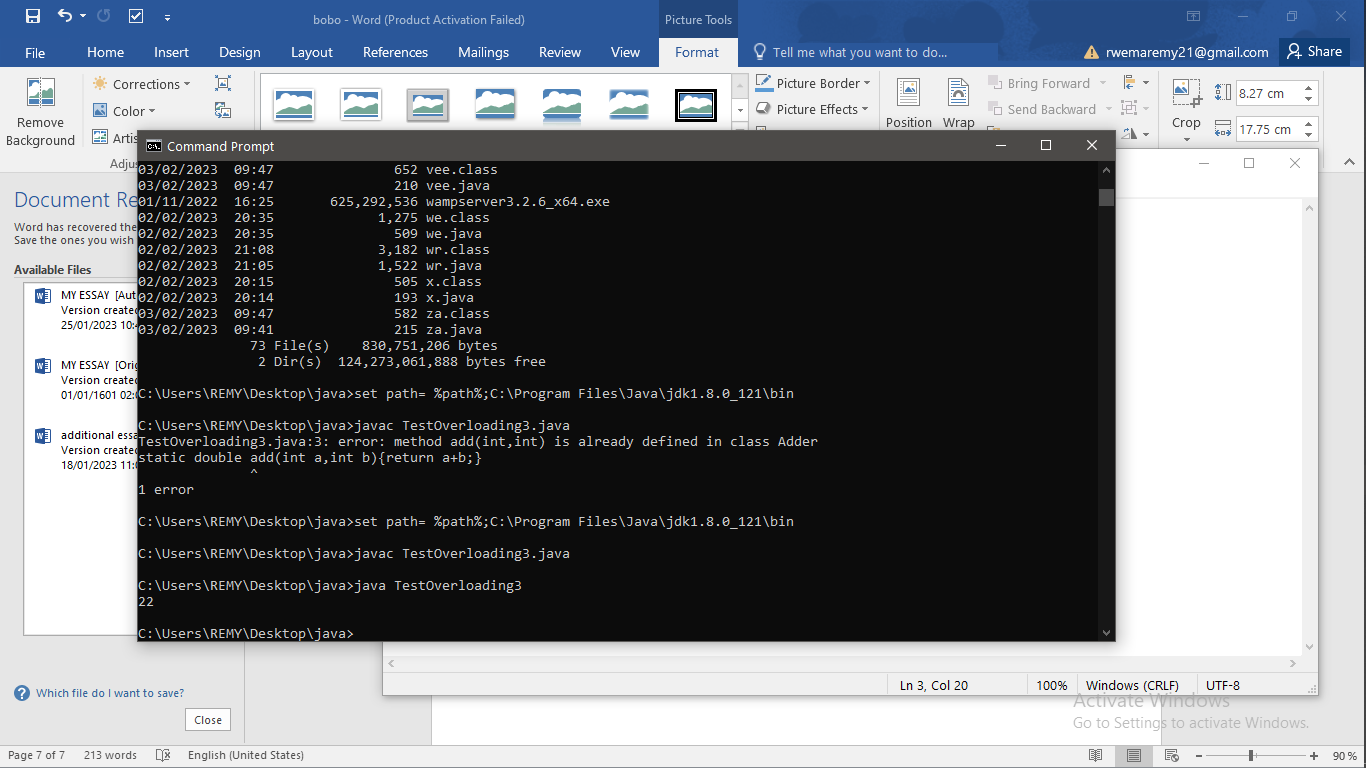
}

class TestOverloading3{

public static void main(String[] args){

System.out.println(Adder.add(11,11));//ambiguity

}}



10.

abstract class Bike{

abstract void run();

}

class Honda4 extends Bike{

void run(){System.out.println("running safely");}

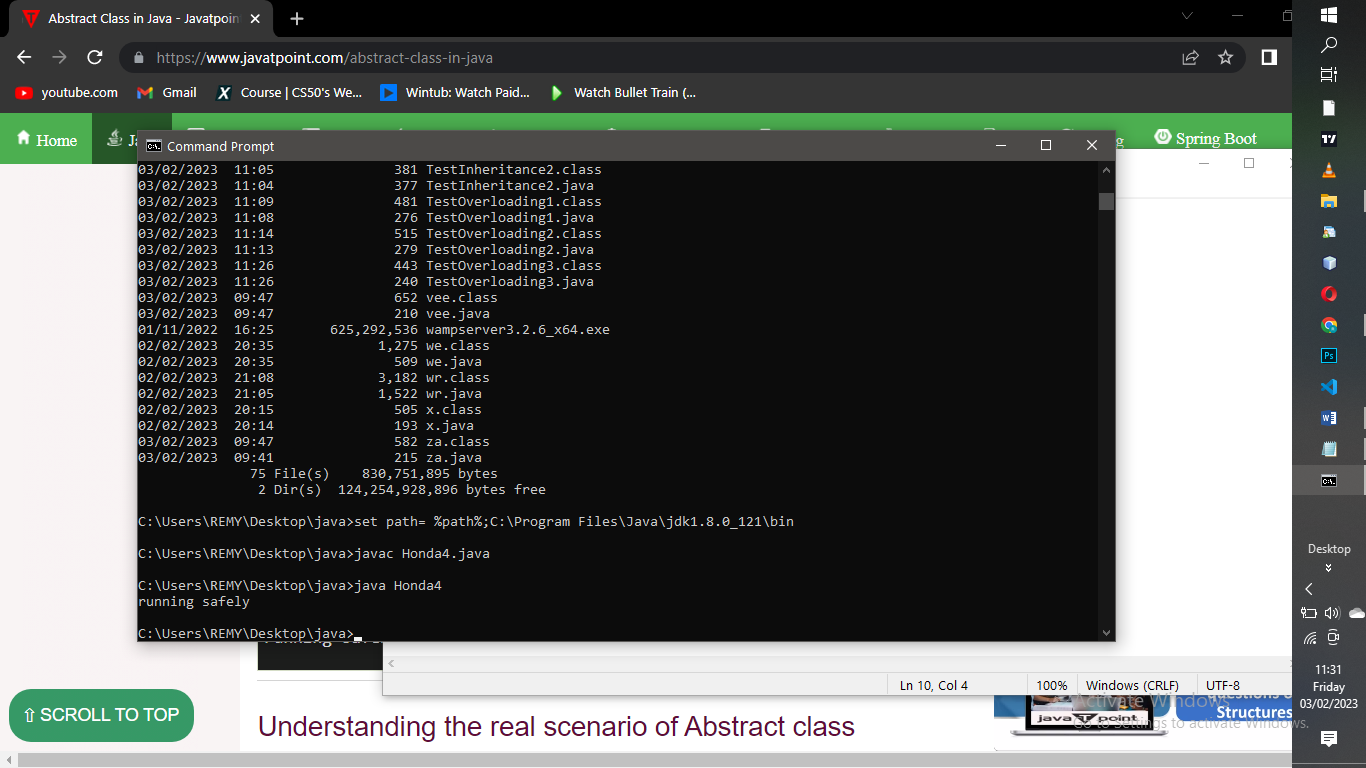
public static void main(String args[]){

Bike obj = new Honda4();

obj.run();

}

}



11.

abstract class Bank{

abstract int getRateOfInterest();

}

class SBI extends Bank{

int getRateOfInterest(){return 7;}

}

class PNB extends Bank{

int getRateOfInterest(){return 8;}

}

class TestBank{

public static void main(String args[]){

Bank b;

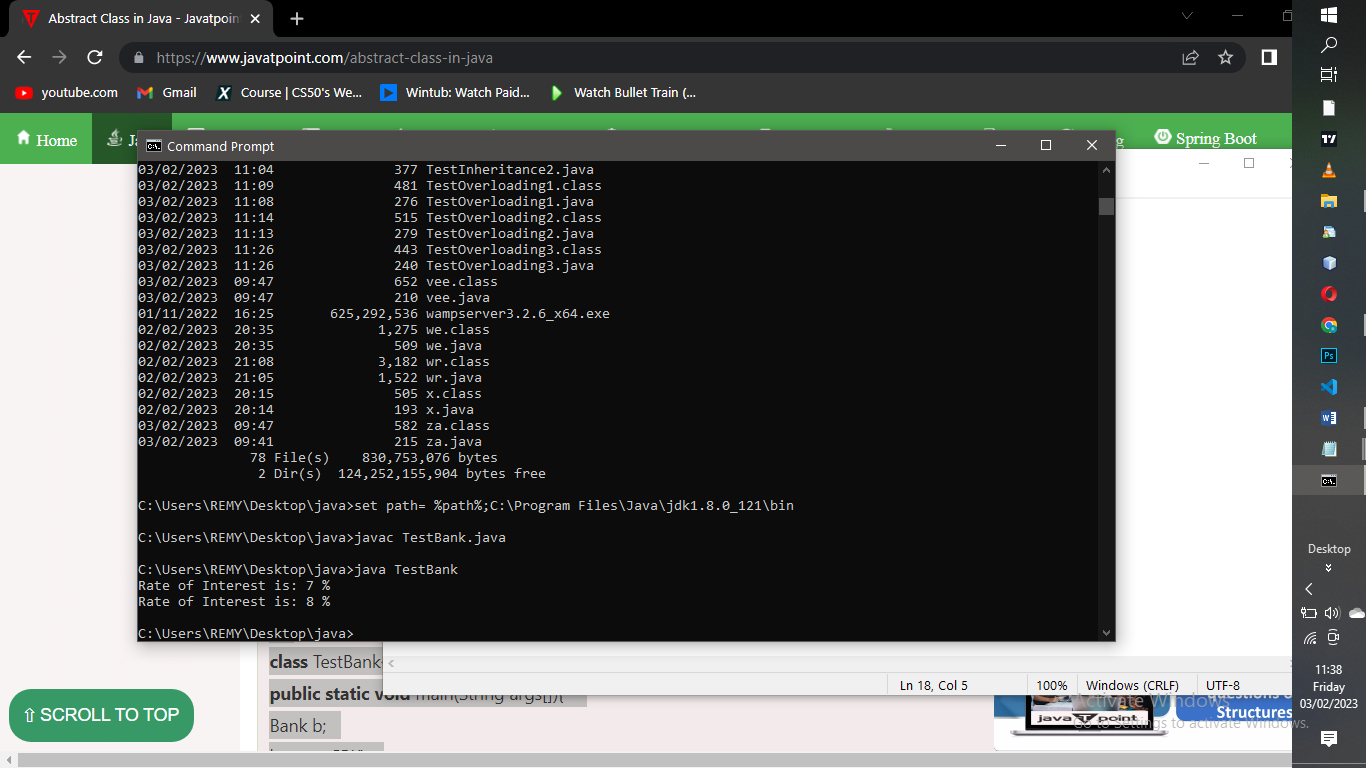
b=new SBI();

System.out.println("Rate of Interest is: "+b.getRateOfInterest()+" %");

b=new PNB();

System.out.println("Rate of Interest is: "+b.getRateOfInterest()+" %");

}}



12.

abstract class Bike{

Bike(){System.out.println("bike is created");}

abstract void run();

void changeGear(){System.out.println("gear changed");}

}

class Honda extends Bike{

void run(){System.out.println("running safely..");}

}

class TestAbstraction2{

public static void main(String args[]){

Bike obj = new Honda();

obj.run();

obj.changeGear();

}

}

