**1.AREAPERI**

class AreaPeri{

double side1;

double side2;

double side3;

AreaPeri(){

side1 = side2 = side3 = 1 ;

}

AreaPeri(double s1,double s2,double s3){

side1 = s1;

side2 = s2;

side3 = s3;

}

double getArea(){

double s = ( side1 + side2 + side3 ) / 2.0;

double area=Math.sqrt(s\*(s-side1)\*(s-side2)\*(s-side3));

return area;

}

double getPerimeter(){

return( side1 + side2 + side3 );

}

void set(double s1 , double s2 , double s3){

side1 = s1;

side2 = s2;

side3 = s3;

}

void get(AreaPeri app){

side1 = app.side1;

side2 = app.side2;

side3 = app.side3;

return;

}

}

class Triangle{

public static void main(String args[]){

/\* for default constructor \*/

AreaPeri ap = new AreaPeri();

System . out . println("The area of the triangle" +ap.getArea());

System . out . println("The perimeter of the triangle" +ap.getPerimeter());

/\* for parameterized constructor \*/

AreaPeri ap1 = new AreaPeri(4,5,6);

System . out . println("The area of triangle" +ap1.getArea());

System . out . println("The perimeter of the triangle" +ap1.getPerimeter());

/\* for set method \*/

AreaPeri ap2 = new AreaPeri();

ap2.set(1.5,2.5,3.5);

System . out . println("The area of the triangle" +ap2.getArea());

System . out . println("The perimeter of the triangle" +ap2.getPerimeter());

/\* for get method \*/

AreaPeri ap3 = new AreaPeri();

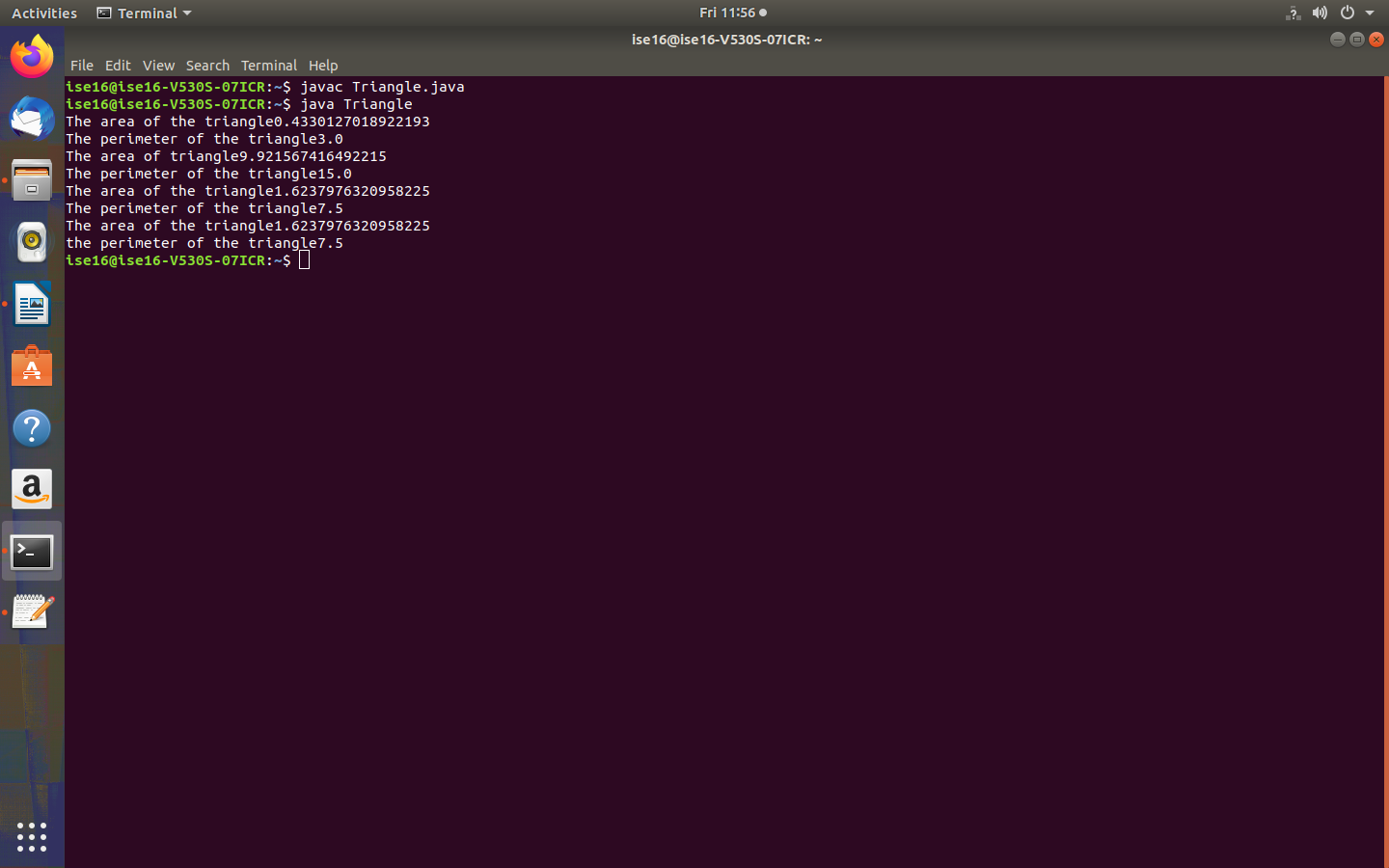
ap3.get(ap2);

System . out . println("The area of the triangle" +ap3.getArea());

System . out . println("the perimeter of the triangle" +ap3.getPerimeter());

}

}



**2.TRUNKCALL**

class TrunkCall

{

double duration;

double charge;

TrunkCall()

{

duration=0;

}

TrunkCall(double d)

{

duration=d;

}

void calcCharge()

{

System.out.println("No Policy");

}

}

class OrdinaryCall extends TrunkCall

{

double call\_rate;

OrdinaryCall()

{

super();

call\_rate=0.60;

}

OrdinaryCall(double d)

{

super(d);

call\_rate=0.60;

}

OrdinaryCall(double d , double f)

{

super(d);

call\_rate=f;

}

void calcCharge()

{

charge=duration\*call\_rate;

System.out.println("For OrdinaryCall charge:" +charge);

}

}

class UrgentCall extends TrunkCall

{

double call\_rate;

UrgentCall()

{

super();

call\_rate=1.0;

}

UrgentCall(double d)

{

super(d);

call\_rate=1.0;

}

UrgentCall(double d , double f)

{

super(d);

call\_rate=f;

}

void calcCharge()

{

charge=duration\*call\_rate;

System.out.println("For UrgentCall charge:" +charge);

}

}

class LightningCall extends TrunkCall

{

double call\_rate;

LightningCall()

{

super();

call\_rate=1.2;

}

LightningCall(double d)

{

super(d);

call\_rate=1.2;

}

LightningCall(double d , double f)

{

super(d);

call\_rate=f;

}

void calcCharge()

{

charge=duration\*call\_rate;

System.out.println("For LightningCall charge:" +charge);

}

}

class Telephone

{

public static void main(String args[])

{

TrunkCall tref;

OrdinaryCall ordCall=new OrdinaryCall(4);

UrgentCall urgCall=new UrgentCall(1.0,2.0);

LightningCall ligCall=new LightningCall(2.0,3.0);

tref=ordCall;

tref.calcCharge();

tref=urgCall;

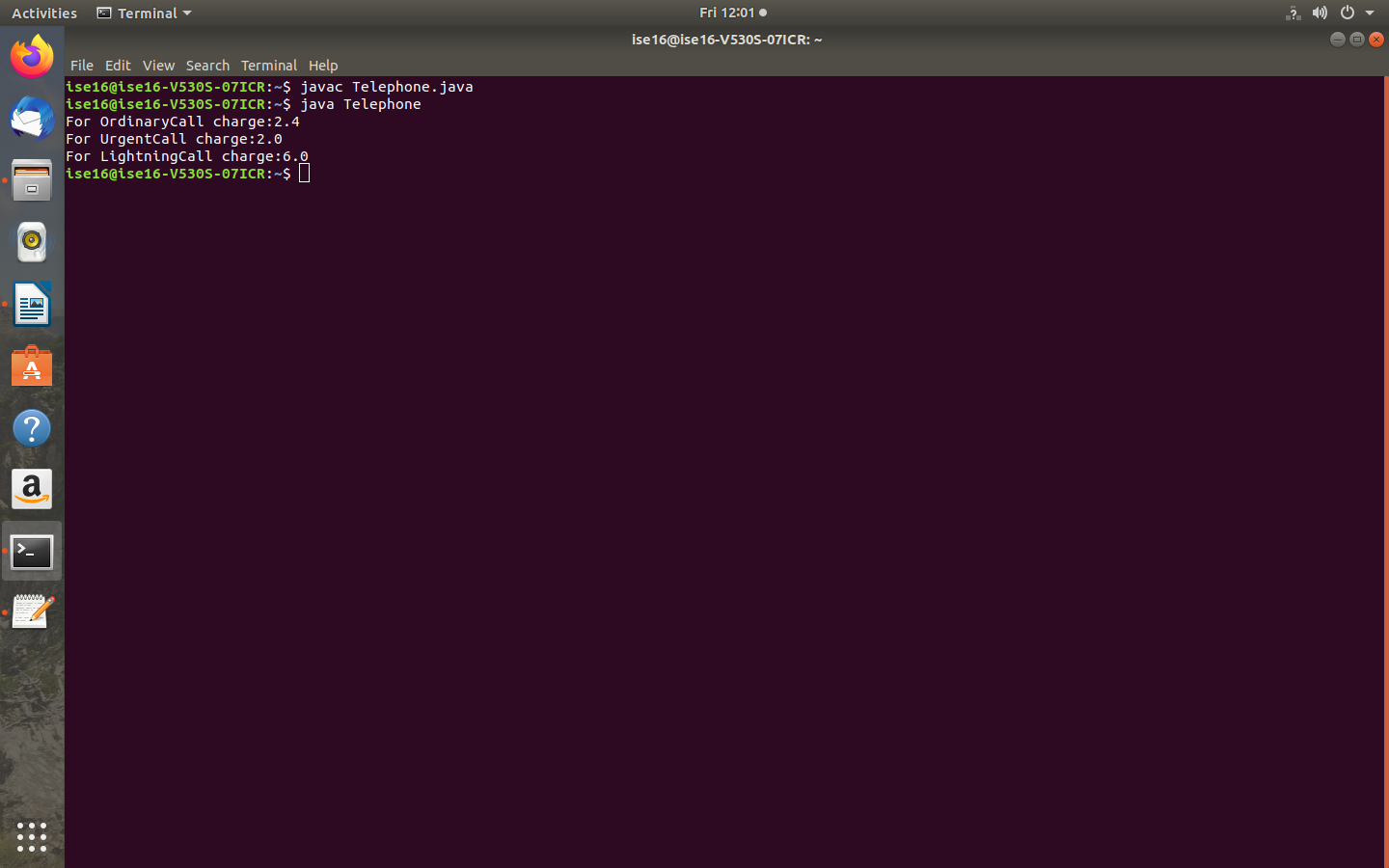
tref.calcCharge();

tref=ligCall;

tref.calcCharge();

}

}



**3.ABSTRACT CLASS**

import java.util.\*;

abstract class Shape{

int length,breadth,radius;

Scanner input=new Scanner(System.in);

abstract void printArea();

}

class Rectangle extends Shape{

void printArea(){

System.out.println("\*\*\*finding the Area of Rectangle\*\*\*");

System.out.println("enter the length and breadth:");

length=input.nextInt();

breadth=input.nextInt();

System.out.println("The Area of Rectangle is:" +(length\*breadth));

}

}

class Triangle extends Shape{

void printArea(){

System.out.println("\*\*\*Finding the Area of Triangle\*\*\*");

System.out.println("enter Base and Height");

length=input.nextInt();

breadth=input.nextInt();

System.out.println("The Area of Triangle is:" +(length\*breadth)/2);

}

}

class Circle extends Shape{

void printArea(){

System.out.println("\*\*\*Finding the Area of Circle\*\*\*");

System.out.println("enter the radius:");

radius=input.nextInt();

System.out.println("The Area of Circle is:" +(3.14f\*radius\*radius));

}

}

public class AbstractClassExample{

public static void main(String args[]){

Rectangle rec=new Rectangle();

rec.printArea();

Triangle tri=new Triangle();

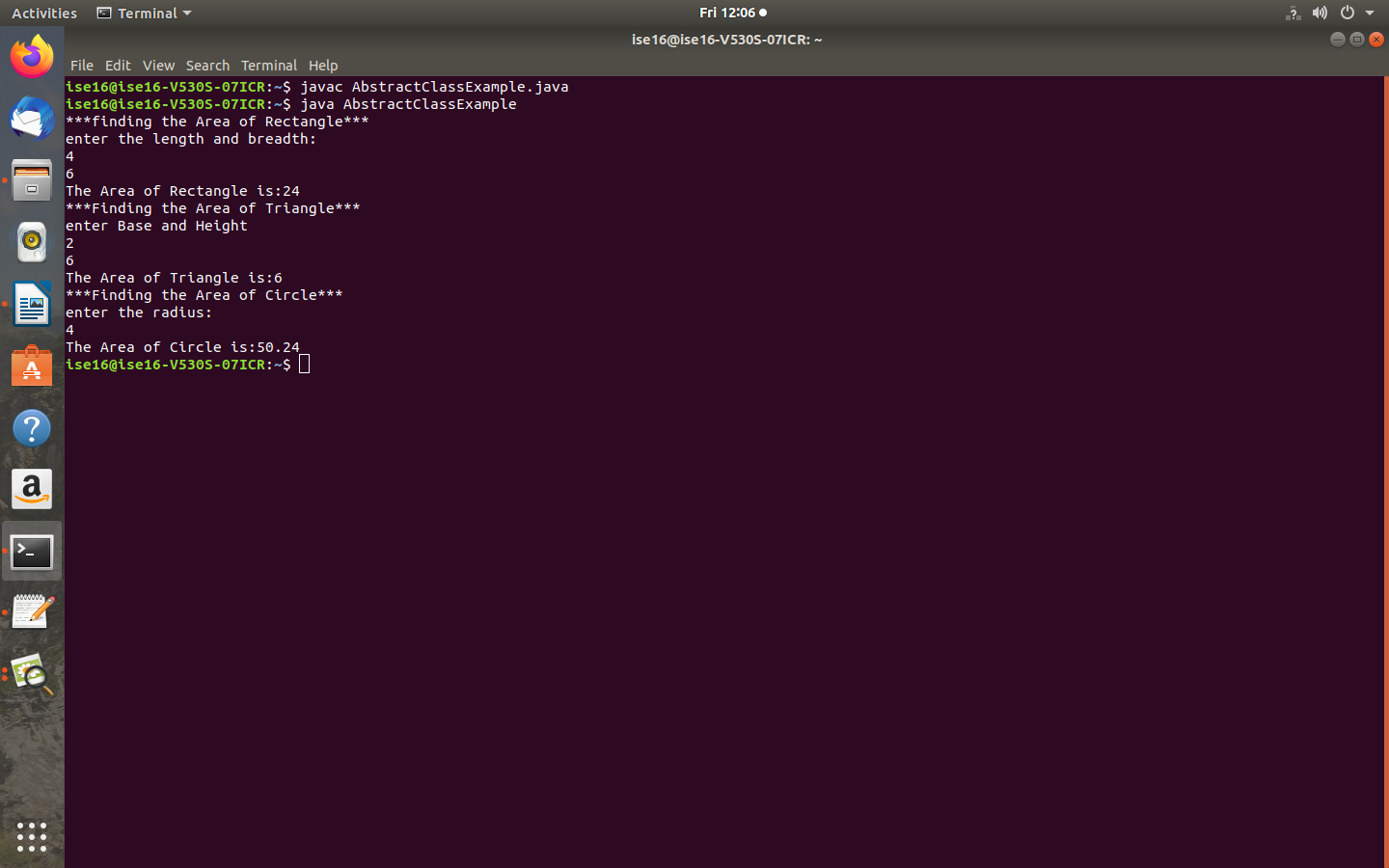
tri.printArea();

Circle cri=new Circle();

cri.printArea();

}

}



**5.PLAYERS**

class Player{

String name;

int age,matches,ranking;

Player(String n,int a,int m,int r){

name=n;

age=a;

matches=m;

ranking=r;

}

}

class CricketPlayer extends Player{

int High\_score,Bowl\_average,Bat\_average;

CricketPlayer(String a,int b,int c,int d,int e,int f,int g){

super(a,b,c,d);

High\_score=e;

Bat\_average=f;

Bowl\_average=g;

}

void disp(){

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

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

System.out.println("No. of matches:" +matches);

System.out.println("Highscore:" +High\_score);

System.out.println("Batting average:" +Bat\_average);

System.out.println("Balling average:" +Bowl\_average);

System.out.println("Player ranking:" +ranking);

}

}

class FootballPlayer extends Player{

int goals,g\_avg,pass;

FootballPlayer(String a,int b,int c,int d,int e,int f, int g){

super(a,b,c,d);

goals=e;

g\_avg=f;

pass=g;

}

void disp(){

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

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

System.out.println("no.of matches:" +matches);

System.out.println("no.of Goals:" +goals);

System.out.println("goals average:" +g\_avg);

System.out.println("passing efficiency:" +pass+"%");

System.out.println("player ranking:" +ranking);

}

}

class HockeyPlayer extends Player{

int goals,g\_avg,pass;

HockeyPlayer(String a,int b,int c,int d,int e, int f,int g){

super(a,b,c,d);

goals=e;

g\_avg=f;

pass=g;

}

void disp(){

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

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

System.out.println("no.of matches:" +matches);

System.out.println("no. of goals:" +goals);

System.out.println("goal average:" +g\_avg);

System.out.println("passing efficiency:" +pass+"%");

System.out.println("player ranking:" +ranking);

}

}

class PlayerDemo{

public static void main(String args[]){

CricketPlayer C=new CricketPlayer("sachin Tendulkar",38,600,8,200,55,60);

FootballPlayer F=new FootballPlayer("Leonel Messi",32,120,90,3,80,94);

HockeyPlayer H=new HockeyPlayer("Dhanraj Pillay",32,120,90,3,80,94);

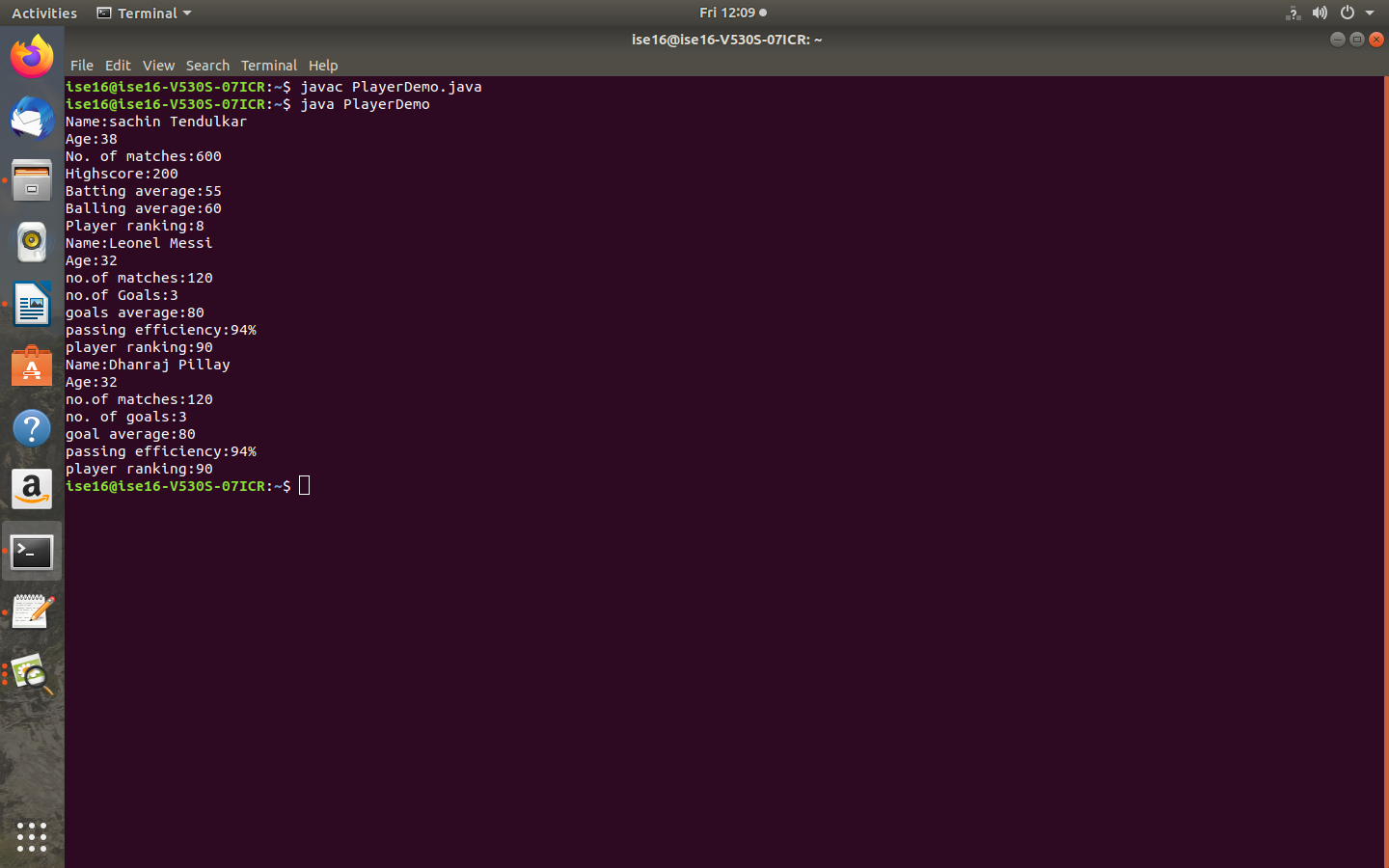
C.disp();

F.disp();

H.disp();

}

}



**6.INTERFACE**

//InterfaceMain.java

import java.io.\*;

interface area{

float compute(float x,float y);

}

class rectangle{

public float compute(float x,float y)

{

return(x\*y);

}

}

class triangle{

public float compute(float x,float y)

{

return(x\*y/2);

}

}

class result extends rectangle implements area{

public float compute(float x,float y)

{

return(x\*y);

}

}

class result1 extends triangle implements area{

public float compute(float x,float y)

{

return(x\*y/2);

}

}

class InterfaceMain{

public static void main(String args[]){

result rect=new result();

result1 tri=new result1();

area a;

a=rect;

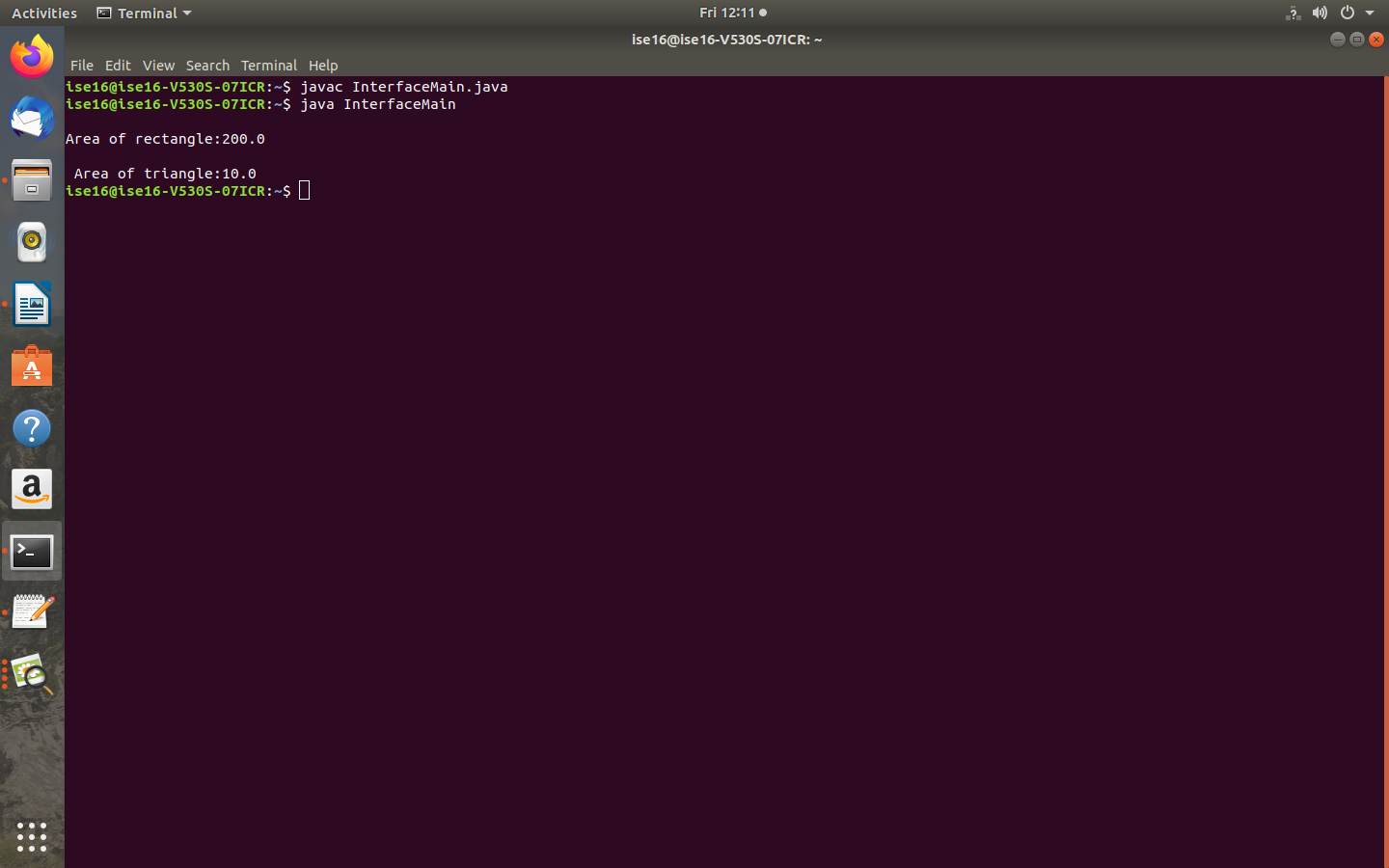
System.out.println("\nArea of rectangle:" +a.compute(10,20));

a=tri;

System.out.println("\n Area of triangle:" +a.compute(10,2));

}

}



**7.ACCOUNT BALANCE**

package Balance;

public class Account

{

double principal,rate,balance;

int time;

public Account(double pr,int ti,double ra)

{

principal=pr;

time=ti;

rate=ra;

}

public void calcAmount()

{

balance=principal\*rate\*time;

}

public void DisplayBalance()

{

System.out.println("\n\n Principal Amount:" +principal+ "Rs\n Time:" +time+ "years\n\n current Balance:" +balance+"Rs");

}

}

import Balance.Account;

class DemoPackage

{

public static void main(String args[])

{

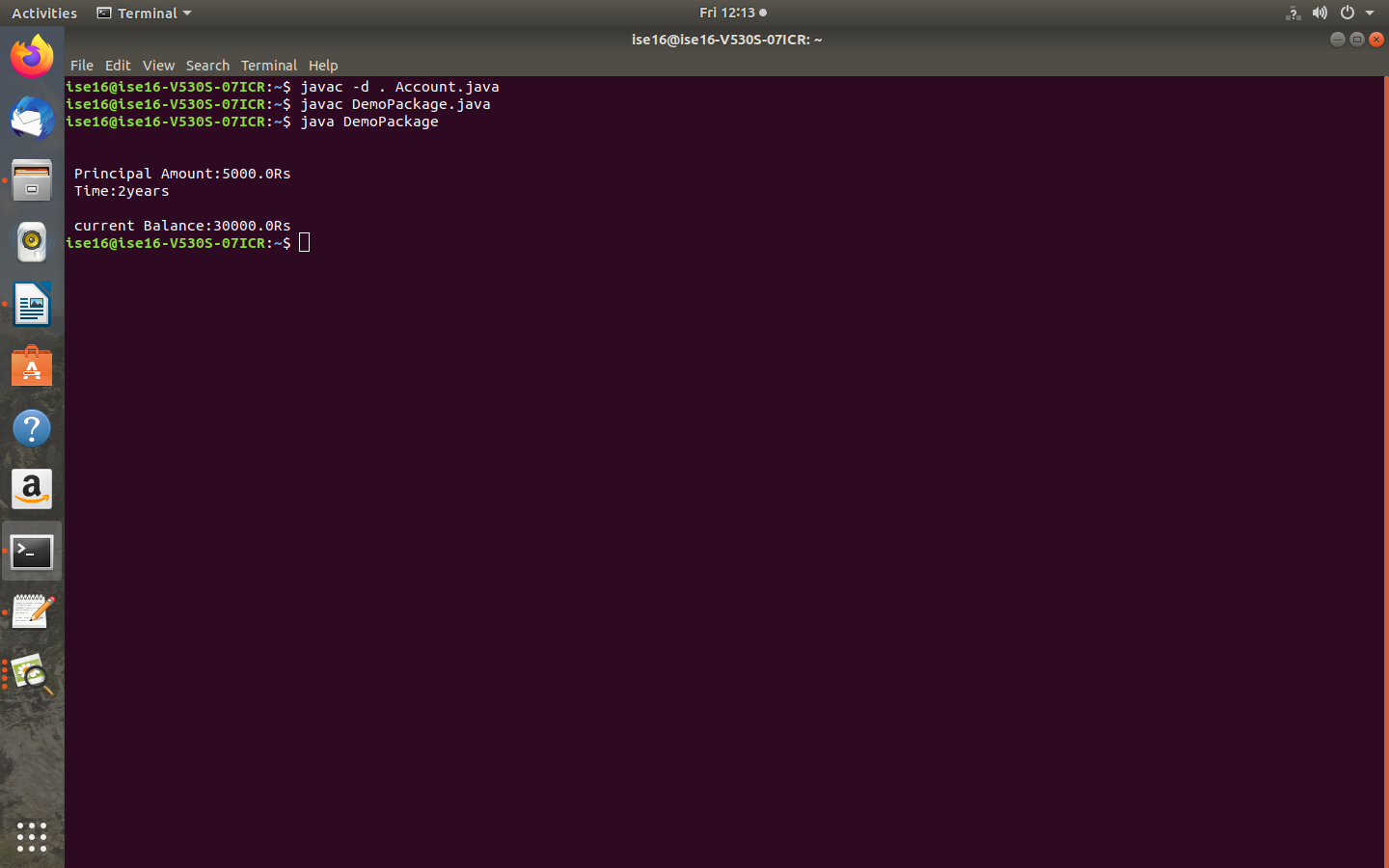
Account acc=new Account(5000,2,3);

acc.calcAmount();

acc.DisplayBalance();

}

}



**9.COLLISSION**

class Collision

{

String DirectionTrain1,DirectionTrain2;

Collision(String dir1,String dir2)

{

DirectionTrain1=dir1;

DirectionTrain2=dir2;

}

void checkCollision()

{

try

{

if(DirectionTrain1==DirectionTrain2)

{

System.out.println("The two vehicles are moving in same direction,hence no collision in pair1");

}

else

{

throw new Exception("The two vehicles are moving in opposite directions,so collission occurs in pair2");

}

}

catch(Exception e)

{

System.out.println(e);

}

}

}

class ExceptionDemo{

public static void main(String args[])

{

Collision pair1=new Collision("north","north");

Collision pair2=new Collision("north","south");

pair1.checkCollision();

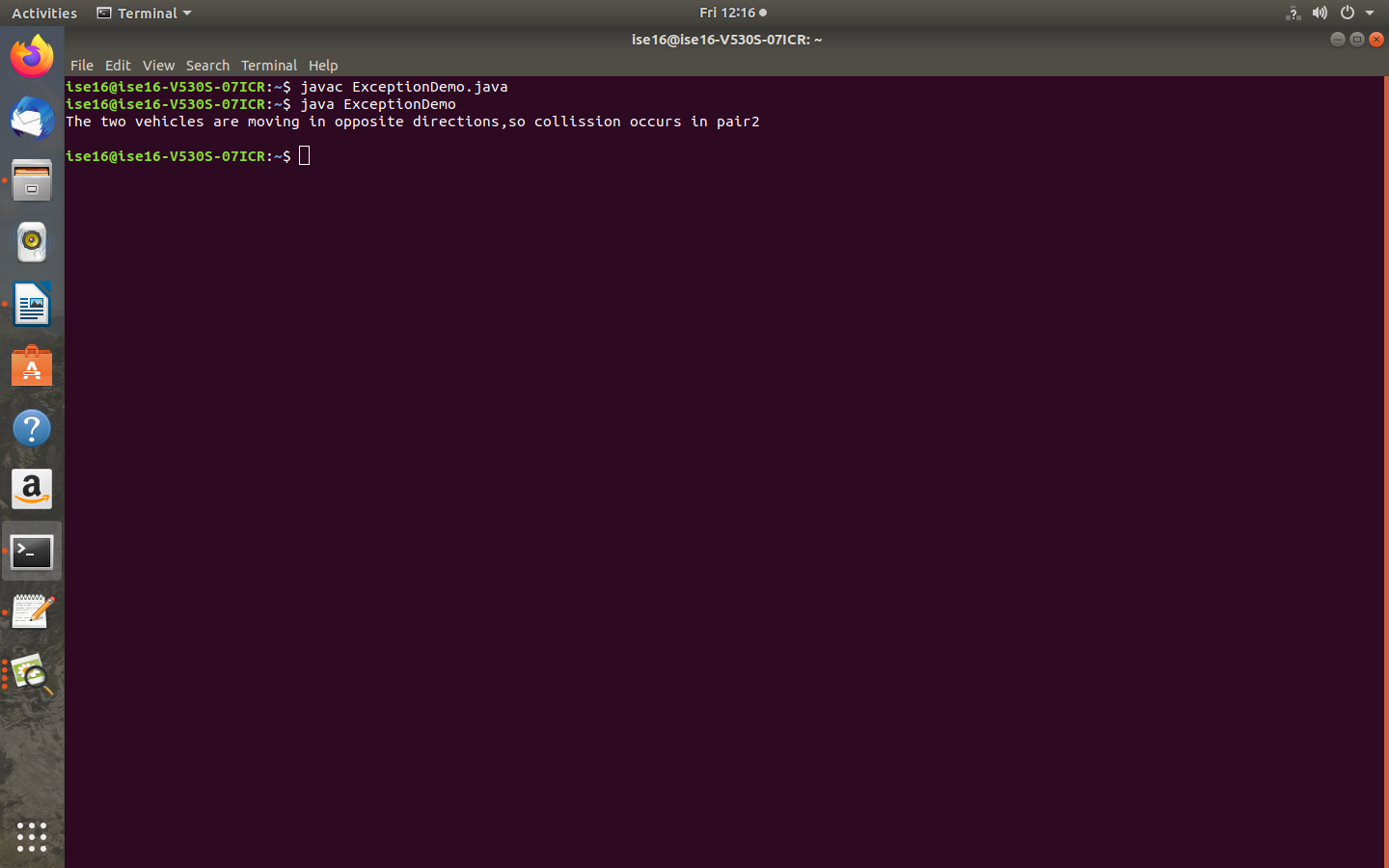
System.out.println();

pair2.checkCollision();

System.out.println();

}

}



**8.STACK OPERATION**

import java.io.\*;

interface operations

{

void push();

void pop();

void display();

}

class Fstack implements operations

{

private int top=-1;

private int[] fstack=new int[5];

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

public void push()

{

if(top==fstack.length-1)

System.out.println("\n Stack is Overflow");

else

{

try

{

System.out.println("enter the item:\t");

fstack[++top]=Integer.parseInt(br.readLine());

}

catch(Exception e)

{

System.out.println(e);

}

}

}

public void pop()

{

if(top==-1)

System.out.println("\n stack is underflow");

else

System.out.println("\n Deleted item is:\b" +fstack[top--]);

}

public void display()

{

if(top==-1)

System.out.println("\n stack is empty");

else

{

System.out.println("the elements of stack are:");

for(int i=top;i>=0;i--)

System.out.println(fstack[i]);

}

}

}

class Dstack implements operations

{

private int top=-1;

private int[] dstack;

Dstack(int size)

{

dstack=new int[size];

}

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

public void push()

{

if(top==dstack.length-1)

System.out.println("\n Stack Overflow");

else

{

try

{

System.out.println("enter the item:\t");

dstack[++top]=Integer.parseInt(br.readLine());

}

catch(Exception e)

{

System.out.println(e);

}

}

}

public void pop()

{

if(top==-1)

System.out.println("\n stack underflow");

else

System.out.println("\n the deleted item:\b" +dstack[top--]);

}

public void display()

{

if(top==-1)

System.out.println("\n stack is empty");

else

System.out.println(" the elements of stack are:");

for(int i=top;i>=0;i--)

System.out.println(dstack[i]);

}

}

class stackinterface

{

public static void main(String args[])throws IOException

{

int ch;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

while(true)

{

System.out.println("\n1.Fixed\_stack\n 2.Dynamic\_stack\n 3.exit");

System.out.println("\n enter your choice:\t");

ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1:operations obj=new Fstack();

while(true)

{

System.out.println("\n1.PUSH\n2.POP\n3.DISPLAY\n4.MAIN\_MENU");

System.out.println("enter your choice:\t");

ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1:obj.push();

break;

case 2:obj.pop();

break;

case 3:obj.display();

break;

case 4:break;

default:System.out.println("invalid choice");

}

if(ch==4)

break;

}

break;

case 2:System.out.println("enter the size of array\n");

int size=Integer.parseInt(br.readLine());

operations obj1=new Dstack(size);

while(true)

{

System.out.println("\n 1.PUSH\n2.POP\n3.DISPLAY\n4.MAIN\_MENU");

System.out.println("enter your choice:\t");

ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1:obj1.push();

break;

case 2:obj1.pop();

break;

case 3:obj1.display();

break;

case 4:break;

default:System.out.println("invalid choice");

}

if(ch==4)

break;

}

break;

case 3:System.exit(0);

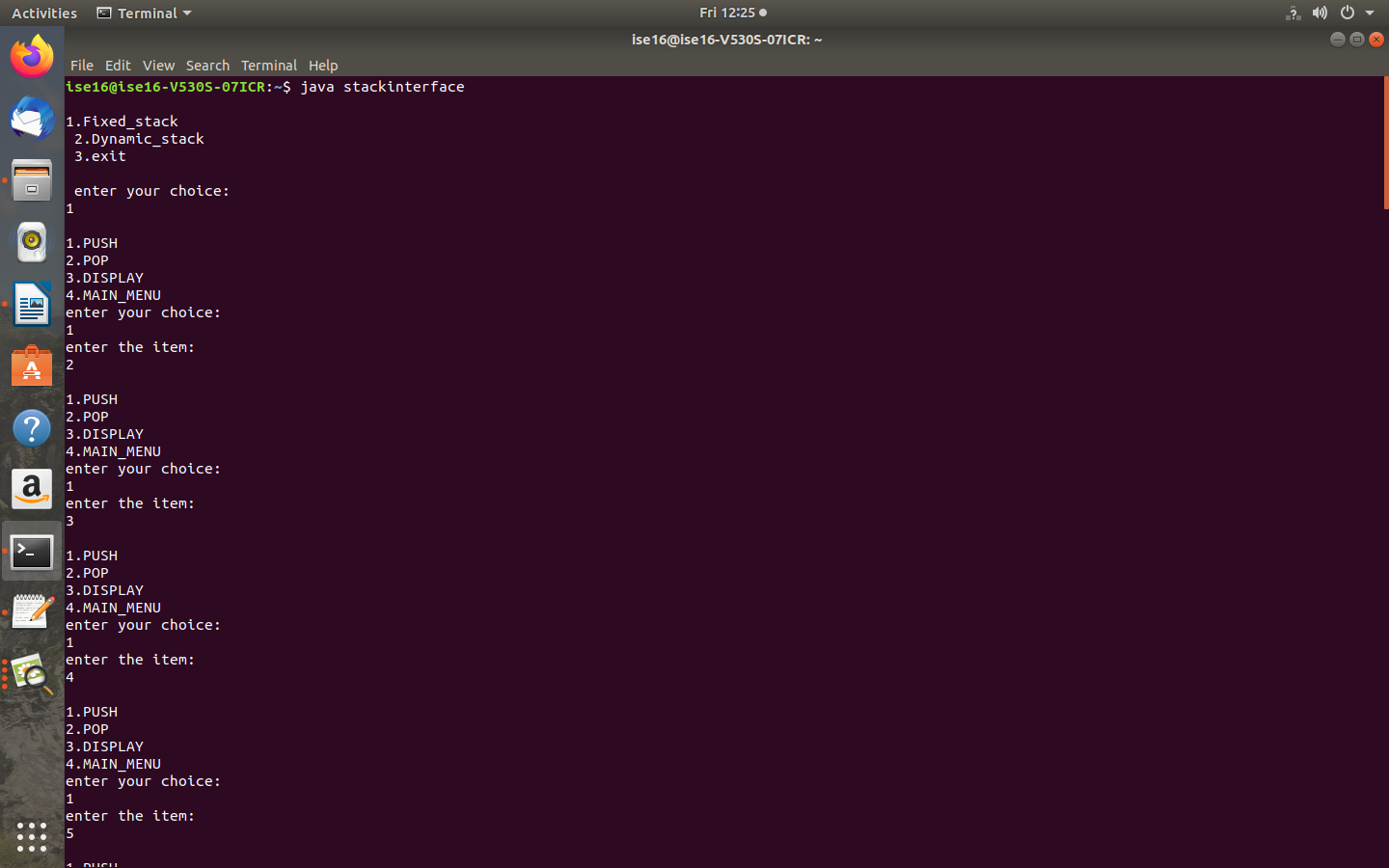
default:System.out.println("\ninvalid choice");

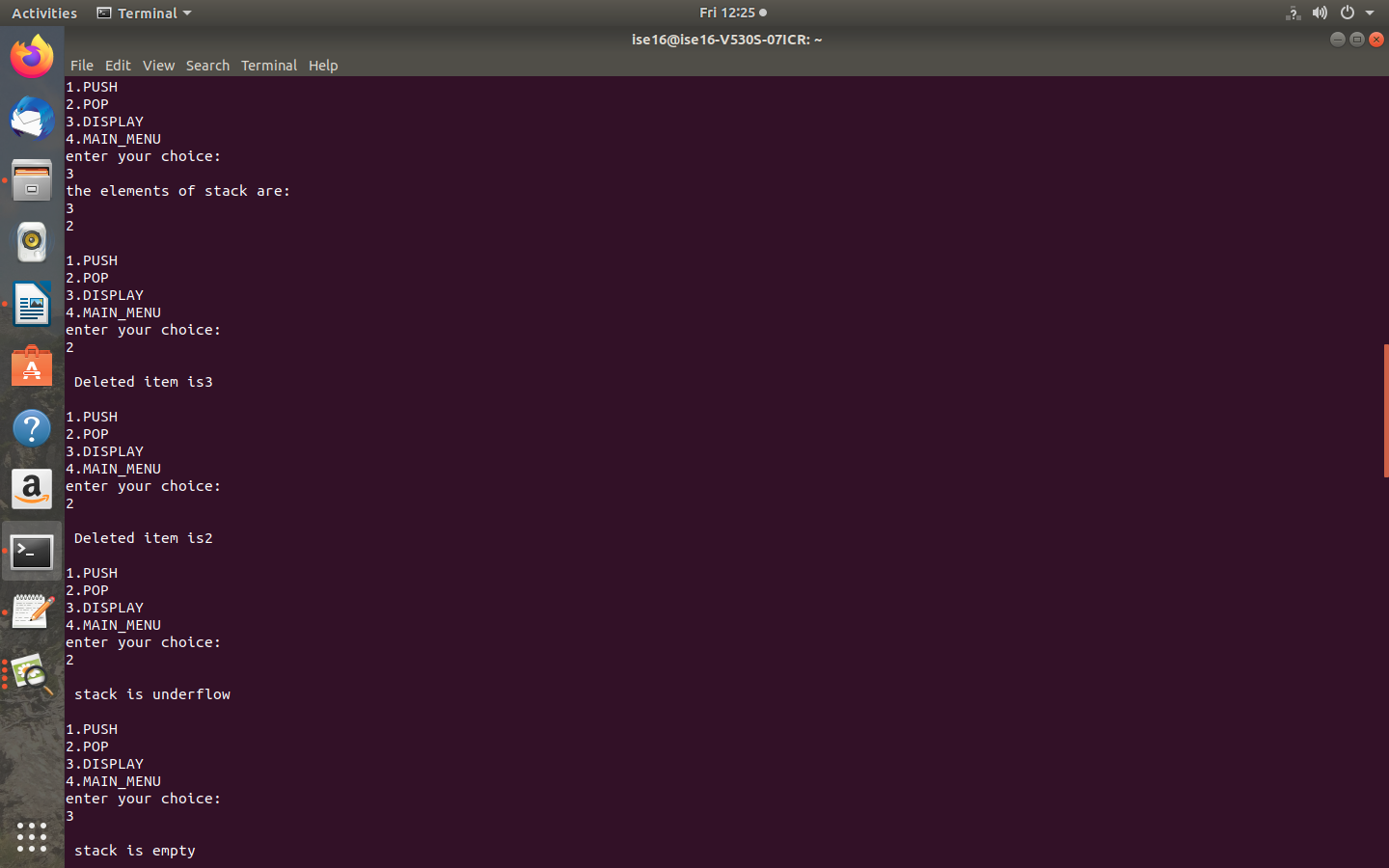
}

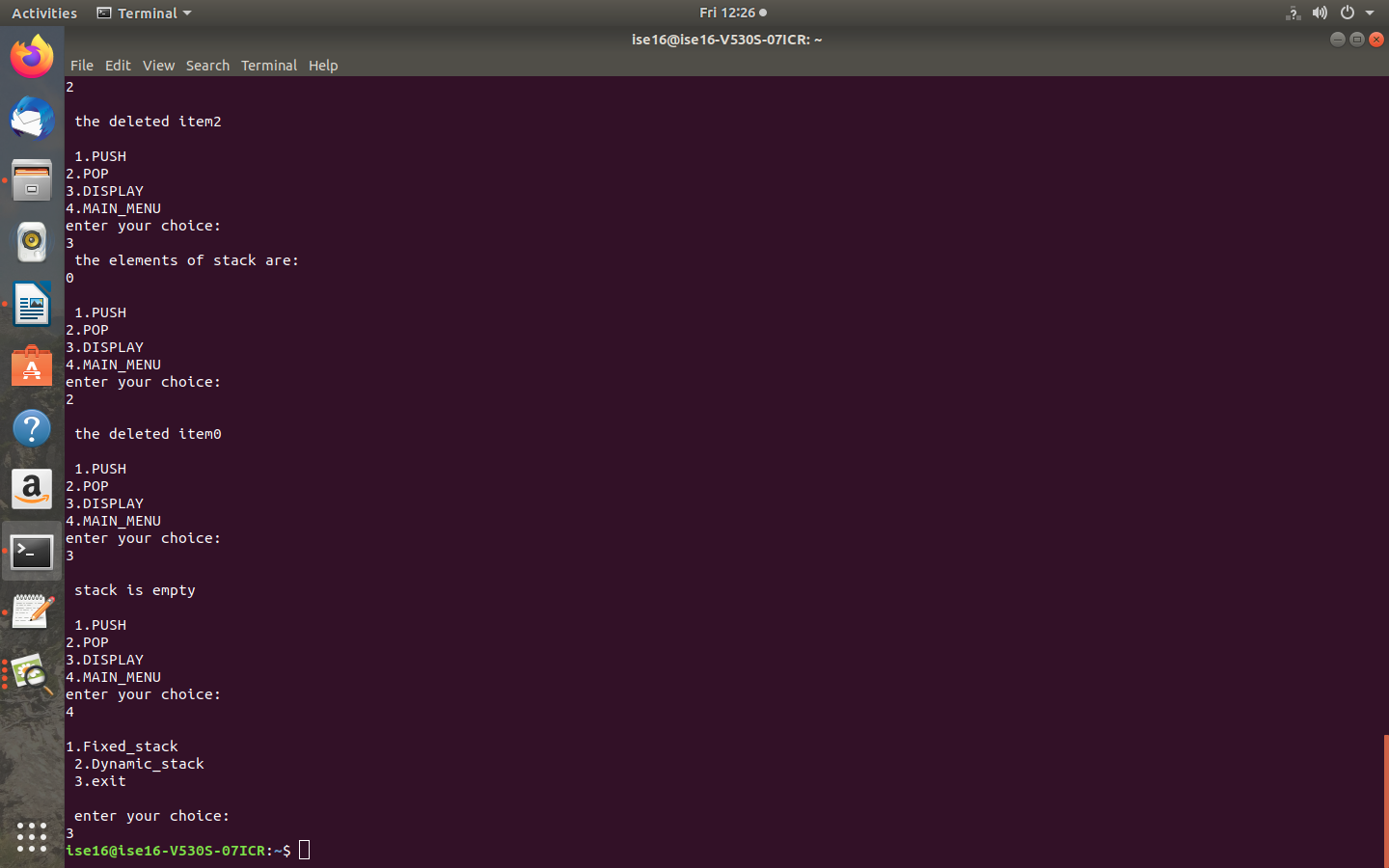
}

}

}







**10.ENUMARATION**

import java.io.\*;

class Enumaration

{

public enum DayOfWeek

{

MONDAY(1),TUESDAY(2),WEDNESDAY(3),THURSDAY(4),FRIDAY(5),SATURDAY(6),SUNDAY(7);

public int val;

DayOfWeek(int val)

{

this.val=val;

}

boolean isWorkDay()

{

if(val<6)

return true;

else

return false;

}

}

public static void main(String args[])

{

DayOfWeek Day;

System.out.println("verification of sunday(isWorkDay())");

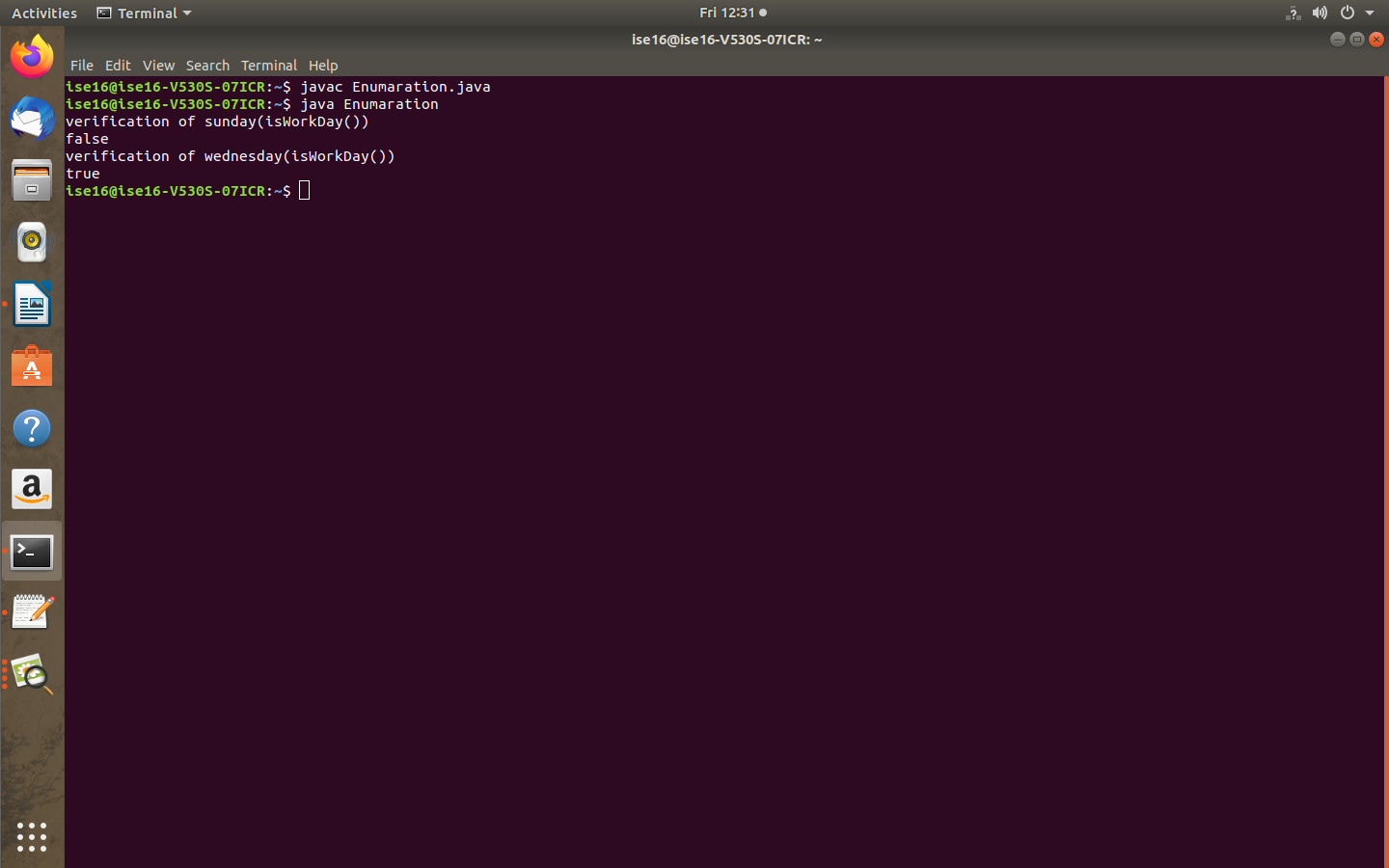
System.out.println(DayOfWeek.SUNDAY.isWorkDay());

System.out.println("verification of wednesday(isWorkDay())");

System.out.println(DayOfWeek.WEDNESDAY.isWorkDay());

}

}



**4. PACKAGE**

package p1;

public class Protection{

int n = 1;

private int n\_pri = 2;

protected int n\_pro = 3;

public int n\_pub = 4;

public Protection(){

System.out.println("Base Constructor");

System.out.println("n=" +n);

System.out.println("n\_pri" +n\_pri);

System.out.println("n\_pro" +n\_pro);

System.out.println("n\_pub" +n\_pub);

}

}

package p1;

class Derived extends Protection{

Derived(){

System.out.println("Derived Constructor");

System.out.println("n=" +n);

//class only

// System.out.println("n\_pri" +n\_pri);

System.out.println("n\_pro" +n\_pro);

System.out.println("n\_pub" +n\_pub);

}

}

package p1;

class SamePackage{

SamePackage(){

Protection p = new Protection();

System.out.println("Same Package Constructor");

System.out.println("n=" +p.n);

//class only

//System.out.println("n\_pri" +p.n\_pri);

System.out.println("n\_pro" +p.n\_pro);

System.out.println("n\_pub" +p.n\_pub);

}

}

package p1;

public class Demo{

public static void main(String args[]){

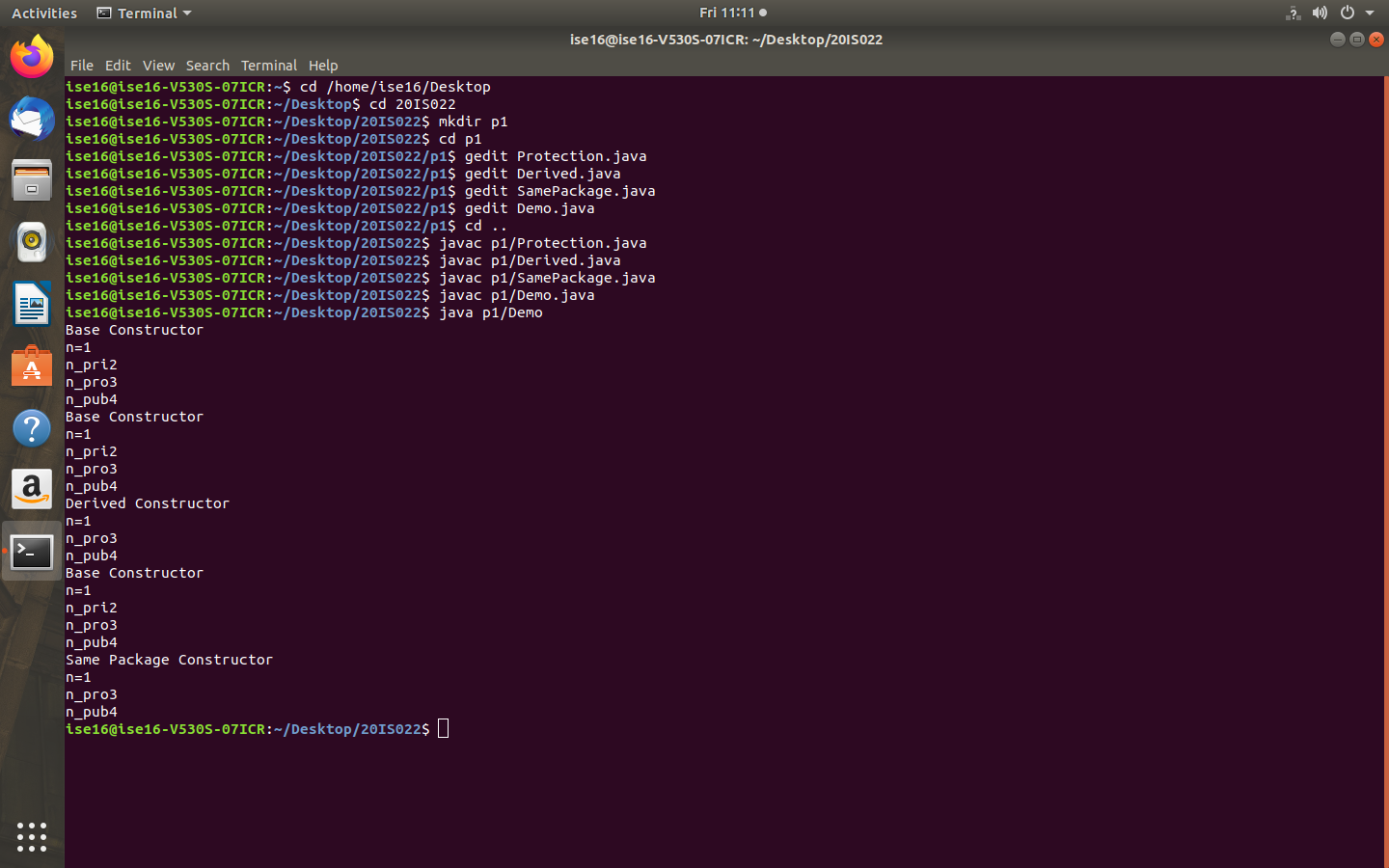
Protection ob1=new Protection();

Derived ob2=new Derived();

SamePackage ob3=new SamePackage();

}

}



package p2;

class Protection2 extends p1.Protection{

Protection2(){

System.out.println("Derived Other Package Constructor");

//class or package only

//System.out.println("n=" +n);

//class only

//System.out.println("n\_pri" +n\_pri);

System.out.println("n\_pro" +n\_pro);

System.out.println("n\_pub" +n\_pub);

}

}

package p2;

class OtherPackage{

OtherPackage(){

p1.Protection p=new p1.Protection();

System.out.println("Other Package Constructor");

//class or package only

//System.out.println("n=" +p.n);

//class only

//System.out.println("n\_pri" +p.n\_pri);

//class,subclass or package only

//System.out.println("n\_pro" +p.n\_pro);

System.out.println("n\_pub" +p.n\_pub);

}

}

package p2;

public class Demo2{

public static void main(String args[]){

Protection2 ob1=new Protection2();

OtherPackage ob2=new OtherPackage();

}

}

