**Introduction-Java2-Assingment**

Ques 1:Create Java classes having suitable attributes for Library management system.Use OOPs concepts in your design.Also try to use interfaces and abstract classes.

Ans:**JAVA-CODE**

interface admininterface{

void addnewlibrarian(String name,int id,String email);

}

interface librarianinterface{

void librarianshowdetails();

void addnewstudent(String name1,int id1,String email1);

void addnewbook(String bookname,String author,String subject);

}

interface studentinterface{

void registerbook(String bookname);

void showbookdetails();

}

class Admin implements admininterface{

String librarianname;

int librarianid;

String librarianemail;

@Override

public void addnewlibrarian(String name, int id, String email) {

this.librarianname=name;

this.librarianid=id;

this.librarianemail=email;

}

}

class Librarian implements librarianinterface {

String bookname;

String author;

String subject;

Map<Integer,String> studentdata = new HashMap<Integer, String>();

Admin a = new Admin();

List<String> registeredbooks = new ArrayList<String>();

@Override

public void librarianshowdetails() {

System.out.println("Librarian name is:"+a.librariannamename);

System.out.println("Librarian id is"+a.librarianidid);

System.out.println("Librarian email is :"+a.librarianemailemail);

}

@Override

public void addnewstudent(String name1, int id1, String email1) {

studentdata.put(id1,name1);

}

@Override

public void addnewbook(String bookname1, String author1, String subject1) {

this.bookname=bookname1;

this.author=author1;

this.subject=subject1;

}

}

class Student implements studentinterface{

Librarian lb = new Librarian();

@Override

public void registerbook(String bookname){;

lb.registeredbooks.add(bookname);

}

@Override

public void showbookdetails(String bookname) {

System.out.println("Author of book is "+lb.author);

System.out.println("Book belongs to: "+lb.subject);

}

}

class Mainclass{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter 1 for student and 0 for librarian");

int response = sc.nextInt();

if(response == 1){

System.out.println("Enter 1 for login and 0 for signup");

int response1=sc.nextInt();

if(response1 ==1){

System.out.println("Enter id and name");

int id =sc.nextInt();

String name= sc.next();

Librarian lb = new Librarian();

if(lb.studentdata.containsKey(id) && lb.studentdata.containsValue(name)){

System.out.println("Login Sucessfull");

}

else{

System.out.println("Invalid input");

}

}

else{

System.out.println("Enter details to sign up");

String name = sc.next();

int id = sc.nextInt();

String email =sc.next();

Librarian lb = new Librarian();

lb.addnewstudent(name,id,email);

}

}

else if(response ==0){

System.out.println("Enter 1 for login and 0 for signup");

int response2 = sc.nextInt();

if(response2 ==1){

System.out.println("Enter id and name");

int id =sc.nextInt();

String name= sc.next();

Admin ad = new Admin();

if(ad.librarianname.equals(name) && ad.librarianid==id){

System.out.println("Login Sucessfull");

}

else{

System.out.println("Invalid input");

}

}

else{

System.out.println("Enter details to sign up");

String name = sc.next();

int id = sc.nextInt();

String email =sc.next();

Admin ad = new Admin();

ad.addnewlibrarian(name,id,email);

}

}

else{

System.out.println("Invalid Input");

}

}

}

Ques 2:WAP to sorting string without using string Methods?.

Ans: **JAVA-CODE**

public class StringSort{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter a string to sort");

String str = sc.nextLine();

String sortedstring ="";

int len =str.length();

char[] ans = new char[len+1];

for(int i=0; i<len; i++)

{

ans[i] = str.charAt(i);

}

for(int i=0; i<len-1; i++)

{

for(int j=i+1; j<len; j++)

{

if(ans[i] > ans[j])

{

char temp =ans[i];

ans[i] = ans[j];

ans[j] = temp;

}

}

}

for(int i=0; i<len; i++)

{

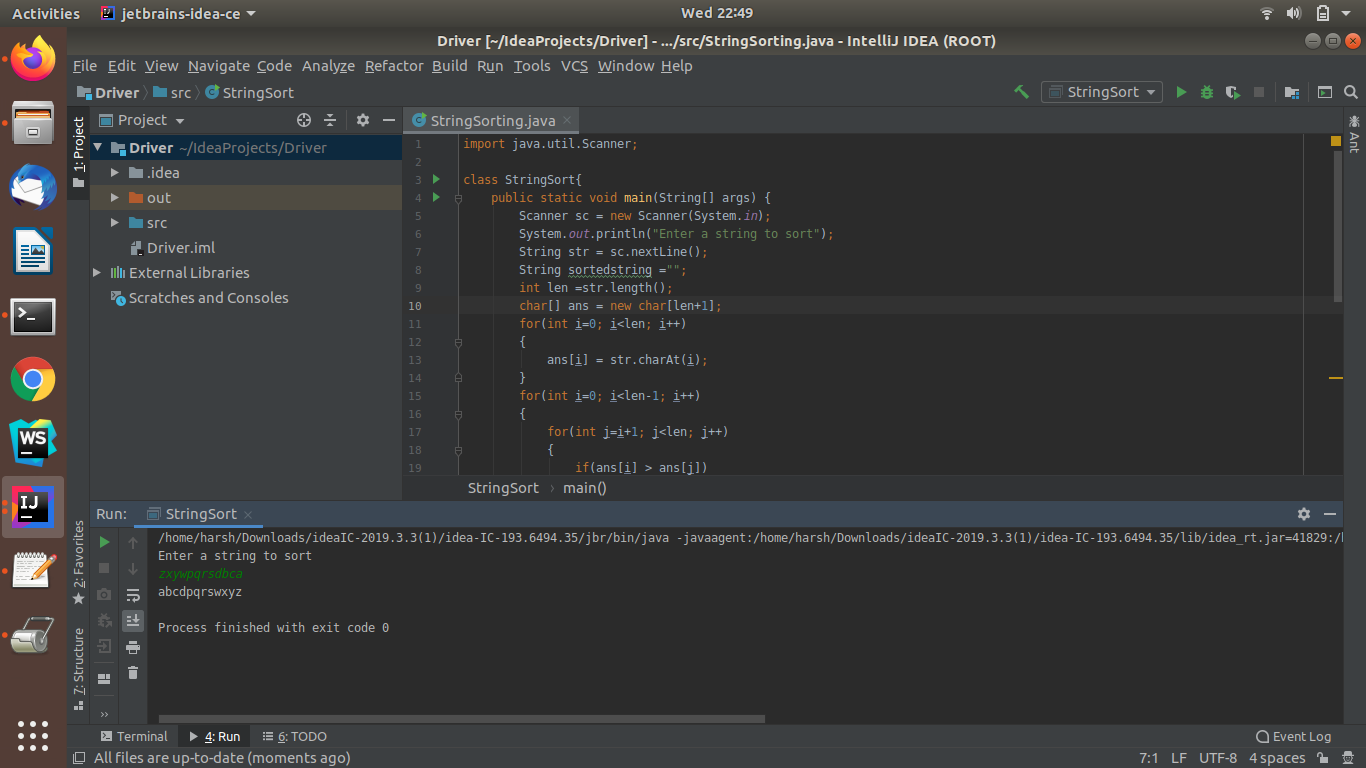
sortedstring = sortedstring+ans[i];

}

System.out.println(sortedstring);

}

}



Ques 3:WAP to produce NoClassDefFoundError and ClassNotFoundException exception.

Ans:**JAVA-CODE**

class Driver{

void showmessage(){

System.out.println("This is driver class message");

}

}

public class noClassDefFoundError{

public static void main(String[] args){

Driver d = new Driver();

d.showmessage();

}

}

//

public class classNotfoundException{

public static void main(String[] args){

try{

Class.forName("AnonymousClass");

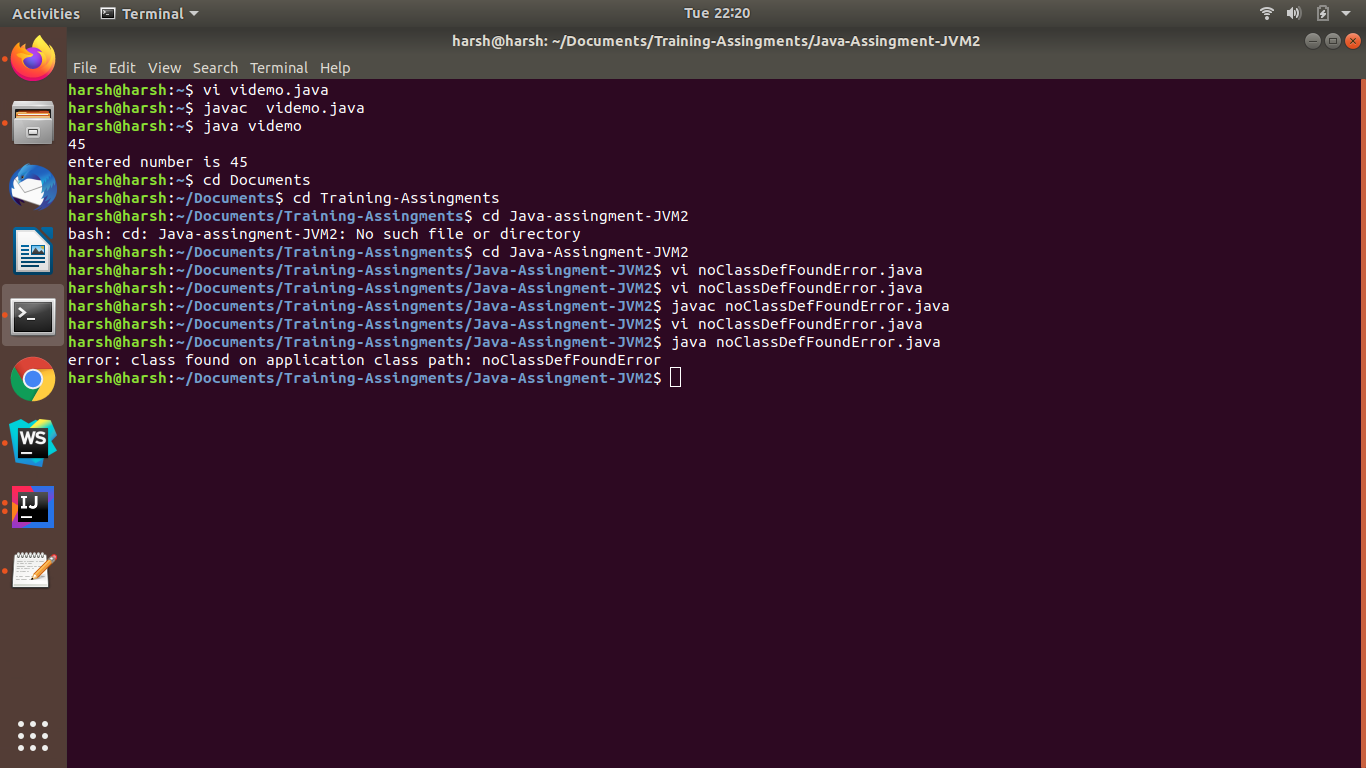
}

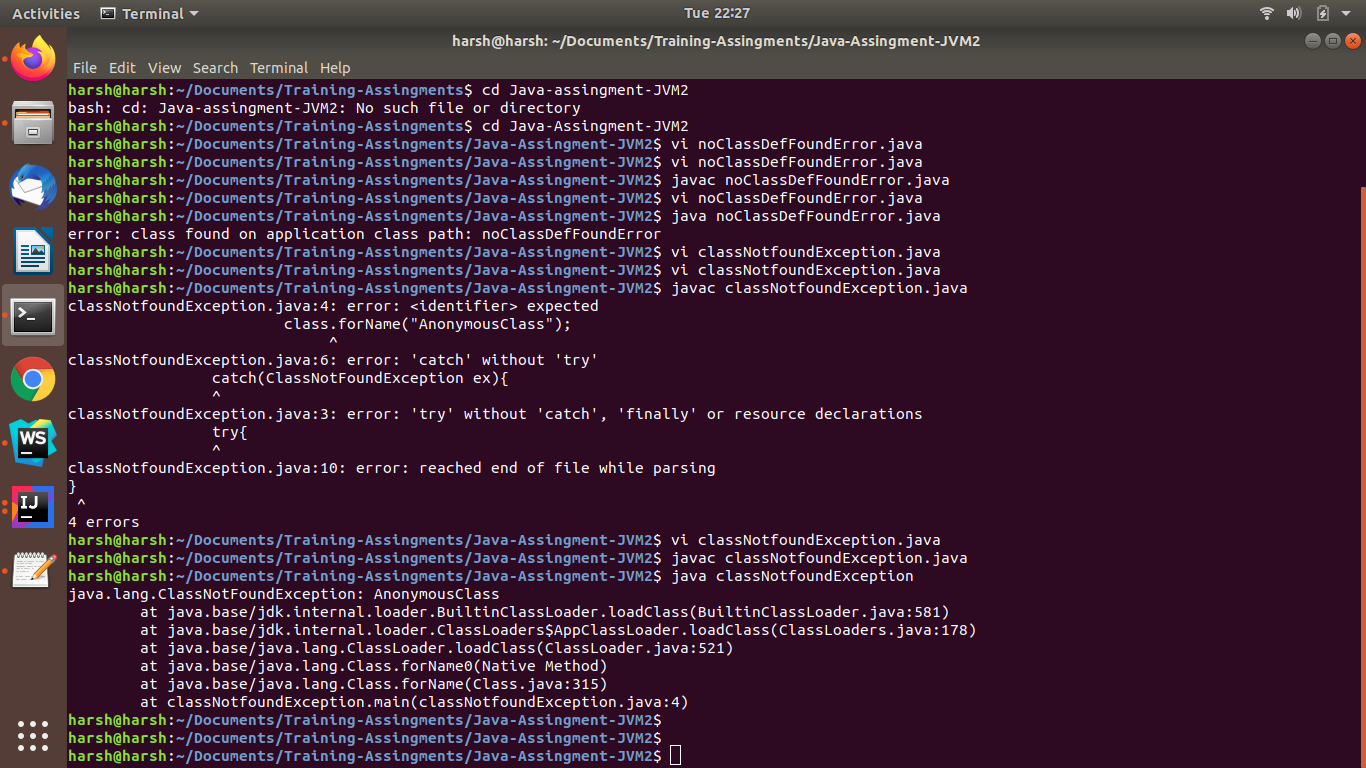
catch(ClassNotFoundException ex){

ex.printStackTrace();

}

}

}



Ques 4:WAP to create singleton class.

Ans:**JAVA-CODE**

public class Ques4{

public static void main(String[] args) {

singleton obj1 =singleton.getInstance();

singleton obj2 = singleton.getInstance();

System.out.println(obj1);

System.out.println(obj2);

obj1.show();

}

}

class singleton{

public static singleton s ;

private singleton(){

System.out.println("Instance Created\n");

}

public static singleton getInstance(){

if(s == null){

s = new singleton();

}

return s;

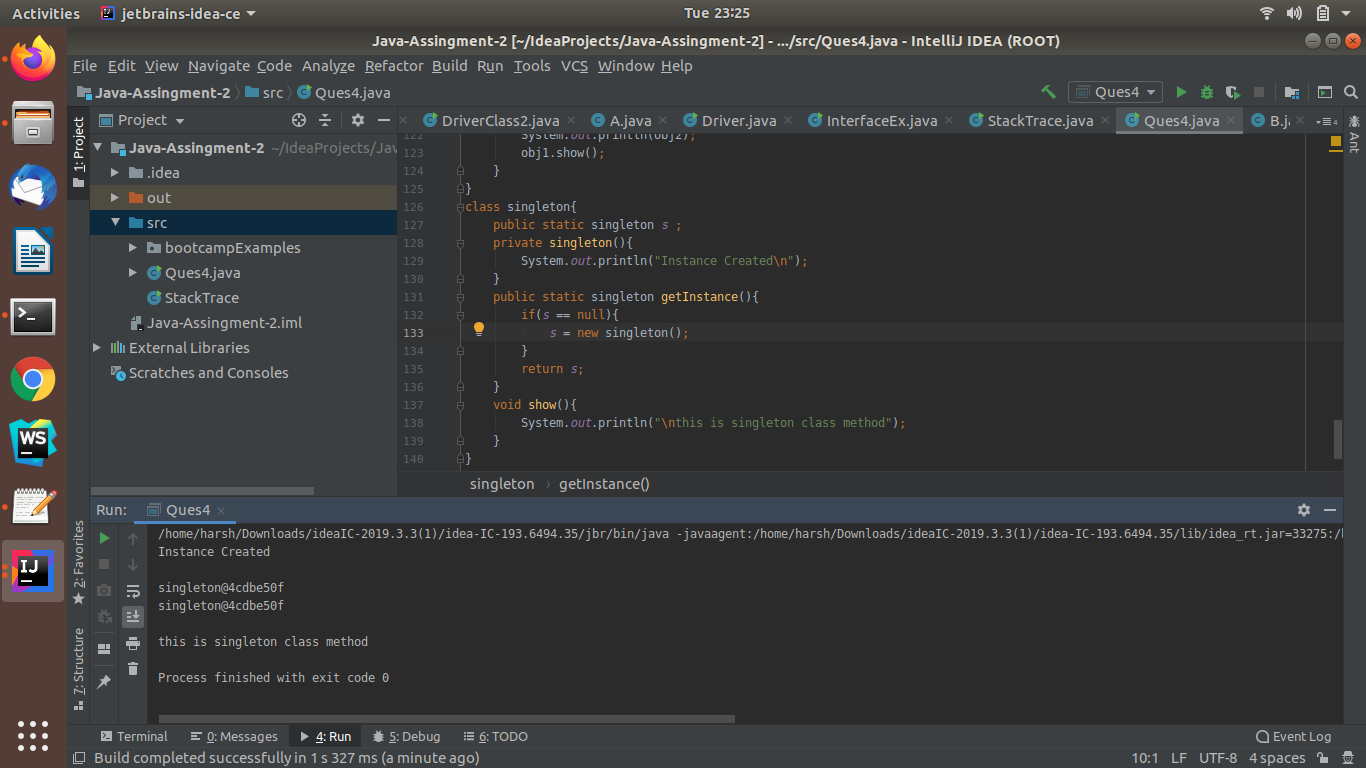
}

void show(){

System.out.println("\nthis is singleton class method");

}

}



Ques 5: WAP to show object cloning in java using cloneable and copy constructor both.

Ans: **JAVA-CODE**

class Ques5{

int i,j;

public Ques5(int a, int b){

this.i=a;

this.j=b;

}

Ques5(Ques5 q){

System.out.println("\nCopy constructor called\n");

i=q.i;

j=q.j;

}

@Override

public String toString(){

return "(" + i + " + " + j + "i)";

}

}

class Driver{

public static void main(String[] args) {

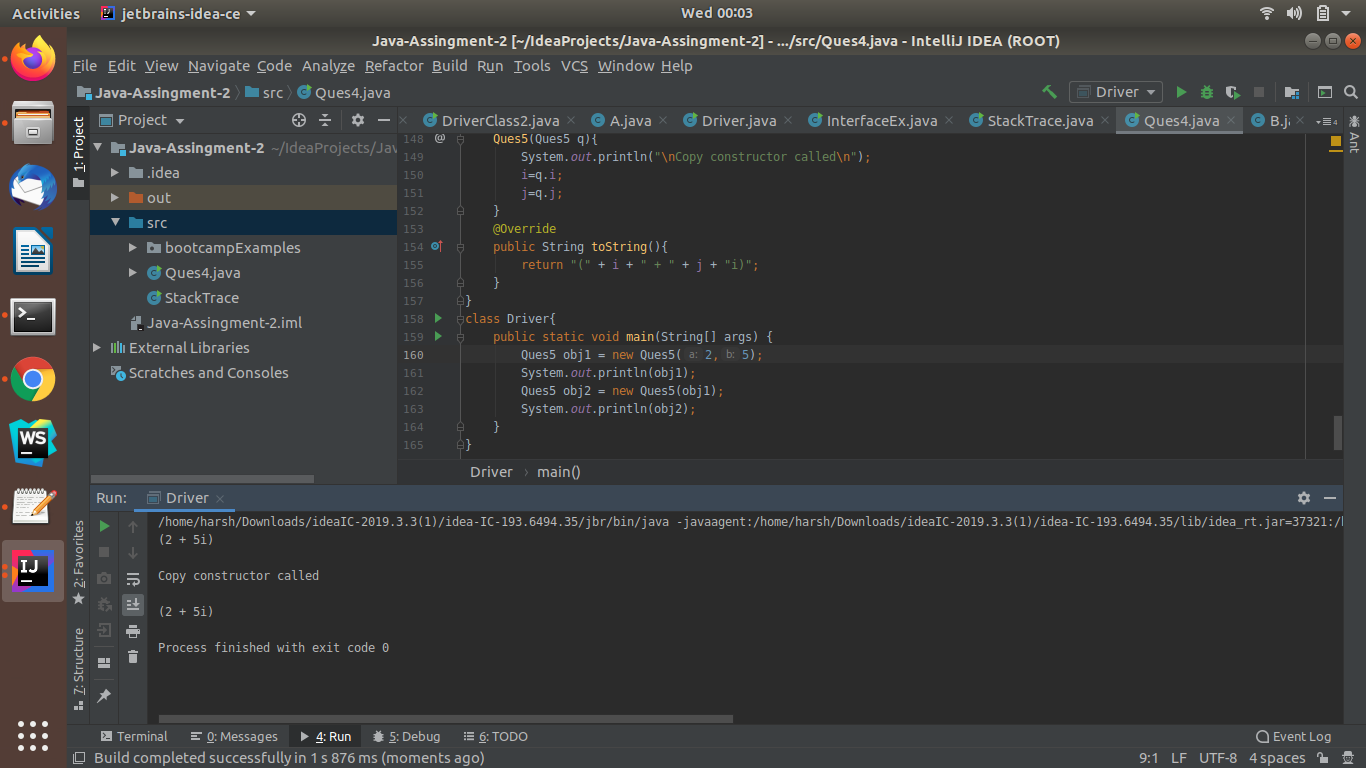
Ques5 obj1 = new Ques5(2,5);

System.out.println(obj1);

Ques5 obj2 = new Ques5(obj1);

System.out.println(obj2);

}

}

//using cloneable interface method clone()

class Ques5 {

int i, j;

}

class Driver implements Cloneable {

int a,b;

Ques5 q1 = new Ques5();

public Object clone() throws CloneNotSupportedException{

Driver d = (Driver)super.clone();

d.q1 = new Ques5();

return d;

}

public static void main(String[] args) throws CloneNotSupportedException {

Driver d1 = new Driver();

d1.a=100;

d1.b=200;

d1.q1.i=300;

d1.q1.j=400;

Driver d2 =(Driver)d1.clone();

d2.a=500;

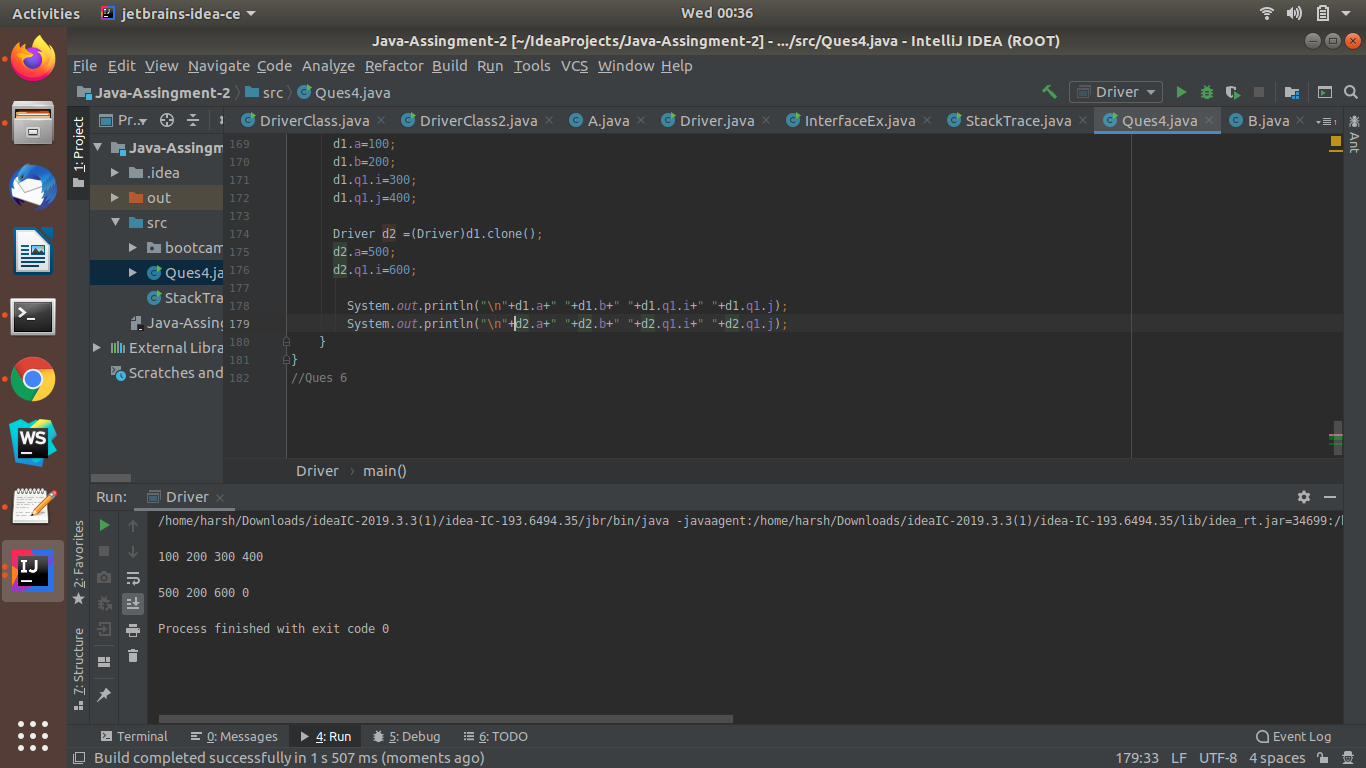
d2.q1.i=600;

System.out.println(d1.a+" "+d1.b+" "+d1.q1.i+" "+d1.q1.j);

System.out.println(d2.a+" "+d2.b+" "+d2.q1.i+" "+d2.q1.j);

}

}



Ques 6:WAP showing try, multi-catch and finally blocks.

Ans: **JAVA-CODE**

class mutlicatch{

public static void main(String[] args) {

int[] arr =new int[5];

try{

int division = 98/0;

int result = arr[7];

}

catch (ArrayIndexOutOfBoundsException e){

System.out.println("\nArrayIndexOutOfBoundsException:");

}

catch(ArithmeticException e){

System.out.println("\nDivide by zero exception :");

}

catch(Exception e){

System.out.println("\nAnykind of Exception:");

}

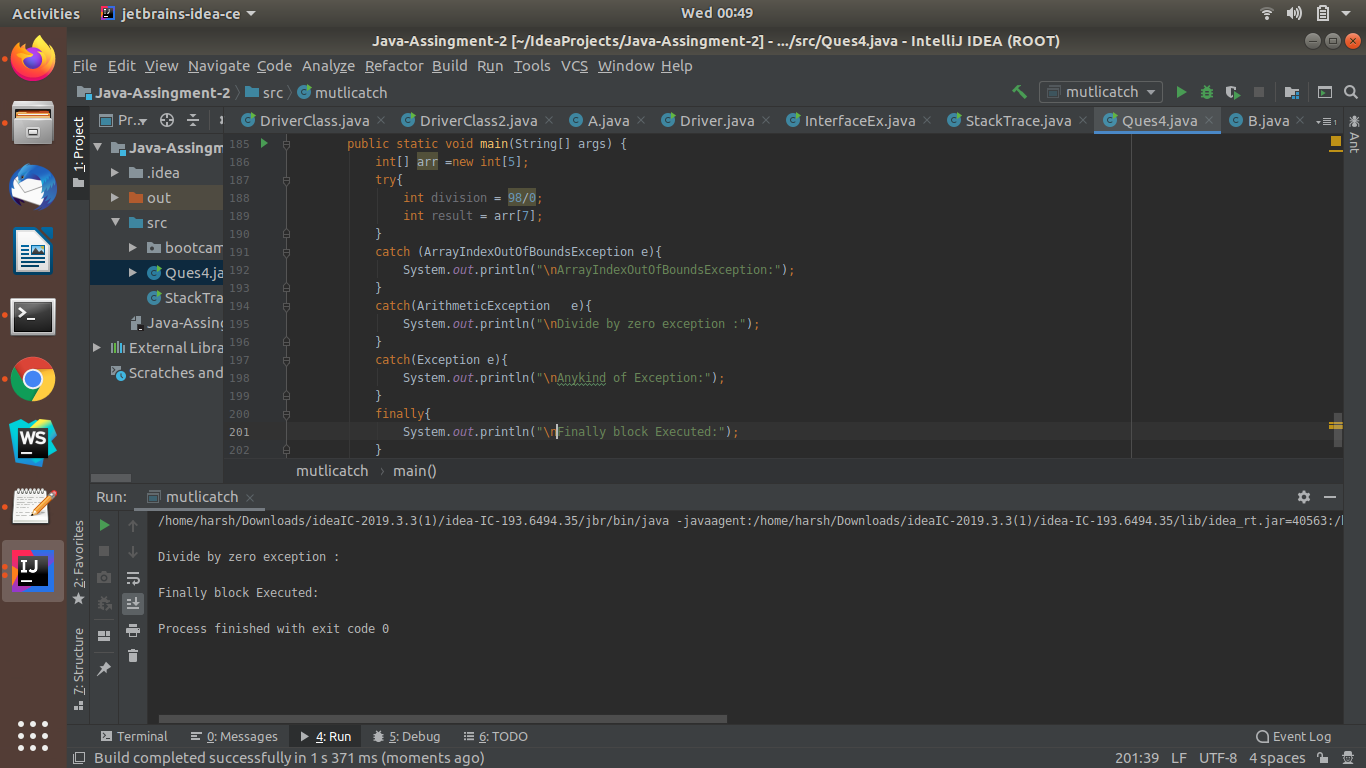
finally{

System.out.println("\nFinally block Executed:");

}

}

}



Ques 7:WAP to convert seconds into days, hours, minutes and seconds.

Ans:**JAVA-CODE**

class SecondConversion{

void secondconversion(int seconds) {

int days = seconds/(24\*3600);

seconds=seconds%(24\*3600);

int hour = seconds/(3600);

seconds =seconds/3600;

int minutes = seconds/60;

seconds = seconds/60;

System.out.println("\ndays: "+days+" hours: "+hour+" minutes: "+minutes+" seconds: "+seconds);

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("\nEnter seconds to find days hours minutes and seconds");

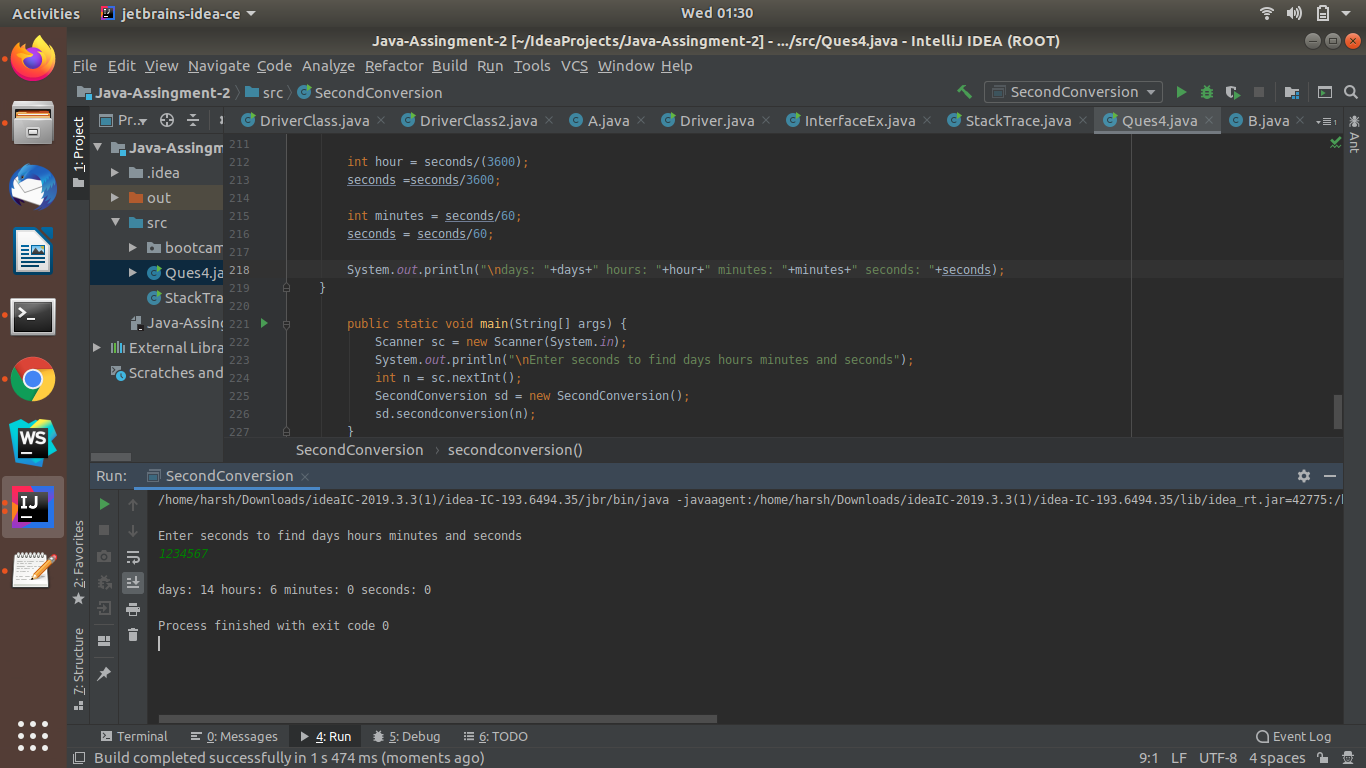
int n = sc.nextInt();

SecondConversion sd = new SecondConversion();

sd.secondconversion(n);

}

}



Ques 8:WAP to read words from the keyboard until the word done is entered. For each word except done, report whether its first character is equal to its last character. For the required loop, use a

a)while statement

b)do-while statement

Ans:**JAVA-CODE**

class Ques8{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String str="";

while(true){

str = sc.next();

if(str.equals("done")){

System.out.println("\nentered Done");

break;

}

else{

if(str.charAt(0) == str.charAt(str.length()-1)){

System.out.println("first character and last character are same: ");

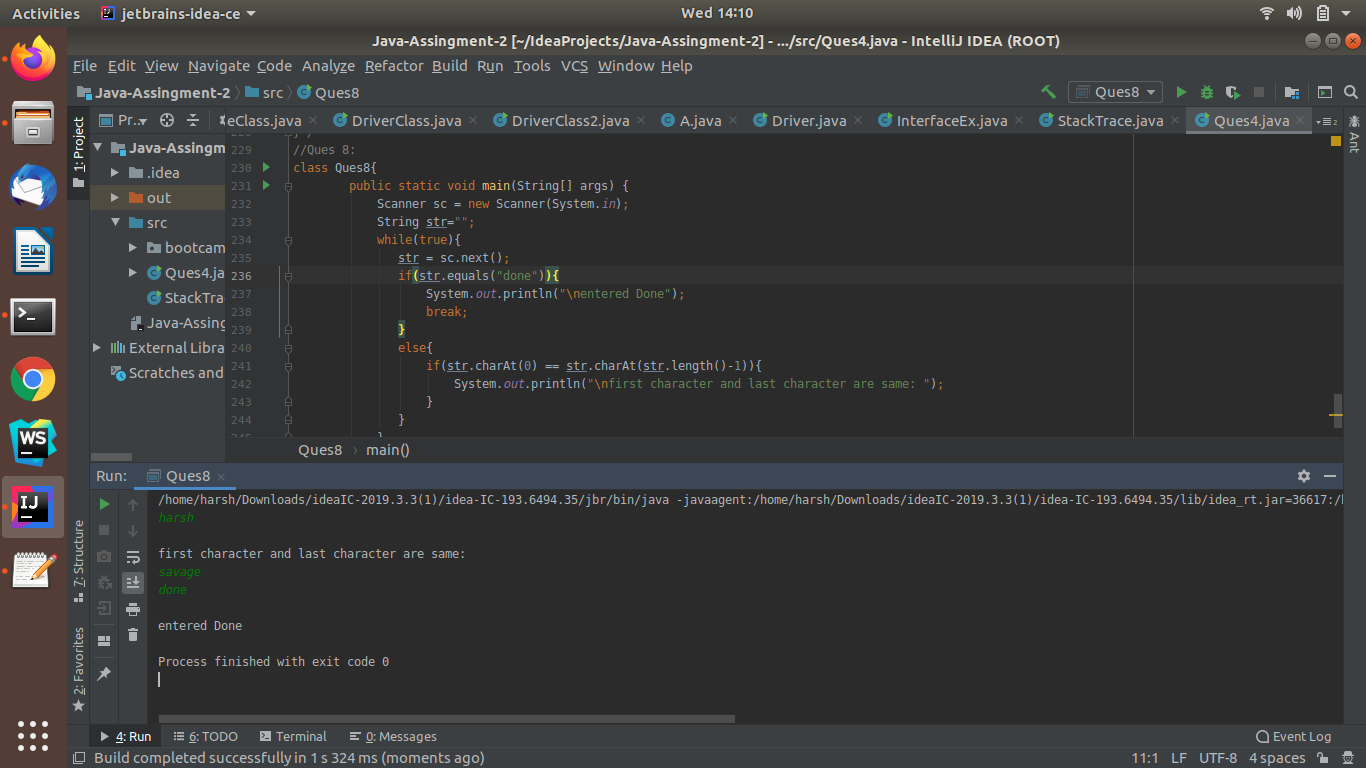
}

}

}

}

}



Ques 9: Design classes having attributes for furniture where there are wooden chairs and tables, metal chairs and tables. There are stress and fire tests for each products.

Ans: **JAVA-CODE**

abstract class Furniture{

abstract void stress();

abstract void fire();

abstract void showdetails();

}

class WoodenChairs extends Furniture {

String color ="Brown";

char size='S';

public void stress(){

System.out.println("Stress");

System.out.println(color);

}

public void fire(){

System.out.println("Fire");

}

public void showdetails(){

System.out.println("color of chair: "+color+" size of chair"+size);

}

}

class Tables extends Furniture{

String color ="Golden Brown";

char size='L';

public void stress(){

System.out.println("Stress");

}

public void fire(){

System.out.println("Fire");

}

public void showdetails(){

System.out.println("color of chair: "+color+" size of chair"+size);

}

}

class MetalChairs extends Furniture{

String color ="Metal Brown";

char size='M';

public void stress(){

System.out.println("Stress");

}

public void fire(){

System.out.println("Fire");

}

public void showdetails(){

System.out.println("color of chair: "+color+" size of chair"+size);

}

}

class Driver {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter type of furniture ie. WoodenChairs, Tables, MetalChairs");

String type = sc.next();

Furniture d = null;

if(type.equals("WoodenChairs")){

d = new WoodenChairs();

}

else if(type.equals("Tables"))

{

d = new Tables();

}

else if(type.equals("MetalChairs"))

{

d = new MetalChairs();

}

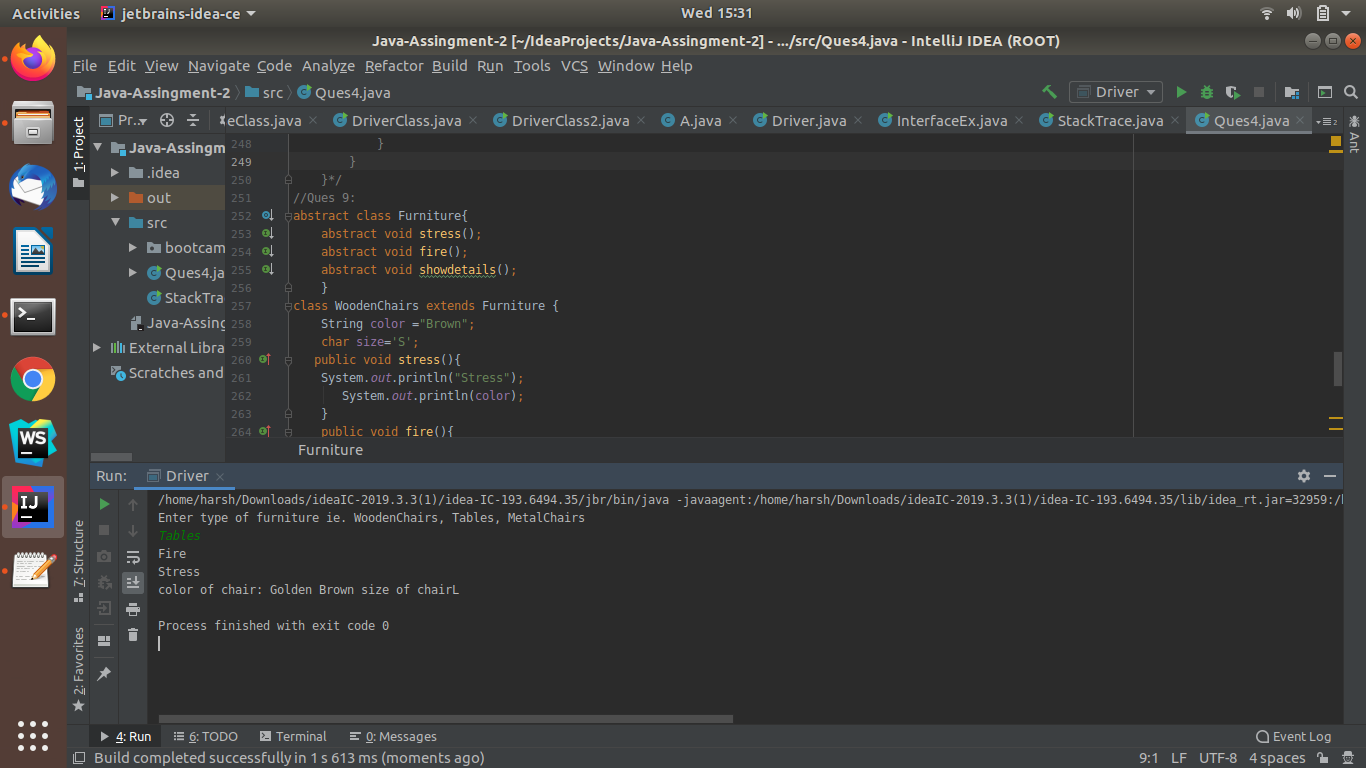
d.fire();

d.stress();

d.showdetails();

}

}



Ques 10:Design classes having attributes and method(only skeleton) for a coffee shop. There are three different actors in our scenario and i have listed the different actions they do also below

\* Customer

- Pays the cash to the cashier and places his order, get a token number back

- Waits for the intimation that order for his token is ready

- Upon intimation/notification he collects the coffee and enjoys his drink

( Assumption: Customer waits till the coffee is done, he wont timeout and cancel the order. Customer always likes the drink served. Exceptions like he not liking his coffee, he getting wrong coffee are not considered to keep the design simple.)

\* Cashier

- Takes an order and payment from the customer

- Upon payment, creates an order and places it into the order queue

- Intimates the customer that he has to wait for his token and gives him his token

( Assumption: Token returned to the customer is the order id. Order queue is unlimited. With a simple modification, we can design for a limited queue size)

\* Barista

- Gets the next order from the queue

- Prepares the coffee

- Places the coffee in the completed order queue

- Places a notification that order for token is ready

Ans:**JAVA-CODE**

package Ques10;

import java.util.\*;

public class Orders { **//this class is for maintaining order details**

HashMap<String,Double> order = new HashMap<String, Double>();

List<String> completedorder = new ArrayList<String>();

Queue<String> queue = new PriorityQueue<String>();

void addorder(String orderid,double amount){

order.put(orderid,amount);

System.out.println("order is added");

}

void removeorder(String orderid){

order.remove(orderid);

System.out.println("Order is removed");

}

void completeorder(String orderid){

completedorder.add(orderid);

System.out.println("order is completed");

}

}

package Ques10;

public class Customer {  **// this class is used to maintain customer details**

private String name;

private String token;

Cashier cashier;

double amount;

String orderid;

String customername

void placeOrder() {

token = cashier.AcceptOrderAndAddCustomerToCustomerList(String customername,String orderid, double amount);

System.out.println("This is the order token: " + token);

}

boolean waitingState(){

System.out.println("Customer" + this.name + "is waiting");

return true;

}

boolean drinkingState() {

System.out.println("Customer " + this.name + " has collected coffee");

return true;

}

}

package Ques10; **// this class is used to maintain cashier activity**

import java.util.ArrayList;

import java.util.List;

import java.util.PriorityQueue;

import java.util.Queue;

public class Cashier extends Orders {

String cashiername;

List<String> customerList = new ArrayList<String>();

String AcceptOrderAndAddCustomerToCustomerList(String customername,String orderid,double amount){

addorder(orderid,amount);

customerList.add(customername);

System.out.println("Order accepted");

return "token";

}

void addordertoorderqueue(String orderid){

queue.add(orderid);

}

}

package Ques10; **//this class is used to maintain Barista activity**

public class Barista extends Orders{

String Baristaname;

String ordertobeprepared;

Cashier cash= new Cashier();

void Cofeeprepared(){

completeorder("Coffee");

}

void orderidReady(String token){

System.out.println("order for"+token+"is ready");

}

void getnextorderfromorderqueue(){

ordertobeprepared=queue.remove();

}

}

public class main class{ //to perforn different operations

public static void main(String[] args){

}

}

Ques 11:Convert the following code so that it uses nested while statements instead of for statements:

int s = 0;

int t = 1;

for (int i = 0; i < 10; i++)

{

s = s + i;

for (int j = i; j > 0; j−−)

{

t = t \* (j - i);

}

s = s \* t;

System.out.println("T is " + t);

}

System.out.println("S is " + s);

Ans:**JAVA-CODE**

class Driver{

public static void main(String[] args) {

int s = 0;

int t = 1;

int i=0,j=0;

//for (int i = 0; i < 10; i++)

while(i <10)

{

j=i;

s = s + i;

//for (int j = i; j > 0; j−−)

while( j > 0)

{

t = t \* (j - i);

j--;

}

s = s \* t;

System.out.println("T is " + t);

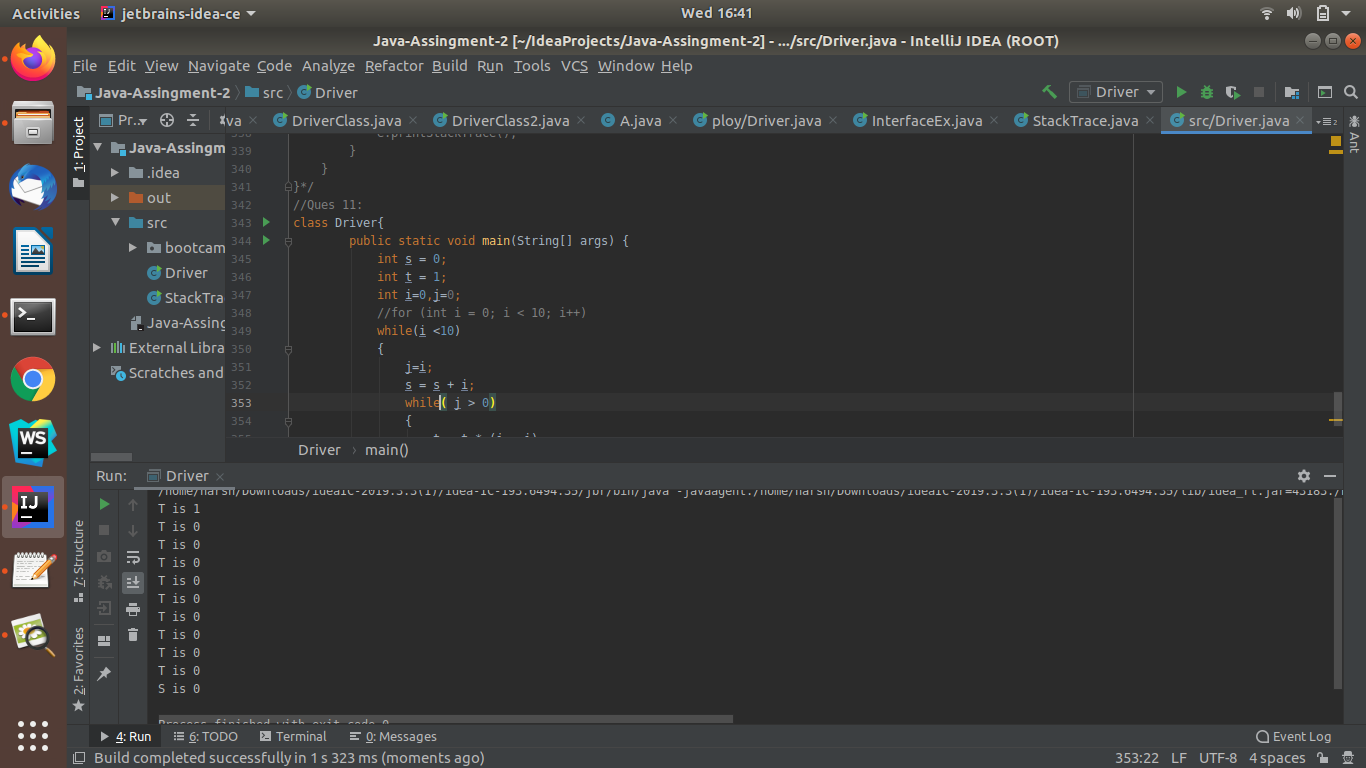
i++;

}

System.out.println("S is " + s);

}

}



Ques 12:What will be the output on new Child(); ?

class Parent extends Grandparent {

{

System.out.println("instance - parent");

}

public Parent() {

System.out.println("constructor - parent");

}

static {

System.out.println("static - parent");

}

}

class Grandparent {

static {

System.out.println("static - grandparent");

}

{

System.out.println("instance - grandparent");

}

public Grandparent() {

System.out.println("constructor - grandparent");

}

}

class Child extends Parent {

public Child() {

System.out.println("constructor - child");

}

static {

System.out.println("static - child");

}

{

System.out.println("instance - child");

}

}

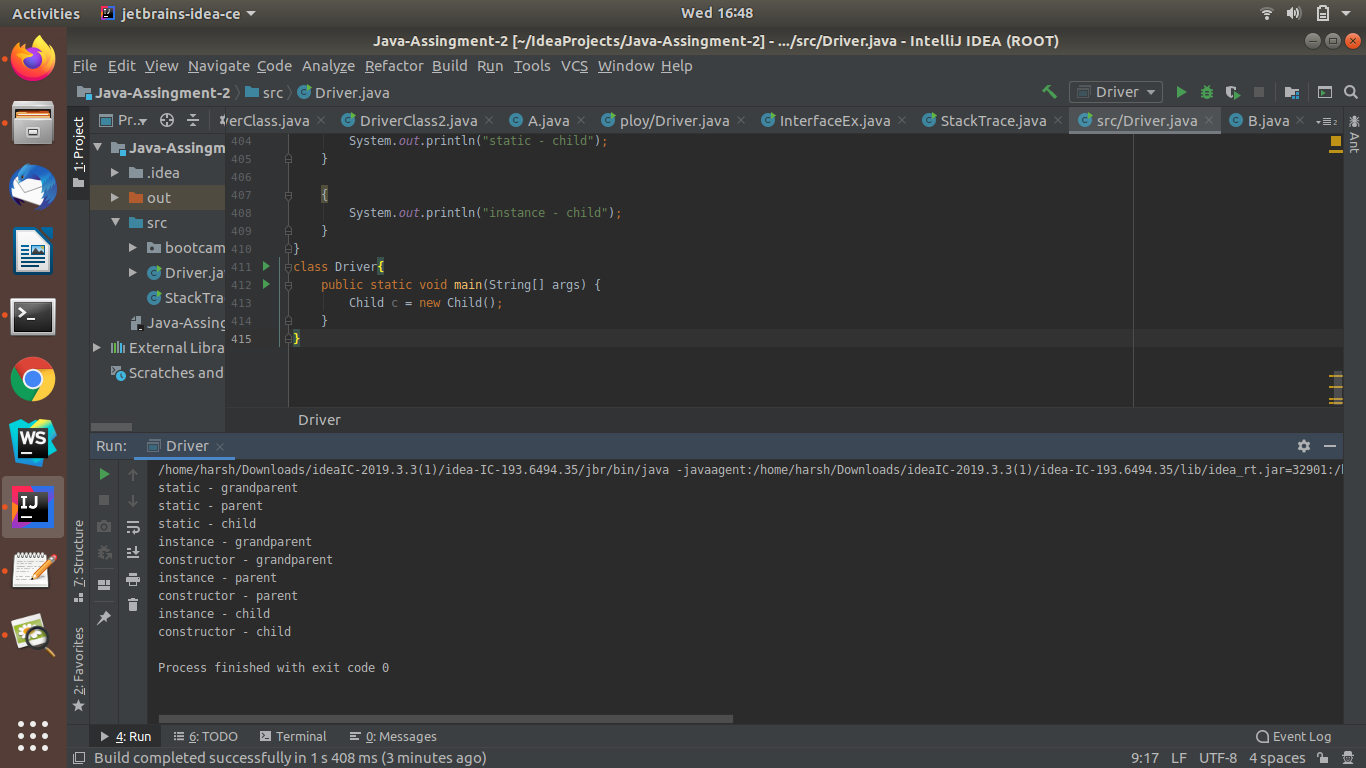
class Driver{

public static void main(String[] args) {

Child c = new Child();

}

}

Output:

Ques 13:Create a custom exception that do not have any stack trace.

Ans:**JAVA-CODE**

class CustomException extends Exception{

CustomException(String msg){

super(msg,null,true,false);

}

}

public class Driver{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter value greater than 1000:");

int num = sc.nextInt();

try{

if(num < 1000){

throw new CustomException("Exception entered number is less than 1000");

}

}

catch(CustomException e){

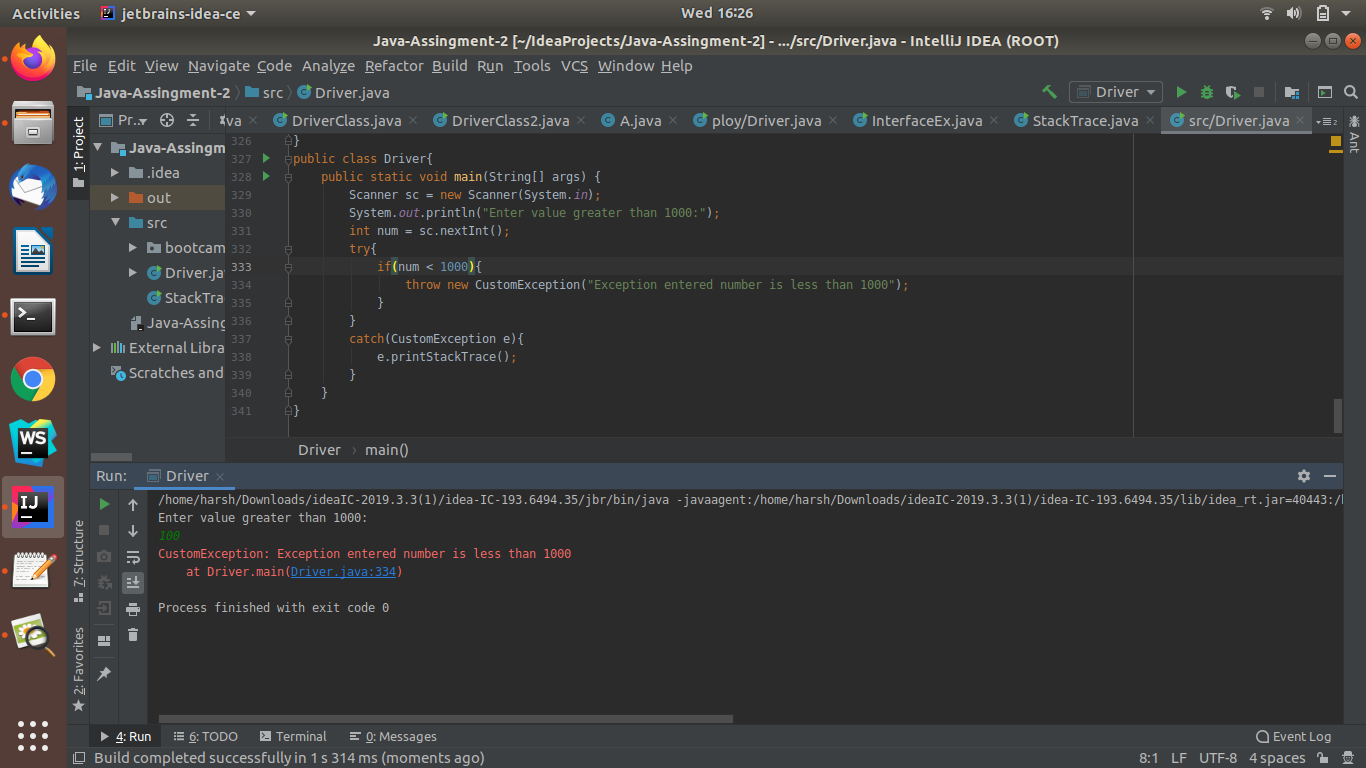
e.printStackTrace();

}

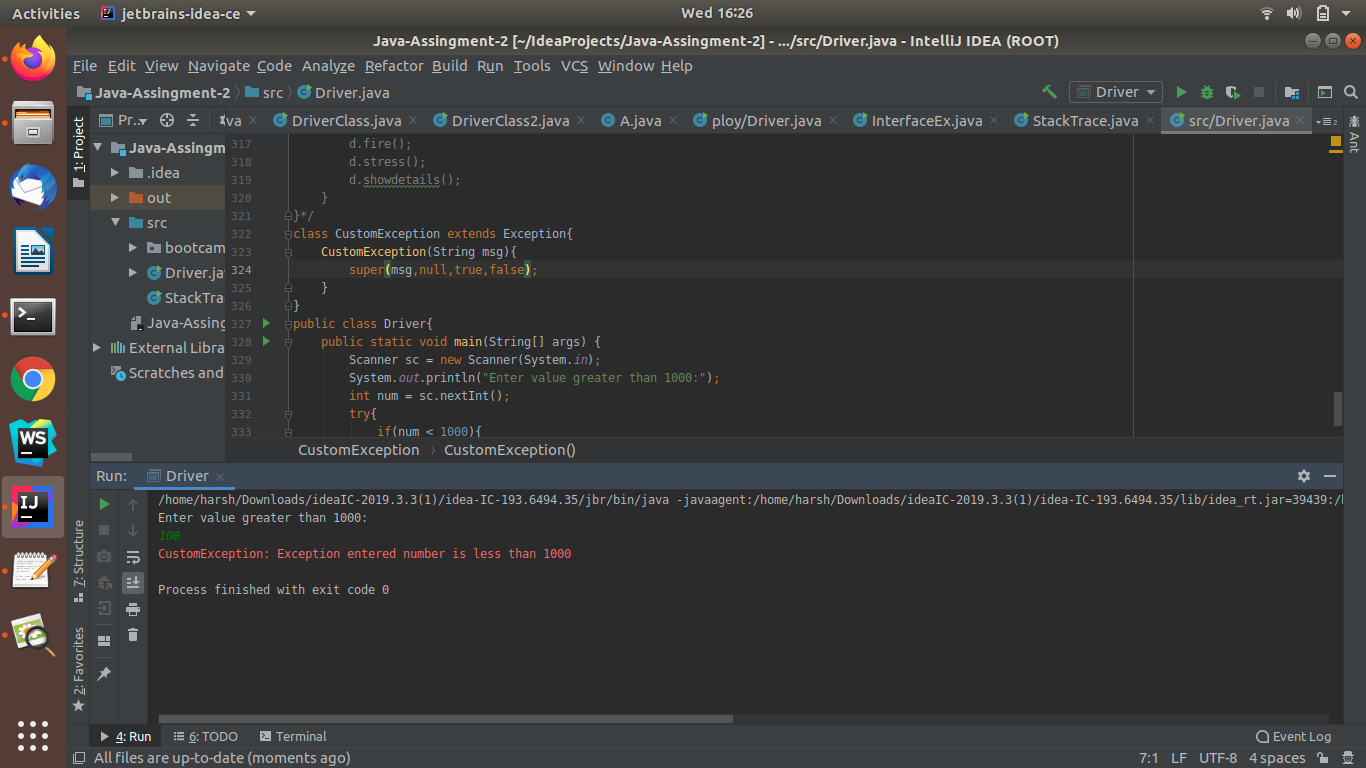
}

}

/**/StackTrace shown**



**//StackTrace NotShown**

****