**Module 3**

**Question 1:**

Write a program to read file, encrypt it (encryption key is +3), and store encrypted data to another file.(Use FileInputStream/OutputStream).

**Code**

import java.io.\*;

class Qu1 {

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

        FileInputStream fin = new FileInputStream("Qu1.java");

        FileOutputStream fout = new FileOutputStream("Qu1.txt");

        int i;

        while ((i = fin.read()) != -1) {

            fout.write(i + 3);

        }

        fin.close();

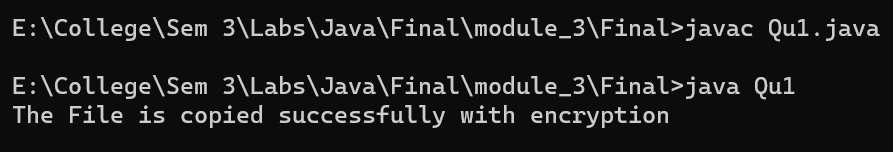
        fout.close();

        System.out.println("The File is copied successfully with encryption");

    }

}

**Output**

****

**Qu1.txt**

lpsruw#mdyd1lr1->

fodvv#Tx4#~

####sxeolf#vwdwlf#yrlg#pdlq+Vwulqj#dujv^`,#wkurzv#LRH{fhswlrq#~

########IlohLqsxwVwuhdp#ilq#@#qhz#IlohLqsxwVwuhdp+%Tx41mdyd%,>

########IlohRxwsxwVwuhdp#irxw#@#qhz#IlohRxwsxwVwuhdp+%Tx41w{w%,>

########lqw#l>

########zkloh#++l#@#ilq1uhdg+,,#$@#04,#~

############irxw1zulwh+l#.#6,>

########�

########ilq1forvh+,>

########irxw1forvh+,>

########V|vwhp1rxw1sulqwoq+%Wkh#Iloh#lv#frslhg#vxffhvvixoo|#zlwk#hqfu|swlrq%,>

####�

�

**Question 2:**

Write a program, which reads encrypted file generated in 1st program, decrypt it, print it to the console.(Use FileInputStream).

**Code**

import java.io.\*;

public class Qu2 {

    public static void main(String[] args) {

        try {

            FileInputStream fin = new FileInputStream("Qu1.txt");

            int i;

            while ((i = fin.read()) != -1) {

                System.out.print((char) (i - 3));

            }

            fin.close();

        } catch (Exception e) {

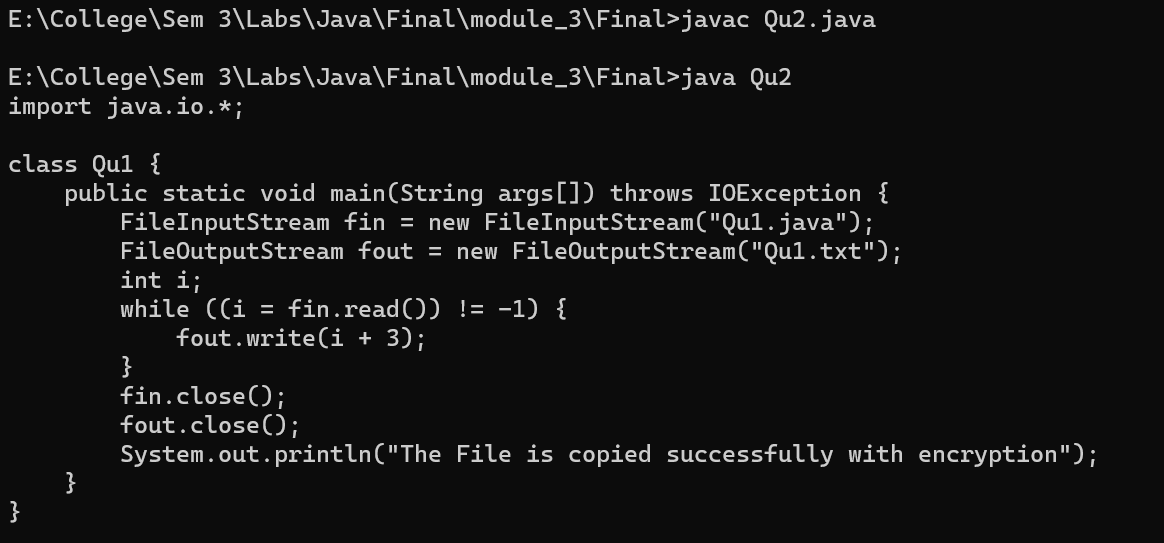
            System.out.println(e);

        }

    }

}

**Output**

****

**Question 3:**

Write a program to Copy the data of a file, file name provided from command line argument, to another file.(Use Buffered I/O Stream).

**Code**

import java.io.\*;

public class Qu3 {

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

        FileInputStream fin = new FileInputStream(args[0]);

        FileOutputStream fos = new FileOutputStream(args[1]);

        BufferedInputStream buffin = new BufferedInputStream(fin);

        BufferedOutputStream buffout = new BufferedOutputStream(fos);

        int i;

        while ((i = buffin.read()) != -1) {

            buffout.write(i);

        }

        buffin.close();

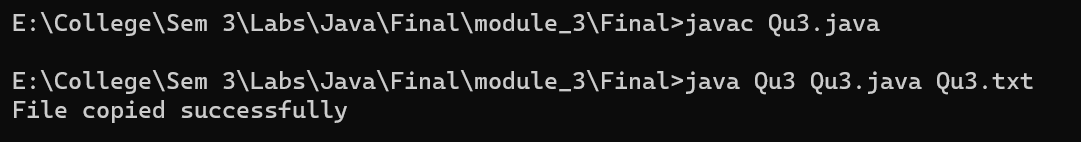
        buffout.close();

        System.out.println("File copied successfully");

    }

}

**Output**

****

**Qu3.txt**

import java.io.\*;

public class Qu3 {

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

FileInputStream fin = new FileInputStream(args[0]);

FileOutputStream fos = new FileOutputStream(args[1]);

BufferedInputStream buffin = new BufferedInputStream(fin);

BufferedOutputStream buffout = new BufferedOutputStream(fos);

int i;

while ((i = buffin.read()) != -1) {

buffout.write(i);

}

buffin.close();

buffout.close();

System.out.println("File copied successfully");

}

}

**Question 4:**

Write a ProductManagement program, which will store 5 objects of Product class to a file name “ProductDetails.dat”.(Use ObjectOutputStream)

**Code**

import java.io.\*;

class Product implements Serializable

{

    int pid;

    String pname;

    double price;

    Product(int pid,String pname,double price)

    {

        this.pid=pid;

        this.pname=pname;

        this.price=price;

    }

    public String toString()

    {

        return pid+" "+pname+" "+price;

    }

}

public class Qu4

{

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

    {

        Product p1=new Product(1,"Apple",100);

        Product p2=new Product(2,"Samsung",200);

        Product p3=new Product(3,"Vivo",3000);

        Product p4=new Product(4,"Oppo",400);

        Product p5=new Product(5,"Nothing",500);

        FileOutputStream fout=new FileOutputStream("ProductDetails.dat");

        ObjectOutputStream out=new ObjectOutputStream(fout);

        out.writeObject(p1);

        out.writeObject(p2);

        out.writeObject(p3);

        out.writeObject(p4);

        out.writeObject(p5);

        out.close();

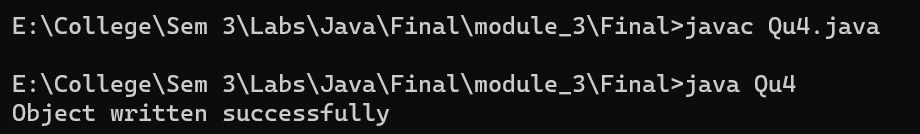
        fout.close();

        System.out.println("Object written successfully");

    }

}

**Output**

****

**Question 5:**

Write a program, Which will read file “ProductDetails.dat” (created by above program).Print the details of the object having highest price and lowest price. (Use ObjectInputStream).

**Code**

import java.io.\*;

public class Qu5 {

    public static void main(String[] args) {

        try {

            FileInputStream fin = new FileInputStream("ProductDetails.dat");

            ObjectInputStream in = new ObjectInputStream(fin);

            Product p1 = (Product) in.readObject();

            Product p2 = (Product) in.readObject();

            Product p3 = (Product) in.readObject();

            Product p4 = (Product) in.readObject();

            Product p5 = (Product) in.readObject();

            Product[] p = {p1, p2, p3, p4, p5};

            double max = p[0].price;

            double min = p[0].price;

            for (int i = 1; i < p.length; i++) {

                if (p[i].price > max) {

                    max = p[i].price;

                }

                if (p[i].price < min) {

                    min = p[i].price;

                }

            }

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

                if (p[i].price == max) {

                    System.out.println("Highest Price: " + p[i]);

                }

                if (p[i].price == min) {

                    System.out.println("Lowest Price: " + p[i]);

                }

            }

            in.close();

            fin.close();

        } catch (Exception e) {

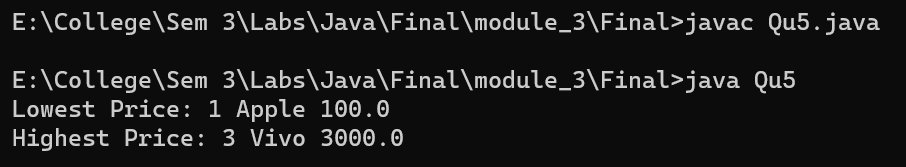
            System.out.println(e);

        }

    }

}

**Output**

****

**Question 6:**

Write a program which will read a text file and print the count of total number of Lines, Words and Characters in it. (Use BufferedReader).

**Code**

import java.io.\*;

public class Qu6 {

    public static void main(String[] args) {

        try {

            BufferedReader br = new BufferedReader(new FileReader("Qu6.java"));

            int lines = 0, words = 0, chars = 0;

            String s;

            while ((s = br.readLine()) != null) {

                lines++;

                String[] w = s.split(" ");

                words += w.length;

                chars += s.length();

            }

            System.out.println("Lines: " + lines);

            System.out.println("Words: " + words);

            System.out.println("Characters: " + chars);

            br.close();

        } catch (Exception e) {

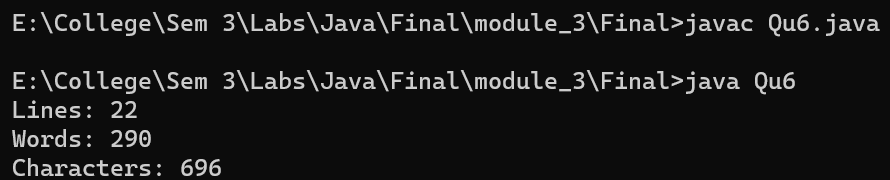
            System.out.println(e);

        }

    }

}

**Output**

****

**Question 7:**

Write a program to read a text file and copy it’s content in uppercase form to another file.(Use BufferedWriter\Reader).

**Code**

import java.io.\*;

public class Qu7 {

    public static void main(String[] args) {

        try {

            BufferedReader br = new BufferedReader(new FileReader("Qu7.java"));

            BufferedWriter bw = new BufferedWriter(new FileWriter("Qu7.txt"));

            String s;

            while ((s = br.readLine()) != null) {

                bw.write(s.toUpperCase());

                bw.newLine();

            }

            br.close();

            bw.close();

        } catch (Exception e) {

            System.out.println(e);

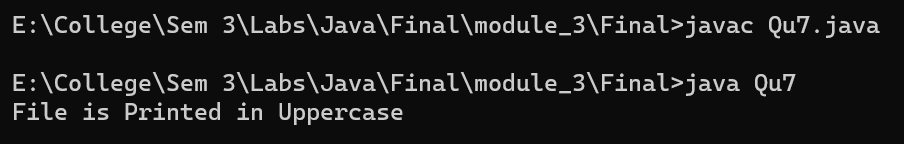
        }

        System.out.println("File is Printed in Uppercase");

    }

}

**Output**

****

**Qu7.txt**

IMPORT JAVA.IO.\*;

PUBLIC CLASS QU7 {

PUBLIC STATIC VOID MAIN(STRING[] ARGS) {

TRY {

BUFFEREDREADER BR = NEW BUFFEREDREADER(NEW FILEREADER("QU7.JAVA"));

BUFFEREDWRITER BW = NEW BUFFEREDWRITER(NEW FILEWRITER("QU7.TXT"));

STRING S;

WHILE ((S = BR.READLINE()) != NULL) {

BW.WRITE(S.TOUPPERCASE());

BW.NEWLINE();

}

BR.CLOSE();

BW.CLOSE();

} CATCH (EXCEPTION E) {

SYSTEM.OUT.PRINTLN(E);

}

SYSTEM.OUT.PRINTLN("FILE IS PRINTED IN UPPERCASE");

}

}

**Question 8:**

Write a program to copy content of a file while removing duplicates lines.

**Code**

import java.io.\*;

import java.util.HashSet;

public class Qu8 {

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

        // PrintWriter pw = new PrintWriter("Qu81.txt");

        FileOutputStream fos = new FileOutputStream("Qu8-1.txt");

        BufferedOutputStream buffout = new BufferedOutputStream(fos);

        BufferedReader br = new BufferedReader(new FileReader("Qu8.txt"));

        String line = br.readLine();

        HashSet<String> hs = new HashSet<String>();

        while (line != null) {

            hs.add(line);

            line = br.readLine();

        }

        for (String s : hs) {

            buffout.write(s.getBytes());

            buffout.write("\n".getBytes());

        }

        br.close();

        buffout.close();

        System.out.println("File operation performed successfully");

    }

}

**Qu8.txt**

this is line 1

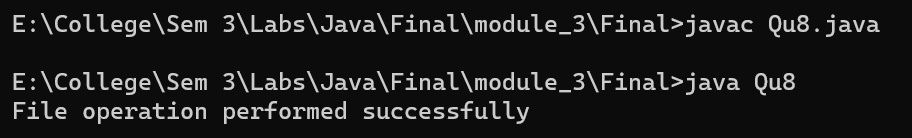
this is line 1

this is line 2

this is line 3

this is line 3

**Output**

****

**Qu8-1.txt**

this is line 3

this is line 2

this is line 1

**Question 9:**

Write an Exception handling program, which will handle RuntimeException, ArrayIndexOutOfBoundsException, NumberFormatException, ArithmeticException, NullPointerException.(Use Multiple catch with single try block).

**Code**

public class Qu9 {

    public static void main(String[] args) {

        try {

            int a = Integer.parseInt(args[0]);

            int b = Integer.parseInt(args[1]);

            int c = a / b;

            System.out.println(c);

        } catch (ArithmeticException e) {

            System.out.println("ArithmeticException: " + e);

        } catch (ArrayIndexOutOfBoundsException e) {

            System.out.println("ArrayIndexOutOfBoundsException: " + e);

        } catch (NumberFormatException e) {

            System.out.println("NumberFormatException: " + e);

        } catch (NullPointerException e) {

            System.out.println("NullPointerException: " + e);

        } catch (RuntimeException e) {

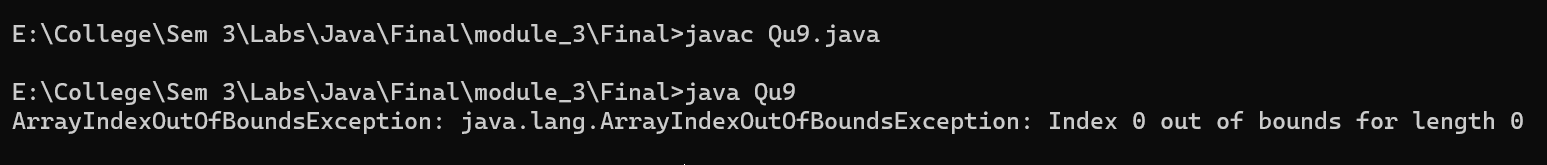
            System.out.println("RuntimeException: " + e);

        }

    }

}

**Output**

****

**Question 10:**

Write a program, to demonstrate nested try-catch-finally structure.

**Code**

//Write a program, to demonstrate nested try-catch-finally structure.

import java.io.\*;

public class Qu10 {

    public static void main(String[] args) {

        try {

            try {

                int a = 10 / 0;

            } catch (ArithmeticException e) {

                System.out.println("Arithmetic Exception");

            }

            try {

                int[] a = new int[5];

                a[10] = 10;

            } catch (ArrayIndexOutOfBoundsException e) {

                System.out.println("ArrayIndexOutOfBounds Exception");

            }

        } catch (Exception e) {

            System.out.println("Parent Exception");

        } finally {

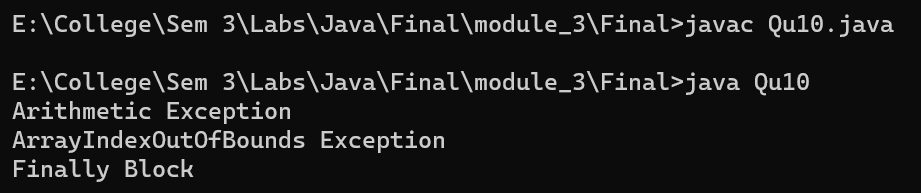
            System.out.println("Finally Block");

        }

    }

}

**Output**

****

**Question 11:**

Write a program, to create and handle user defined Unchecked Exception – InvalidBoxException which will be thrown from the constructor of the Box class, when either length or width or height of Box is less than zero.

**Code**

class InvalidBoxException extends Exception {

    InvalidBoxException(String s) {

        super(s);

    }

}

class Box {

    int length;

    int width;

    int height;

    Box(int l, int w, int h) throws InvalidBoxException {

        if (l < 0 || w < 0 || h < 0) {

            throw new InvalidBoxException("invaild input");

        } else {

            System.out.println("Done");

        }

    }

}

public class Qu11 {

    public static void main(String[] args) {

        try {

            Box b = new Box(1, -22, 4);

        } catch (InvalidBoxException e) {

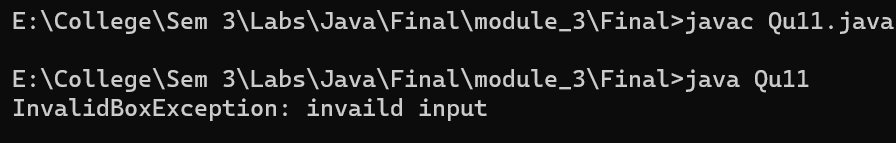
            System.out.println(e);

        }

    }

}

**Output**

****

**Question 12:**

Write a program to create and handle User Defined CheckedException – InsufficientFundsException, generated while withdrawing amount more than available balance.Create necessary class and methods to support this scenario.

**Code**

class InsufficientFunds extends Throwable {

    InsufficientFunds(String s) {

        super(s);

    }

}

class IllegalTransfer extends Throwable {

    IllegalTransfer(String s) {

        super(s);

    }

}

class BankAccount {

    String name;

    int balance;

    BankAccount(String s, int b) {

        name = s;

        balance = b;

    }

    void withdraw(int a) {

        try {

            if (a > balance) {

                throw new InsufficientFunds("insufficient balance in the account");

            } else {

                balance = balance - a;

                System.out.println("Withdraw Done");

            }

        } catch (InsufficientFunds e) {

            System.out.println(e);

        }

    }

    void deposit(int a) {

    }

}

public class Qu12 {

    public static void main(String[] args) {

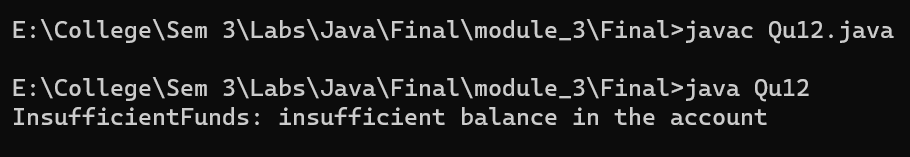
        BankAccount a1 = new BankAccount("abc", 100);

        a1.withdraw(1000);

    }

}

**Output**

****

**Question 13:**

Write a program to create an ArrayList of Products. Traverse the list and print it to the console. Provide a searching of product on name basis using contains() method of List.

**Code**

import java.util.Scanner;

import java.util.ArrayList;

public class Qu13 {

    public static void main(String[] args) {

        ArrayList<String> products = new ArrayList<>();

        products.add("samsung");

        products.add("apple");

        products.add("mi");

        products.add("vivo");

        products.add("oneplus");

        System.out.println(products + " ");

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter the product to search in ArrayList");

        String name = sc.nextLine();

        if (products.contains(name)) {

            System.out.println("Product found");

        } else {

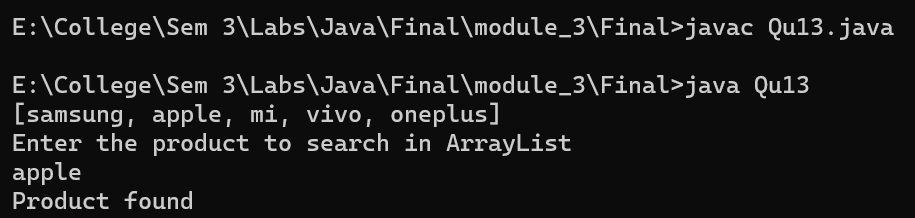
            System.out.println("Product not found");

        }

    }

}

**Output**

****

**Question 14:**

Write a program, to create a TreeSet of Products.Traverse it and provide ordering on base of id.

**Code**

import java.util.\*;

class Product {

    String name;

    int Id;

    Product(int i, String name) {

        this.Id = i;

        this.name = name;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public int getId() {

        return Id;

    }

}

class idComparator implements Comparator<Product> {

    public int compare(Product p1, Product p2) {

        return p1.getId() - p2.getId();

    }

}

public class Qu14 {

    public static void main(String[] args) {

        TreeSet<Product> set = new TreeSet<Product>(new idComparator());

        set.add(new Product(1000, "samsung"));

        set.add(new Product(02, "apple"));

        set.add(new Product(300, "oneplus"));

        set.add(new Product(4, "vivo"));

        set.add(new Product(555, "mi"));

        for (Product p : set) {

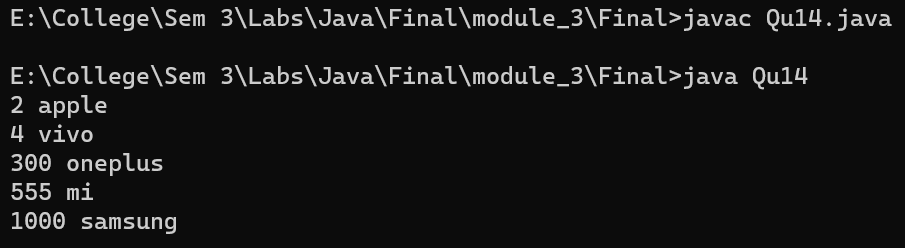
            System.out.println(p.Id + " " + p.name);

        }

    }

}

**Output**

****

**Question 15:**

Write a program to create a TreeSet of Products. Traverse it and provide ordering on basis of name(Use comparator interface).

**Code**

//Write a program to create a TreeSet of Products. Traverse it and provide ordering on basis of name(Use comparator interface).

import java.util.\*;

class Product{

    String name;

    int Id;

    Product(int i,String name){

        this.Id=i;

        this.name=name;

    }

    public String getName(){

        return name;

    }

    public void setName(String name){

        this.name=name;

    }

    public int getId(){

        return Id;

    }

}

class nameComparator implements Comparator<Product>{

    public int compare(Product p1,Product p2){

        return p1.getName().compareTo(p2.getName());

    }

}

public class Qu15 {

    public static void main(String[] args) {

        TreeSet<Product> set = new TreeSet<Product>(new nameComparator());

        set.add(new Product(1000, "samsung"));

        set.add(new Product(02, "apple"));

        set.add(new Product(300, "oneplus"));

        set.add(new Product(4, "vivo"));

        set.add(new Product(555, "mi"));

        for (Product p : set) {

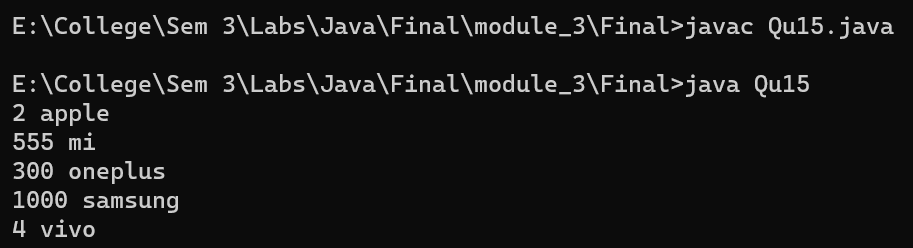
            System.out.println(p.Id + " " + p.name);

        }

    }

}

**Output**

****

**Question 16:**

Write a program to create a HashSet of Products.Demonstrate that no duplicates value are allowed in HashSet.

**Code**

import java.util.\*;

public class Qu16 {

    public static void main(String[] args) {

        HashSet<String> set = new HashSet<>();

        set.add("samsung");

        set.add("apple");

        set.add("mi");

        set.add("vivo");

        set.add("oneplus");

        set.add("samsung");

        set.add("apple");

        set.add("mi");

        set.add("vivo");

        set.add("oneplus");

        for (String p : set) {

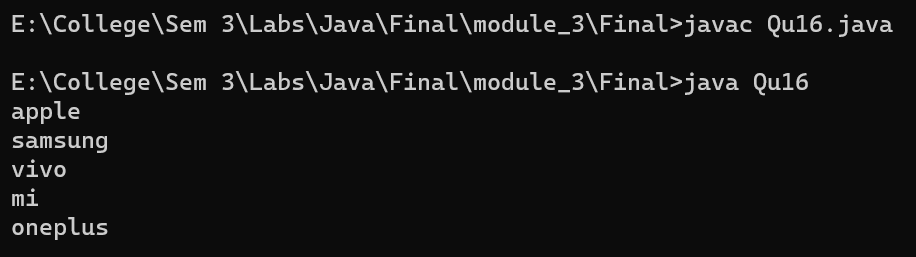
            System.out.println(p);

        }

    }

}

**Output**

****

**Question 17:**

Write a program to demonstrate Thread creation using Runnable interface. While main thread prints 1 to 50 with 1 millisecond pose and child thread print 1 to 100 using 0.5 millisecond pose. Print the name of child and main thread. Main thread needs to wait for child thread to complete.

**Code**

import java.util.\*;

public class Qu17 {

    public static void main(String[] args) {

        Runnable r1 = new Runnable() {

            public void run() {

                for (int i = 1; i <= 50; i++) {

                    System.out.println("Main thread--" + " : " + i);

                    try {

                        Thread.sleep(1000); // main Thread

                    } catch (InterruptedException e) {

                        System.out.println("Thread in sleep mode");

                    }

                }

            }

        };

        for (int i = 1; i <= 100; i++) {

            System.out.println("child thread--" +  " : " + i);

            try {

                Thread.sleep(500); // child Thread

            } catch (InterruptedException e) {

                System.out.println("Thread in sleep mode");

            }

        }

        Thread t1 = new Thread(r1);

        t1.start();

        try {

            t1.join();

        } catch (InterruptedException e) {

            throw new RuntimeException(e);

        }

    }

}