NAME	JACOB JOHN
REGISTER NO.	16BCE2205
E-MAIL	jacob.john2016@vitstudent.ac.in
COURSE	Java Programming

LAB ASSESSMENT #2

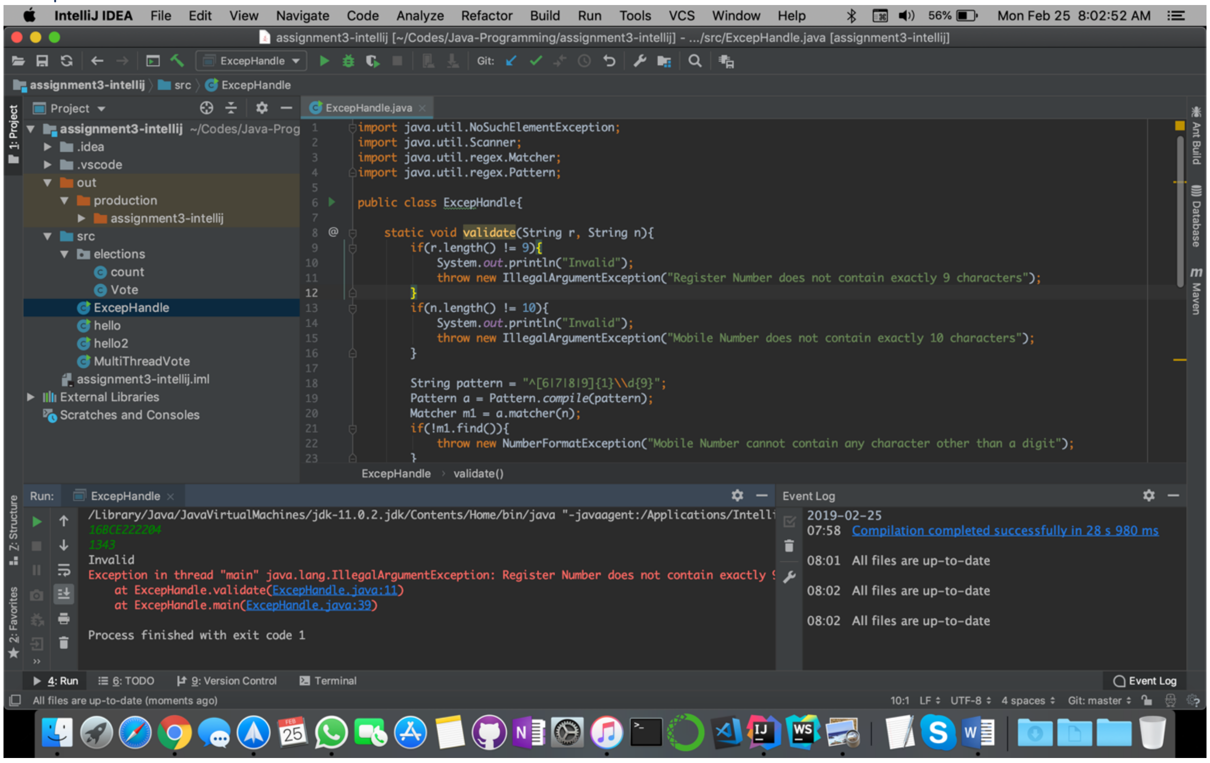
Exception Handling

Read the Register Number and Mobile Number of a student. If the Register Number does not contain exactly 9 characters or if the Mobile Number does not contain exactly 10 characters, throw an IllegalArgumentException. If the Mobile Number contains any character other than a digit, raise a NumberFormatException. If the Register Number contains any character other than digits and alphabets, throw a NoSuchElementException. If they are valid, print the message ‘valid’ else ‘invalid’

Code

```java
import java.util.NoSuchElementException;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class ExcepHandle{

    static void validate(String r, String n){
        if(r.length() != 9){
            System.out.println("Invalid");
            throw new IllegalArgumentException("Register Number does not contain exactly 9 characters");
        }
        if(n.length() != 10){
            System.out.println("Invalid");
            throw new IllegalArgumentException("Mobile Number does not contain exactly 10 characters");
        }

        String pattern = "^[6|7|8|9]{1}\\d{9}";
        Pattern a = Pattern.compile(pattern);
        Matcher m1 = a.matcher(n);
        if(!m1.find()){
            throw new NumberFormatException("Mobile Number cannot contain any character other than a digit");
        }

        String pattern2 = "^[1-9]{2}[A-Z]{3}[0-9]{4}$";
        Pattern b = Pattern.compile(pattern2);
        Matcher m2 = b.matcher(r);
        if(!m2.find()){
            throw new NoSuchElementException("Registration Number cannot contain any character other than digits and alphabets");
        }

    }

    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        String reg = sc.nextLine();
        String no = sc.nextLine();
        sc.close();
        validate(reg, no);
        System.out.println("Valid");
    }
}
```

Output



```java
import java.util.NoSuchElementException;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class ExcepHandle{

    static void validate(String r, String n){
        if(r.length() != 9){
            System.out.println("Invalid");
            throw new IllegalArgumentException("Register Number does not contain exactly 9 characters");
        }
        if(n.length() != 10){
            System.out.println("Invalid");
            throw new IllegalArgumentException("Mobile Number does not contain exactly 10 characters");
        }

        String pattern = "^[6|7|8|9]{1}\\d{9}";
        Pattern a = Pattern.compile(pattern);
        Matcher m1 = a.matcher(n);
        if(!m1.find()){
            throw new NumberFormatException("Mobile Number cannot contain any character other than a digit");
        }
```

```
/Library/Java/JavaVirtualMachines/jdk-11.0.2.jdk/Contents/Home/bin/java "-javaagent:/Applications/Intelli
16BCE222204
1343
Invalid
Exception in thread "main" java.lang.IllegalArgumentException: Register Number does not contain exactly 9
    at ExcepHandle.validate(ExcepHandle.java:11)
    at ExcepHandle.main(ExcepHandle.java:39)

Process finished with exit code 1
```

```
2019-02-25
07:58  Compilation completed successfully in 28 s 980 ms

08:01  All files are up-to-date

08:02  All files are up-to-date

08:02  All files are up-to-date
```

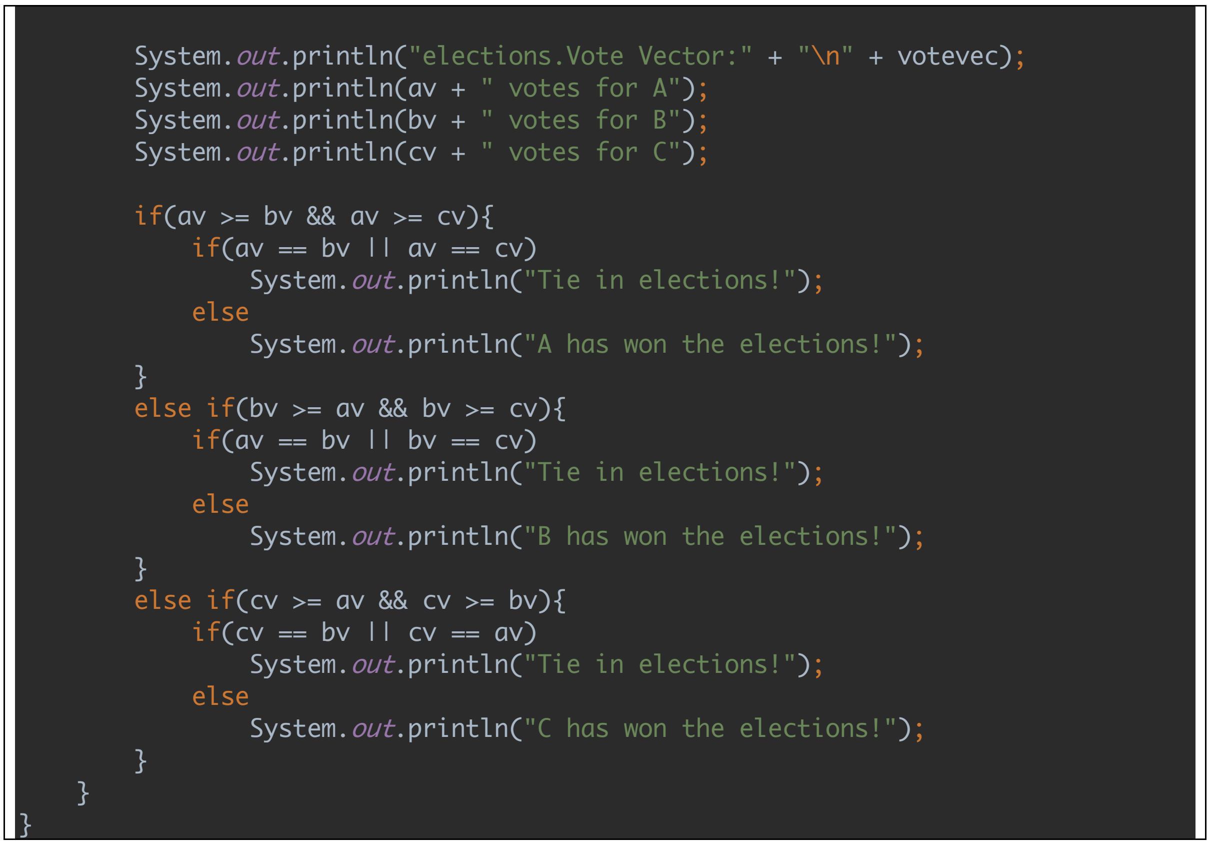
IntelliJ IDEA File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Mon Feb 25 8:02:52 AM
assignment3-intellij [~/Codes/Java-Programming/assignment3-intellij] - .../src/ExcepHandle.java [assignment3-intellij]
assignment3-intellij
.idea
.vscode
out
production
src
elections
count
Vote
ExcepHandle
hello
hello2
MultiThreadVote
assignment3-intellij.iml
External Libraries
Scratches and Consoles
4: Run
6: TODO
9: Version Control
Terminal
Event Log
All files are up-to-date (moments ago)

Multithreading

Three students A, B and C of B.Tech- II year contest for the PR election. With the total strength of 240 students in II year, simulate the vote casting by generating 240 random numbers (1 for student A, 2 for B and 3 for C) and store them in an array. Create four threads to equally share the task of counting the number of votes cast for all the three candidates. Use synchronized method or synchronized block to update the three count variables. The main thread should receive the final vote count for all three contestants and hence decide the PR based on the values received.

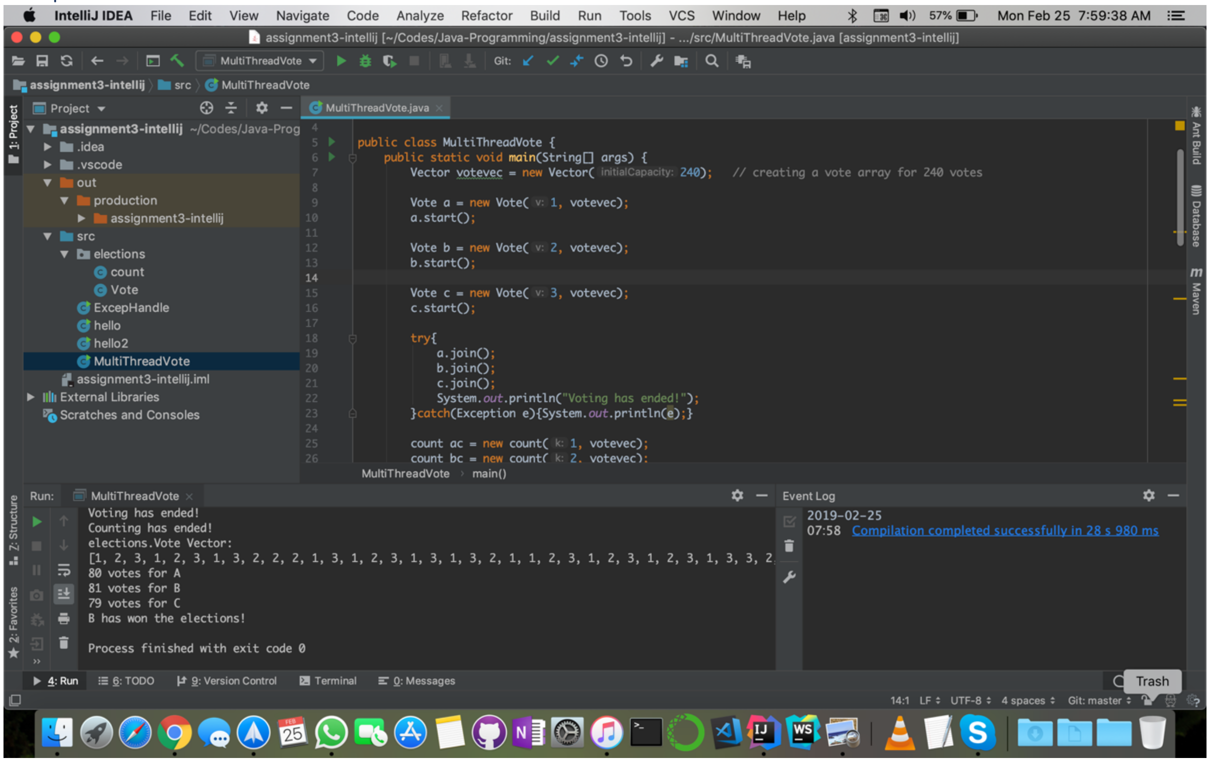
Code

```java
import elections.Vote;
import elections.count;
import java.util.Vector;

public class MultiThreadVote {
    public static void main(String[] args) {
        Vector votevec = new Vector(240);   // creating a vote array for 240
votes

        Vote a = new Vote(1, votevec);
        a.start();

        Vote b = new Vote(2, votevec);
        b.start();

        Vote c = new Vote(3, votevec);
        c.start();

        try{
            a.join();
            b.join();
            c.join();
            System.out.println("Voting has ended!");
        }catch(Exception e){System.out.println(e);}

        count ac = new count(1, votevec);
        count bc = new count(2, votevec);
        count cc = new count(3, votevec);

        ac.start();
        bc.start();
        cc.start();

        try{
            ac.join();
            bc.join();
            cc.join();
            System.out.println("Counting has ended!");
        }catch(Exception e){System.out.println(e);}

        int av = ac.count;
        int bv = bc.count;
        int cv = cc.count;
```



```
        System.out.println("elections.Vote Vector:" + "\n" + votevec);
        System.out.println(av + " votes for A");
        System.out.println(bv + " votes for B");
        System.out.println(cv + " votes for C");

        if(av >= bv && av >= cv){
            if(av == bv || av == cv)
                System.out.println("Tie in elections!");
            else
                System.out.println("A has won the elections!");
        }
        else if(bv >= av && bv >= cv){
            if(av == bv || bv == cv)
                System.out.println("Tie in elections!");
            else
                System.out.println("B has won the elections!");
        }
        else if(cv >= av && cv >= bv){
            if(cv == bv || cv == av)
                System.out.println("Tie in elections!");
            else
                System.out.println("C has won the elections!");
        }
    }
}
```

Output



```
public class MultiThreadVote {
    public static void main(String[] args) {
        Vector votevec = new Vector( initialCapacity: 240);   // creating a vote array for 240 votes

        Vote a = new Vote( v: 1, votevec);
        a.start();

        Vote b = new Vote( v: 2, votevec);
        b.start();

        Vote c = new Vote( v: 3, votevec);
        c.start();

        try{
            a.join();
            b.join();
            c.join();
            System.out.println("Voting has ended!");
        }catch(Exception e){System.out.println(e);}

        count ac = new count( k: 1, votevec);
        count bc = new count( k: 2, votevec);
```

```
Voting has ended!
Counting has ended!
elections.Vote Vector:
[1, 2, 3, 1, 2, 3, 1, 3, 2, 2, 2, 1, 3, 1, 2, 3, 1, 3, 1, 3, 2, 1, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 3, 3, 2
80 votes for A
81 votes for B
79 votes for C
B has won the elections!

Process finished with exit code 0
```

```
2019-02-25
07:58 Compilation completed successfully in 28 s 980 ms
```

IntelliJ IDEA File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
57% Mon Feb 25 7:59:38 AM
assignment3-intellij [~/Codes/Java-Programming/assignment3-intellij] - .../src/MultiThreadVote.java [assignment3-intellij]
assignment3-intellij › src › MultiThreadVote
Project
assignment3-intellij ~/Codes/Java-Prog
.idea
.vscode
out
production
assignment3-intellij
src
elections
count
Vote
ExcepHandle
hello
hello2
MultiThreadVote
assignment3-intellij.iml
External Libraries
Scratches and Consoles
MultiThreadVote.java
MultiThreadVote › main()
Run: MultiThreadVote
Event Log
4: Run
6: TODO
9: Version Control
Terminal
0: Messages
Trash
14:1 LF UTF-8 4 spaces Git: master

File Handling

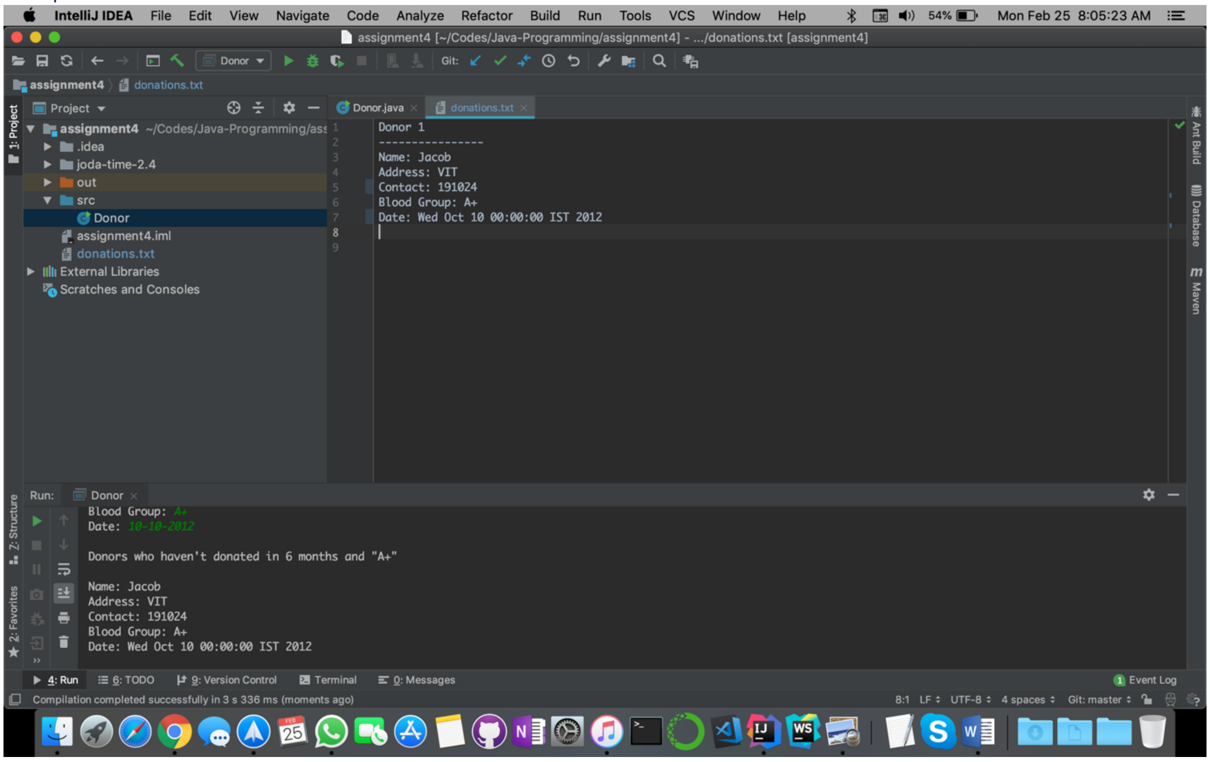
Define a class 'Donor' to store the below mentioned details of a blood donor. Name, age, Address, Contact number, blood group, date of last donation Create 'n' objects of this class for all the regular donors at Vellore. Write these objects to a file. Read these objects from the file and display only those donors' details whose blood group is 'A+ve' and had not donated for the recent six months.

Code

```java
import java.io.*;
import java.nio.file.Files;
import java.nio.file.NoSuchFileException;
import java.nio.file.Paths;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import org.joda.time.LocalDate;
import org.joda.time.DateTime;
import org.joda.time.Period;
import java.util.Date;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Donor {
    String name,addr,contact,bldgrp;
    Date date;

    public static void main(String[] args) throws IOException{
        int n, i;
        SimpleDateFormat ft = new SimpleDateFormat("MM-dd-yyyy");
        String temp;
        String pattern = "[A|B|O|AB][+|-]";
        Matcher m;
        Pattern r = Pattern.compile(pattern);

        // delete existing file first
        try{
            Files.deleteIfExists(Paths.get("donations.txt"));
        }
        catch(NoSuchFileException e)
        {
            System.out.println("No such file/directory exists");
        }
        catch(IOException e)
        {
            System.out.println("Invalid permissions.");
        }
        System.out.println("Deletion successful.");

        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of donors: ");
        n = sc.nextInt();
        sc.nextLine();
        Donor arr[] = new Donor[n];                                // creating
array of n donors
```

```
        ByteArrayOutputStream outputstream = new ByteArrayOutputStream();
        FileOutputStream fos = new FileOutputStream("donations.txt");    //
creating file to write to

        for(i = 0; i < n; i++){
            arr[i] = new Donor();                                    //
initializing new donor
            //taking input from user
            System.out.print("Name: ");
            arr[i].name = sc.nextLine();
            System.out.print("Address: ");
            arr[i].addr = sc.nextLine();
            System.out.print("Contact: ");
            arr[i].contact = sc.nextLine();
            arr[i].bldgrp = "";
            m = r.matcher(arr[i].bldgrp);
            while(!m.find()){
                System.out.print("Blood Group: ");
                arr[i].bldgrp = sc.nextLine();
                m = r.matcher(arr[i].bldgrp);
            }
            boolean flag = false;
            while(!flag){
                System.out.print("Date: ");
                temp = sc.nextLine();
                try {
                    arr[i].date = ft.parse(temp);
                    flag = true;
                } catch (ParseException e) {
                    flag = false;
                    System.out.println("Unparseable using " + ft);
                }
            }


            // concatenating all properties in donor object
            // and converting to a byte stream
            outputstream.write(("Donor " + (i+1) + "\n").getBytes());
            outputstream.write("----------------\n".getBytes());
            outputstream.write("Name: ".getBytes());
            outputstream.write(arr[i].name.getBytes());
            outputstream.write("\n".getBytes());
            outputstream.write("Address: ".getBytes());
            outputstream.write(arr[i].addr.getBytes());
            outputstream.write("\n".getBytes());
            outputstream.write("Contact: ".getBytes());
            outputstream.write(arr[i].contact.getBytes());
            outputstream.write("\n".getBytes());
            outputstream.write("Blood Group: ".getBytes());
            outputstream.write(arr[i].bldgrp.getBytes());
            outputstream.write("\n".getBytes());
            outputstream.write("Date: ".getBytes());
            outputstream.write(arr[i].date.toString().getBytes());
            outputstream.write("\n".getBytes());
            outputstream.write("\n".getBytes());
        }
        // create a byte array to be written
        byte c[] = outputstream.toByteArray();
        fos.write(c);                                           // writing byte
array to file
        fos.close();


        // reading data from file
        byte[] fileContent = Files.readAllBytes(Paths.get("donations.txt"));
        String fileInput = new String(fileContent);
        String[] content = fileInput.split("\n");
        Date dt2;
        String[] fileDate, fileGrp;
        SimpleDateFormat ft2 = new SimpleDateFormat("E MMM dd HH:mm:ss z yyyy");
        LocalDate current = LocalDate.now();

        for(i = 5; i < content.length; i += 8){
            fileGrp = content[i].split(": ");
            fileDate = content[i+1].split(": ");

            System.out.println("\n" + "Donors who haven't donated in 6 months and
\"A+\"" + "\n");
            try {
                dt2 = ft2.parse(fileDate[1]);
                DateTime dt3 = new DateTime(dt2);
                LocalDate dt1 = new LocalDate(dt3);
                Period p = new Period(dt1, current);

                // if more than 6 months and blood group is A+, print details

                if((p.getMonths() > 6 | p.getYears() >= 1) &&
fileGrp[1].equals("A+"))
                {
                    System.out.println(content[i-3]); // name
                    System.out.println(content[i-2]); // address
                    System.out.println(content[i-1]); // contact
                    System.out.println(content[i]);   // blood group
                    System.out.println(content[i+1]); // date
                    System.out.println("\n");
                }
            } catch (ParseException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Output



```
Donor 1
----------------
Name: Jacob
Address: VIT
Contact: 191024
Blood Group: A+
Date: Wed Oct 10 00:00:00 IST 2012
```

```
Blood Group: A+
Date: 10-10-2012

Donors who haven't donated in 6 months and "A+"

Name: Jacob
Address: VIT
Contact: 191024
Blood Group: A+
Date: Wed Oct 10 00:00:00 IST 2012
```

Compilation completed successfully in 3 s 336 ms (moments ago)

Java Collection

Write a program to demonstrate the knowledge of students in working with Java collection framework.

Eg., Assume only a maximum of 3 courses can be registered by a student for week end semester classes. Create a hashmap ‘h1’ with ‘n’ key-value pairs where keys are the names of students and values are the courses registered by them. Create another hashmap ‘h2’ with ‘m’key-value pairs where keys are the names of courses offered for B.Tech and values are the names of faculty handling the courses. Write appropriate code to:

- Add or remove a student from h1
- Iterate over the maps and display the key-value pairs stored in them
- Given a student name, fetch the names of all those who teach him/her.
- if the elements of h1 are:

Stud name	**Courses registered**
A	Python, Math, C
B	C, C++
C	C++, Physics, Chemistry

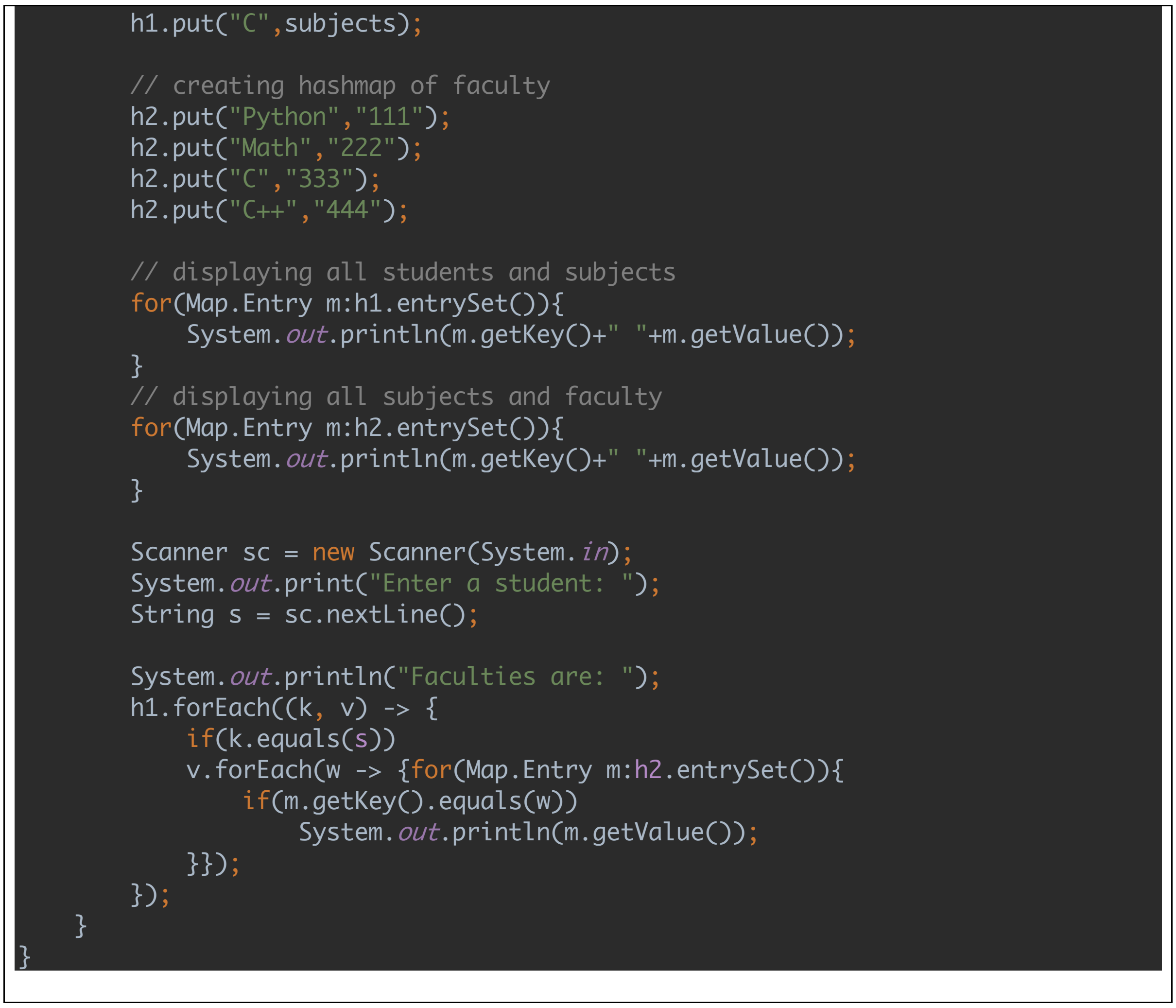
- And if the elements of h2 are:

Course Name	**Faculty**
Python	111
Math	222
C	333
C++	444

- For the student “B”, faculty should be displayed as 333 and 444.

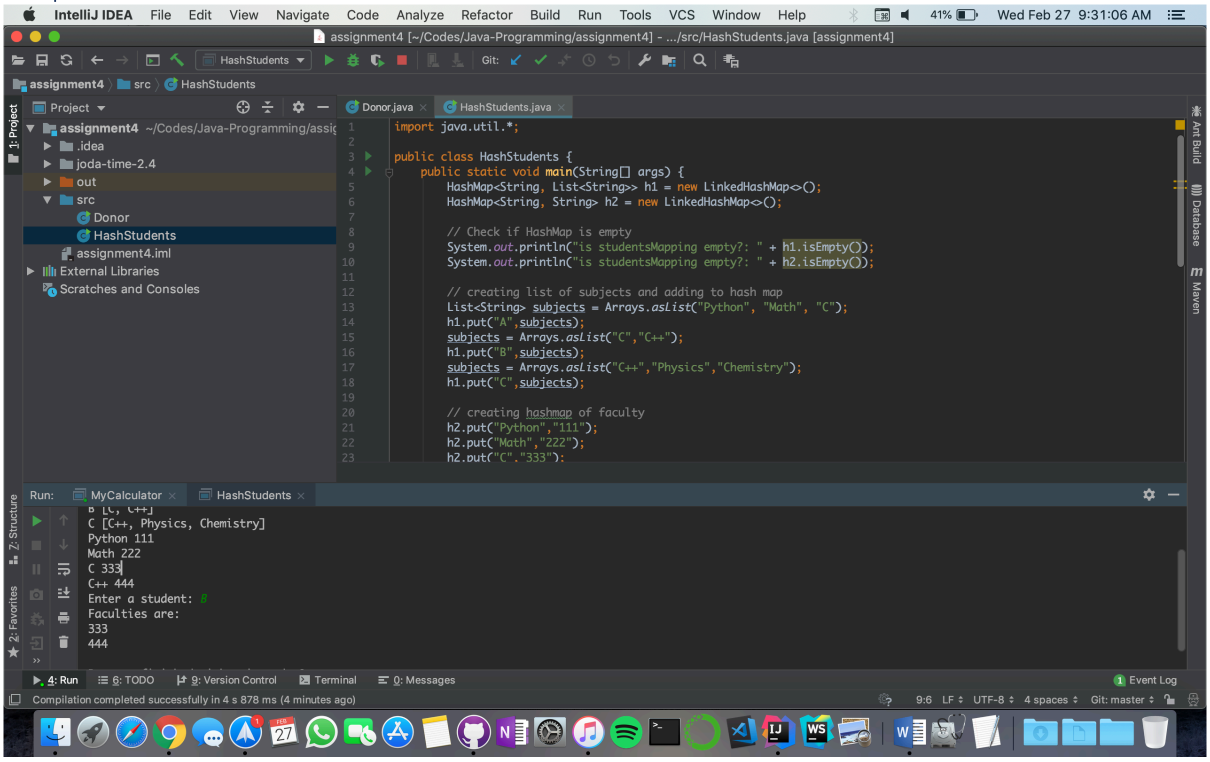
Code

```java
import java.util.*;

public class HashStudents {
    public static void main(String[] args) {
        HashMap<String, List<String>> h1 = new LinkedHashMap<>();
        HashMap<String, String> h2 = new LinkedHashMap<>();

        // Check if HashMap is empty
        System.out.println("is studentsMapping empty?: " + h1.isEmpty());
        System.out.println("is studentsMapping empty?: " + h2.isEmpty());

        // creating list of subjects and adding to hash map
        List<String> subjects = Arrays.asList("Python", "Math", "C");
        h1.put("A",subjects);
        subjects = Arrays.asList("C","C++");
        h1.put("B",subjects);
        subjects = Arrays.asList("C++","Physics","Chemistry");
```



```java
        h1.put("C",subjects);

        // creating hashmap of faculty
        h2.put("Python","111");
        h2.put("Math","222");
        h2.put("C","333");
        h2.put("C++","444");

        // displaying all students and subjects
        for(Map.Entry m:h1.entrySet()){
            System.out.println(m.getKey()+" "+m.getValue());
        }
        // displaying all subjects and faculty
        for(Map.Entry m:h2.entrySet()){
            System.out.println(m.getKey()+" "+m.getValue());
        }

        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a student: ");
        String s = sc.nextLine();

        System.out.println("Faculties are: ");
        h1.forEach((k, v) -> {
            if(k.equals(s))
            v.forEach(w -> {for(Map.Entry m:h2.entrySet()){
                if(m.getKey().equals(w))
                    System.out.println(m.getValue());
            }});
        });
    }
}
```

Output



assignment4 [~/Codes/Java-Programming/assignment4] - .../src/HashStudents.java [assignment4]

```java
import java.util.*;

public class HashStudents {
    public static void main(String[] args) {
        HashMap<String, List<String>> h1 = new LinkedHashMap<>();
        HashMap<String, String> h2 = new LinkedHashMap<>();

        // Check if HashMap is empty
        System.out.println("is studentsMapping empty?: " + h1.isEmpty());
        System.out.println("is studentsMapping empty?: " + h2.isEmpty());

        // creating list of subjects and adding to hash map
        List<String> subjects = Arrays.asList("Python", "Math", "C");
        h1.put("A",subjects);
        subjects = Arrays.asList("C","C++");
        h1.put("B",subjects);
        subjects = Arrays.asList("C++","Physics","Chemistry");
        h1.put("C",subjects);

        // creating hashmap of faculty
        h2.put("Python","111");
        h2.put("Math","222");
        h2.put("C","333");
```

```
B [C, C++]
C [C++, Physics, Chemistry]
Python 111
Math 222
C 333
C++ 444
Enter a student: B
Faculties are:
333
444
```

Compilation completed successfully in 4 s 878 ms (4 minutes ago)