NAME: VENNELA G

REG NO: 20BDS0146

SUBJECT: **JAVA PROGRAMMING**

LAB SLOT: L13+14

ASSESSMENT NO: 3

1. Write a program to demonstrate the knowledge of students in Java Exception handling. Eg., 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’.

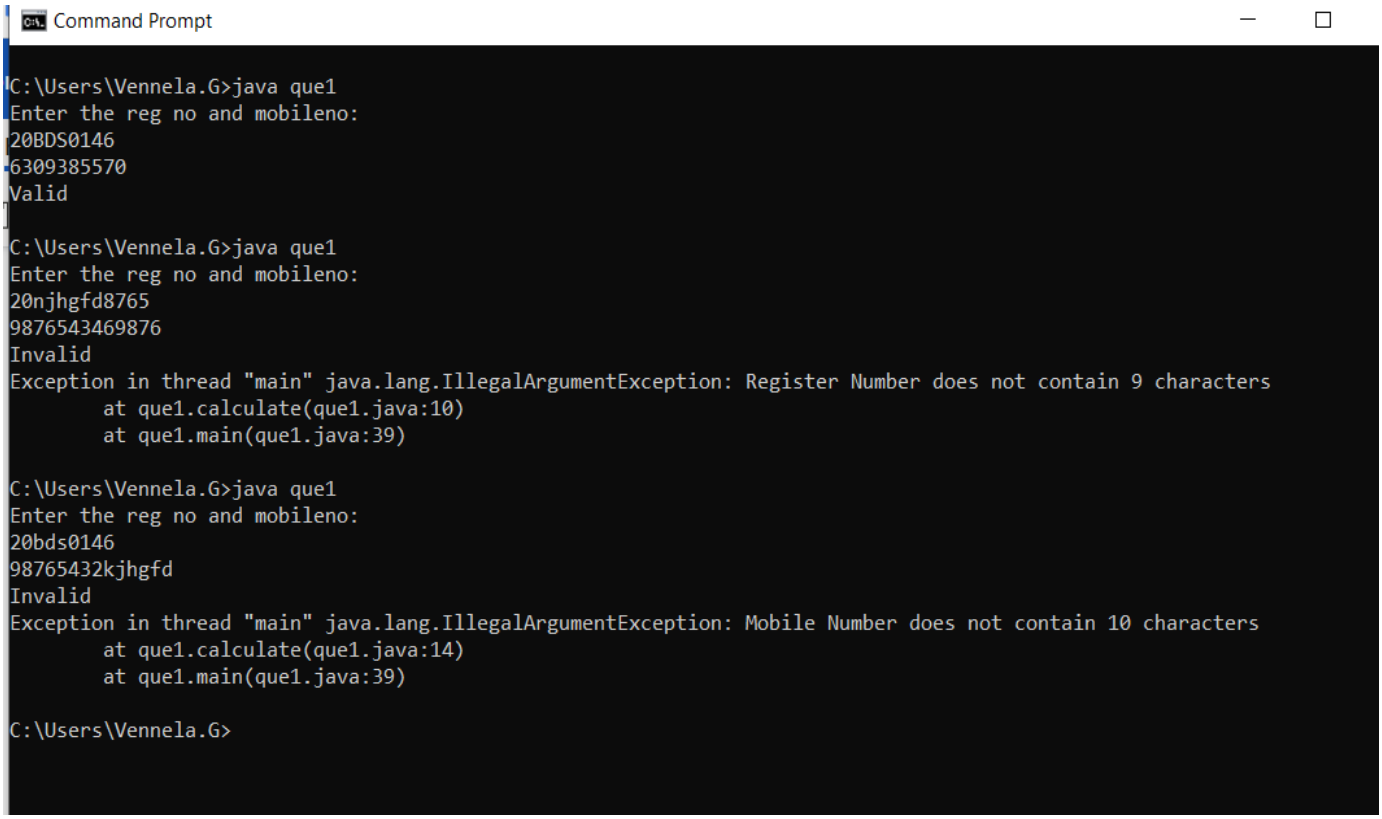
CODE:

```
import java.util.*;
import java.util.regex.Matcher;
import java.util.regex.Pattern;



public class que1{
static void calculate(String s1, String s2){
if(s1.length() != 9){
```

```
System.out.println("Invalid");
throw new IllegalArgumentException("Register Number does not contain 9 characters");
}
if(s2.length() != 10){
System.out.println("Invalid");
throw new IllegalArgumentException("Mobile Number does not contain 10 characters");
}
String pattern2 = "^[1-9]{2}[A-Z a-z]{3}[0-9]{4}$";
Pattern  b = Pattern.compile(pattern2);
Matcher  m2 = b.matcher(s1);
if(m2.find()){
throw new NoSuchElementException("Registration Number cannot contain any character other than digits and alphabets");
}

String pattern = "^[6|7|8|9]{1}\\d{9}";
Pattern a = Pattern.compile(pattern);
Matcher  m1 = a.matcher(s2);
if(!m1.find()){
```

```
throw new NumberFormatException("Mobile Number shouldnt have any character other than a digit");
}


}
public static void main(String args[]){
String reg;
String no;
System.out.println("Enter the reg no and mobileno:");
try (Scanner sc = new Scanner(System.in)) {
reg = sc.nextLine();
no = sc.nextLine();
}
calculate(reg, no);
System.out.println("Valid");
}
}
```

OUTPUT:



2. Write a program to demonstrate the knowledge of students in Java Exception handling. Create a class by name Employee with members – Employee ID, Name and year of birth. The Employee ID is a string that contains the ID in the format year-designation-number. The year is represented with the last two digits. The designation is a single letter code - 'F' for faculty and 'S' for staff. The number is a 3 digit number. (Example: 81-F-112 79-S-254) Write a program to read the employee details and validate the employee code. If the employee code is incorrect throw a user-defined exception "InvalidEmployeeCode" else create the Employee object and display the details of the employee.

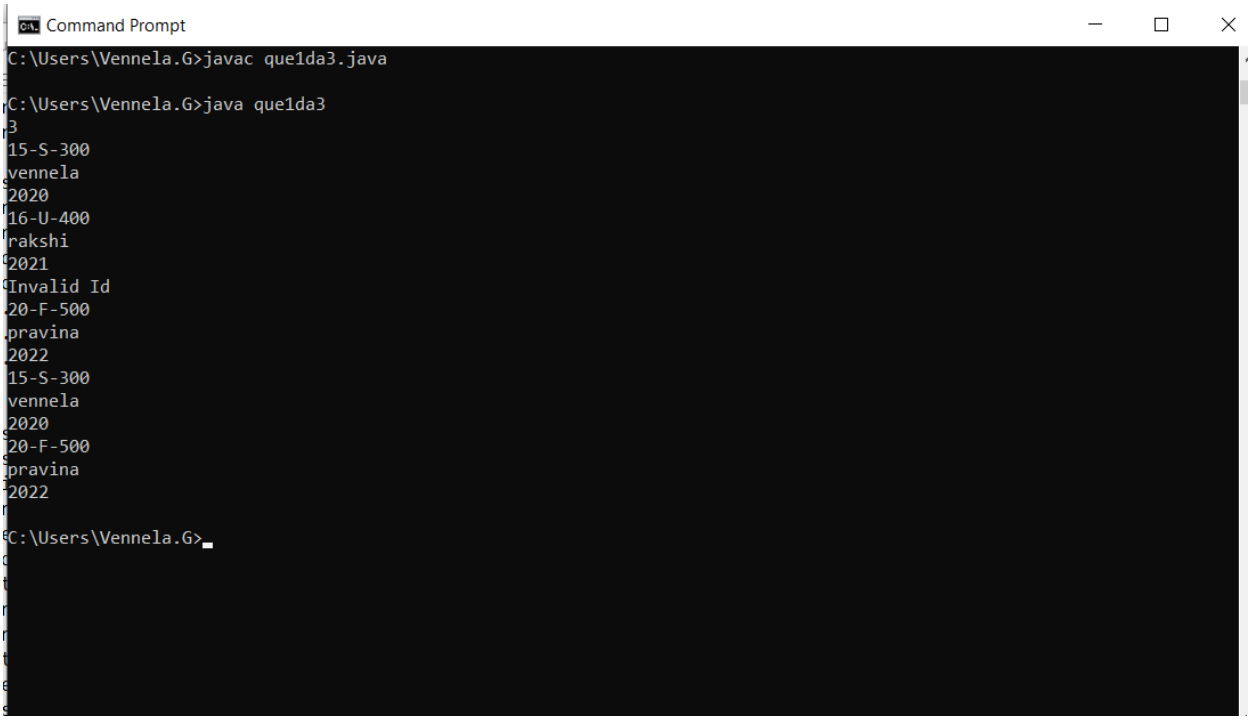
CODE:

```
import java.io.*;
import java.util.*;

class Employee {
String Emp_id;
String emp_name;
int dobyr;
Employee(String Emp_id, String emp_name, int dobyr){
this.Emp_id = Emp_id;
this.emp_name =emp_name;
this.dobyr = dobyr;
}
}
class notvalidId extends Exception{}
class que1da3 {
public static void main(String args[]){
Scanner sc = new Scanner(System.in);
int empno= sc.nextInt();
```

```
Employee[] emp = new Employee[empno];
int temp = empno;
String id;String name; int year;
String [] breakup;
int total = 0;
while(temp>0){
id =sc.next();
name = sc.next();
year = sc.nextInt();
breakup = id.split("-");
try{
if ((breakup[1].equals("S")) ||(breakup[1].equals("F"))){
}else{
throw new notvalidId();
}
for (int i=0;i<breakup[2].length();i++){
if(!(Character.isDigit(breakup[2].charAt(i)))){
System.out.println("Here2");
throw new notvalidId();
}
```

```
}
emp[total] = new Employee(id,name,year);
total++;

}catch(notvalidId ex){
System.out.println("Invalid Id");
}
temp--;
}
for (int i=0;i<total;i++){
System.out.println(emp[i].Emp_id);
System.out.println(emp[i].emp_name);
System.out.println(emp[i].dobyr);

}
}
}
```

OUTPUT:



3. Write a program to demonstrate the knowledge of students in multithreading. Eg., Three students A, B and C of B.Tech-IT 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:

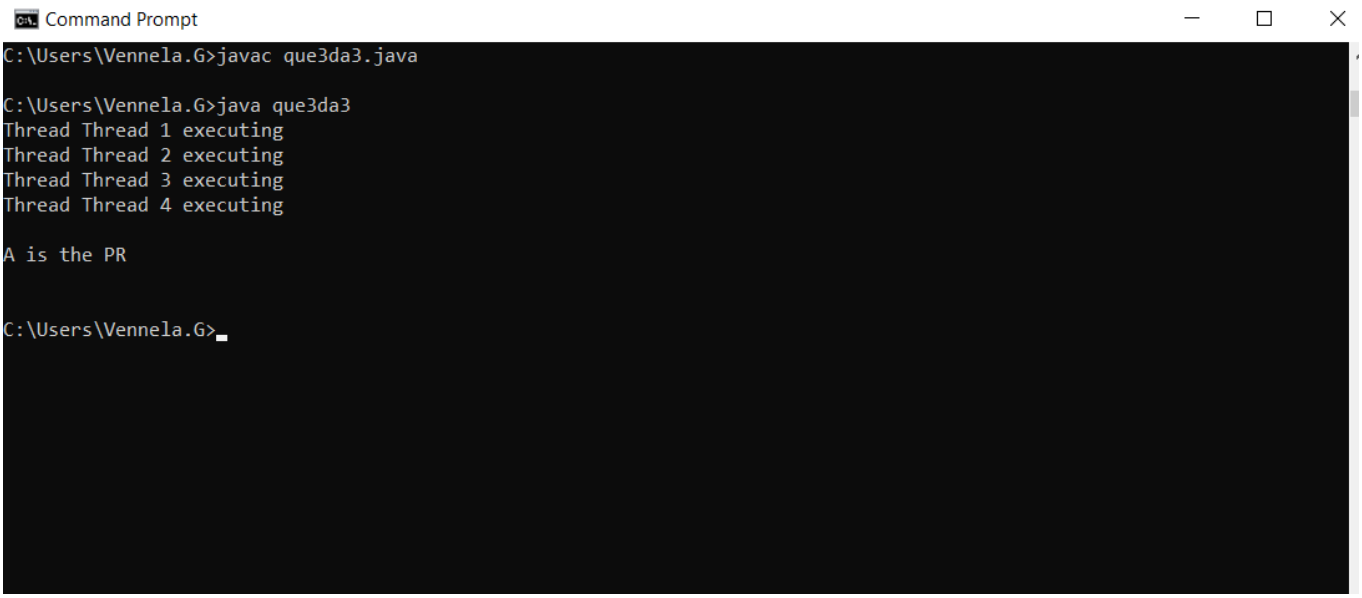
```
import java.io.*;
import java.util.*;

class contest extends Thread
{
int low,high;
static int[]cont = new int[240];
static
{
for(int i=0;i<240;i++)
{
cont[i]=0+(int)(Math.random() * (3-0)+1);
}
}
contest(String name,int l,int h)
{
low =l;
high=h;
```

```
System.out.println("Thread "+name+" executing");
start();
}
public int[] count()
{
int[] arr = new int[3];
for(int i=low;i<high;i++)
{
if(cont[i]==1)
{
arr[0]+=1;
}
else if(cont[i]==2)
{
arr[1]+=1;
}
else
arr[2]+=1;
}
return arr;
```

```
}
}
class que3da3
{
public static void main(String args[])
{
contest t1 = new contest("Thread 1",0,59);
contest t2 = new contest("Thread 2",60,119);
contest t3 = new contest("Thread 3",120,179);
contest t4 = new contest("Thread 4",180,239);
try
{
t1.join();
t2.join();
t3.join();
t4.join();
}
catch(Exception e)
{
System.out.println("Interrupted");
```

```
}
int[] num1 = t1.count();
int[] num2 = t2.count();
int[] num3 = t3.count();
int[] num4 = t4.count();
int x = num1[0]+num2[0]+num3[0]+num4[0];
int y = num1[1]+num2[1]+num3[1]+num4[1];
int z = num1[2]+num2[2]+num3[2]+num4[2];
if(x>y && x>z)
System.out.println("\nA is the PR\n");
else if(y>x && y>z)
System.out.println("\nB is the PR\n");
else
System.out.println("\nC is the PR\n");
}
}
```

OUTPUT:



4. Create a User Defined Exception, Implement a remainder ClockAPP to the person saying Good Morning, Good Afternoon and Good Night by showing the current time. Write a code that assigns different values to the String variable timeofDay based on the value of time in hours. Raise remainder if time is 5 and 12, including 5 but not including 12, set timeofDay to “Morning, Have Fresh Vegetable Juice and then Sugar Tablet with mild walking”. Raise Exception, if time is between 12 and 17, including 12 but not including 17, set timeofDay as “its Day Time, after lunch have tablet to avoid sleep”. If time is between 17 and 18.30. Including 17 but not including 18.30, set timeofDays as “Hello, Good Evening have a dinner” and finally, if time variable does not satisfy all the above condition, set time as “Night, Go for sleep”.

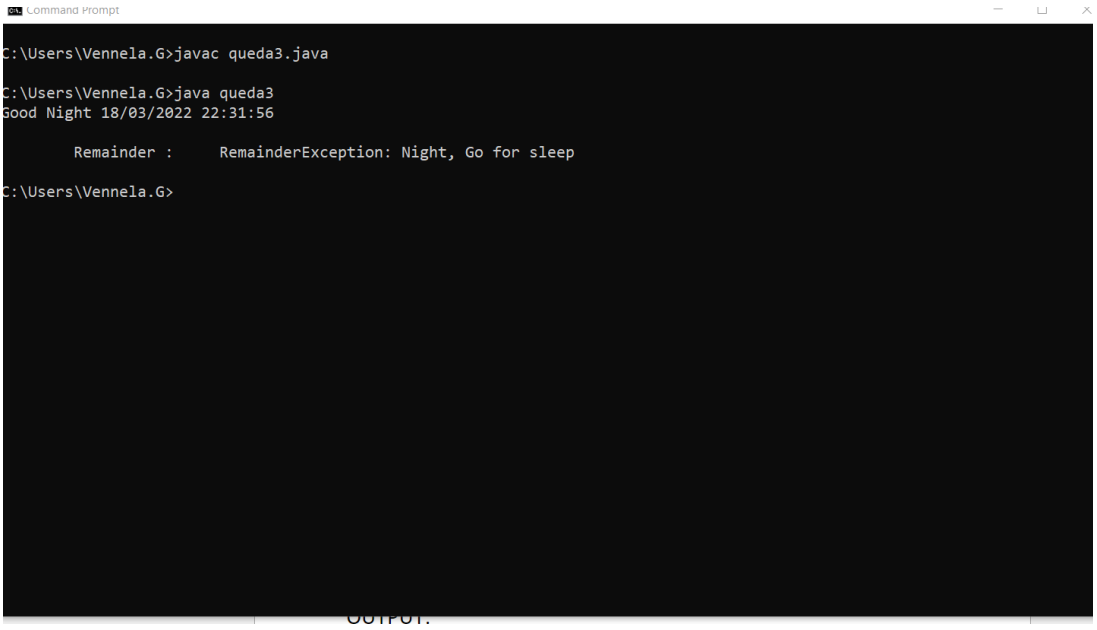
CODE:

```
import java.text.SimpleDateFormat;
import java.util.*;

class queda3{
public static void main(String[] arg){
SimpleDateFormat formatter = new
SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
SimpleDateFormat HourFormatter = new
SimpleDateFormat("HH");
SimpleDateFormat MinFormatter = new
SimpleDateFormat("mm");

Date date = new Date();
int Hours=Integer.parseInt(HourFormatter.format(date));
int Min=Integer.parseInt(MinFormatter.format(date));
try{
if(Hours>=5 && Hours<12){
System.out.print("Good Morning ");
System.out.print(formatter.format(date));
```

```
throw new RemainderException("Morning, Have Fresh
Vegetable Juice and then Sugar Tablet with mild walking");
}
else if(Hours>=12 && Hours<17){
System.out.println("Good Afternoon ");
System.out.print(formatter.format(date));
throw new RemainderException("its Day Time, after lunch have
tablet to avoid sleep");
}
else if(Hours>=17 && (Hours+(Min+30)/60)<19){
System.out.print("Good Evening ");
System.out.print(formatter.format(date));
throw new RemainderException("Hello, Good Evening have a
dinner");

}
else{
System.out.print("Good Night ");
System.out.print(formatter.format(date));
throw new RemainderException("Night, Go for sleep");
}
```

```
}
catch(RemainderException e){
System.out.println("\n\n\tRemainder :\t"+e);
}
}
}
class RemainderException extends Exception{
RemainderException(String s){
super(s);
}
}
```

OUTPUT:



OUTPUT: