OOP LAB 5

Lab Exercises

1. Design an interface called Series with the following methods
    1. getNext (returns the next number in the series)
    2. reset(to restart the series)
    3. setStart (to set the value from which the series should start)

Design two classes named **ByTwos and ByFives** to implement the methods of the interface Series such that it generates a series of numbers, each two/five greater than the previous one. Also design a class which will include the main method for referencing the interface.

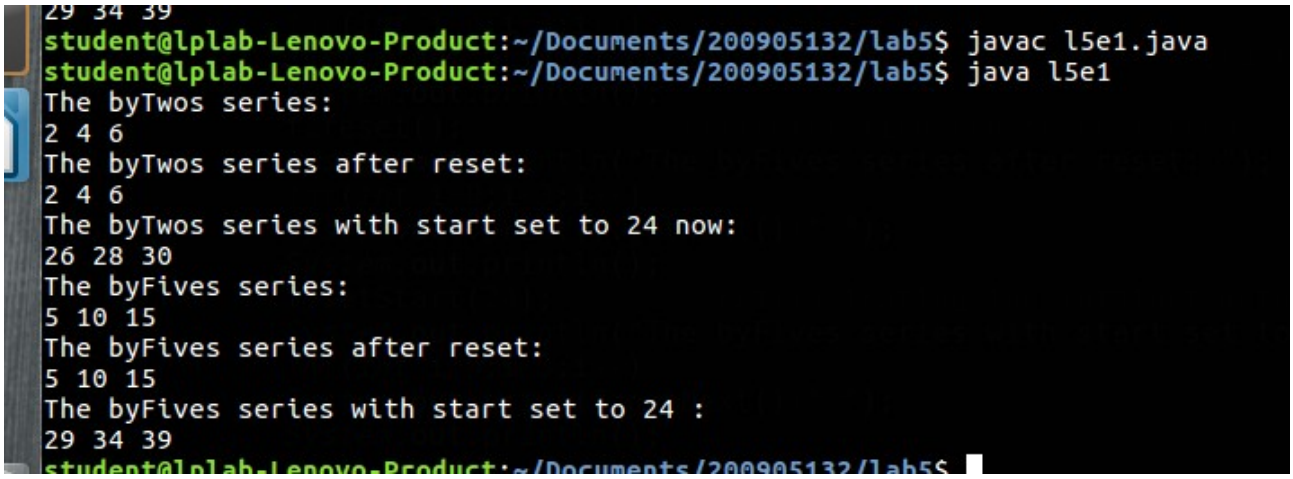
Solution:

```
import java.util.Scanner;


interface Series{
    void reset();
    void setStart(int x);
    int getNext();
}




class byTwos implements Series{
    int start=0;
    public void reset(){
        start=0;
    }
    public void setStart(int x){
```

```
        start=x;
    }
    public int getNext(){
        start+=2;
        return start;
    }
}



class byFives implements Series{
    int start=0;
    public void reset(){
        start=0;
    }
    public void setStart(int x){
        start=x;
    }
    public int getNext(){
        start+=5;
        return start;
    }
}



public class l5e1
{
```

```
public static void main(String[] args) {
    Scanner sc=new Scanner(System.in);
    //illustrating the byTwos class methods
    byTwos t=new byTwos();
    System.out.println("The byTwos series: ");
    for(int i=0;i<3;i++)
    System.out.print(t.getNext()+" ");
    System.out.println();
    t.reset();       //illustrating the reset method
    System.out.println("The byTwos series after reset: ");
    for(int i=0;i<3;i++)
    System.out.print(t.getNext()+" ");      //illustrating the getNext method
    System.out.println();
    t.setStart(24);         //illustrating the setStart method
    System.out.println("The byTwos series with start set to 24 now: ");
    for(int i=0;i<3;i++)
    System.out.print(t.getNext()+" ");
    System.out.println();

    //illustrating the byFives class methods
    byFives f=new byFives();
    System.out.println("The byFives series: ");
    for(int i=0;i<3;i++)
    System.out.print(f.getNext()+" ");      //illustrating the getNext method
    System.out.println();
    f.reset();            //illustrating the reset method
    System.out.println("The byFives series after reset: ");
    for(int i=0;i<3;i++)
```

```
        System.out.print(f.getNext()+" ");
        System.out.println();
        f.setStart(24);         //illustrating the setStart method
        System.out.println("The byFives series with start set to 24 : ");
        for(int i=0;i<3;i++)
        System.out.print(f.getNext()+" ");
        System.out.println();
    }
}
```



2. Define a class CurrentDate with data members day, month and year. Define a method createDate() to create date object by reading values from keyboard. Throw a user defined exception by name InvalidDayException if the day is invalid and InvalidMonthException if month is found invalid and display current date if the date is valid. Write a test program to illustrate the functionality.

```
import java.util.Scanner;

class CurrentDate{
    int day,month,year;
    Scanner sc=new Scanner(System.in);
    void createDate(){
        day=sc.nextInt();
        month=sc.nextInt();
        year=sc.nextInt();
    }
    void display(){
        System.out.println("Current date is "+day+"/"+month+"/"+year);
    }
```

```
}

class InvalidDayException extends Exception{
   public String toString(){
      return ("Invalid day!");
   }
}

class InvalidMonthException extends Exception{
   public String toString(){
      return ("Invalid month!");
   }
}

public class Main
{
        public static void main(String[] args) {
                CurrentDate c=new CurrentDate();
                System.out.println("Enter a date in day/month/year format");
                c.createDate();
                try{
                if(c.day<1)
                throw new InvalidDayException();
                switch(c.month){
                   case 2:if(c.day>28)
                            throw new InvalidDayException();
                         break;
                   case 4:
                   case 6:
                   case 9:
                   case 11: if(c.day>30)
                            throw new InvalidDayException();
                         break;
                   default : if(c.day>31)
                   throw new InvalidDayException();
                }
                if(c.month>12||c.month<1)
                throw new InvalidMonthException();
                c.display();
                }
                catch(InvalidDayException err){
                   System.out.println(err.toString());
                }
                catch(InvalidMonthException err){
                   System.out.println(err.toString());
                }
        }
```

}

```
Enter a date in day/month/year format
4 12 2022
Current date is 4/12/2022
```

```
Enter a date in day/month/year format
-3 4 2003
Invalid day!
```

```
Enter a date in day/month/year format
31 2 2005
Invalid day!
```

```
Enter a date in day/month/year format
3 13 2012
Invalid month!
```

3. Design a class Student with the methods, getNumber and putNumber to read and display the Roll No. of each student and getMarks() and putMarks() to read and display their marks. Create an interface called Sports with a method putGrade() that will display the grade obtained by a student in Sports. Design a class called Result that will implement the method putGrade() and generate the final result based on the grade in sports and the marks obtained from the superclass Student. Include appropriate instance variables for the classes.

```
import java.util.Scanner;
interface Sports{
    void putGrade();
}
class Student {
    int rno,marks;
    char grade;
    Scanner sc=new Scanner(System.in);
    Student(char a){
        grade=a;
```

```
    }
    void getNumber(){
        System.out.println("Enter the rno of student");
        rno=sc.nextInt();
    }
    void putNumber(){
        System.out.println("This student's rno is "+rno);
    }
    void getMarks(){
        System.out.println("Enter the marks of student");
        marks=sc.nextInt();
    }
    void putMarks(){
        System.out.println("This student's marks is "+marks);
    }
}
class result extends Student implements Sports{
    char result;
    result(char ch){
        super(ch);
    }
    public void putGrade(){
        System.out.println("The student's grade in sports is "+grade);
    }
    //if a student scores F(fail) in sports, then he has failed, else he has passed with the
grades
    //determined on the basis of marks.
    void showResult(){
        if(grade=='F')
    result='F';
    else if(marks>=90)
    result='A';
    else if(marks>=80)
    result='B';
    else if(marks>=70)
    result='C';
    else if(marks>=60)
    result='D';
    else if(marks>=40)
    result='E';
    else result='F';
        if(result=='F')
        System.out.println("The student has failed");
        else
        System.out.println("The student's final result is "+result);
```

```
    }
}
public class Main
{
        public static void main(String[] args) {
                result r=new result('C');
                r.getNumber();
                r.getMarks();
                r.putNumber();
                r.putMarks();
                r.putGrade();
                r.showResult();
        }
}
```

```
Enter the rno of student
34
Enter the marks of student
56
This student's rno is 34
This student's marks is 56
The student's grade in sports is F
The student has failed
```

```
Enter the rno of student
23
Enter the marks of student
89
This student's rno is 23
This student's marks is 89
The student's grade in sports is C
The student's final result is B
```