BSCCS2005: Jan 2024 OPE2 Questions with Test Cases and Solutions

1 Session 1

1.1 Session 1 Type 1

Problem Statement

Write a Java program that, given as input `name`, `age` and `chronicCondition` of some patients, prints the filtered stream of patients whose `age` is below `30` and `chronicCondition` is `Diabetes`. Complete the program as specified below.

- Class `Patient` has/should have the following members:
  - Private instance variables `String name`, `int age` and `String chronicCondition`
  - A constructor to initialize instance variables
  - Method `toString` to print in the format shown in the test cases
  - Accessor methods for `age` and `chronicCondition`
  - Method `patientProcessor` should take an `ArrayList` of `Patient` objects as input and returns a filtered stream of diabetic patients who are below 30 years.
- Class `StreamTest` has the following members:
  - Method `main` creates an `ArrayList` of `Patient` objects by taking input in the order of `name`, `age` and `chronicCondition` then invokes the method `patientProcessor` to filter patients whose `age` is below `30` and `chronicCondition` is `Diabetes` and finally display those patients.

What you have to do

- Define method `patientProcessor` in class `Patient`.

Template Code

```
import java.util.*;
import java.util.stream.*;
class Patient {
    private String name;
    private int age;
    private String chronicCondition;
    public Patient(String n, int a, String cC) {
        name = n;
        age = a;
        chronicCondition = cC;
    }
    public String toString() {
        return name + " - " + age ;
    }
```

```
    public int getage() {
        return age;
    }
    public String getchronicCondition() {
        return chronicCondition;
    }

    // define method patientProcessor

}
public class StreamTest {
    public static void main(String[] args) {
        ArrayList<Patient> Patients = new ArrayList<>();
        Scanner sc = new Scanner(System.in);
        for (int i = 0; i < 4; i++) {
            Patient obj = new Patient(sc.next(),
                                sc.nextInt(), sc.next());
            Patients.add(obj);
        }
        Stream<Patient> filteredStream = Patient.patientProcessor(Patients);
        filteredStream.forEach(System.out::println);
        sc.close();
    }
}
```

Solution:

```
import java.util.*;
import java.util.stream.*;
class Patient {
    private String name;
    private int age;
    private String chronicCondition;
    public Patient(String n, int a, String cC) {
        name = n;
        age = a;
        chronicCondition = cC;
    }
    public String toString() {
        return name + " - " + age ;
    }
    public int getage() {
        return age;
```

```
    }
    public String getchronicCondition() {
        return chronicCondition;
    }
    public static Stream<Patient> patientProcessor
                        (ArrayList<Patient> Patients) {

        return Patients.stream()
                .filter(Patient -> Patient.getage() < 30
                && Patient.getchronicCondition().equals("Diabetes"));
    }
}
public class StreamTest {
    public static void main(String[] args) {
        ArrayList<Patient> Patients = new ArrayList<>();
        Scanner sc = new Scanner(System.in);
        for (int i = 0; i < 4; i++) {
            Patient obj = new Patient(sc.next(),
                                sc.nextInt(), sc.next());
            Patients.add(obj);
        }
        Stream<Patient> filteredStream = Patient.patientProcessor(Patients);
        filteredStream.forEach(System.out::println);
        sc.close();
    }
}
```

Public test case 1:
Input:

```
Lavanya 32 Hypertension
Manu 26 Diabetes
Payal 28 Hypertension
Rajini 45 Diabetes
```

Output:

```
Manu - 26
```

Public test case 2:
Input:

```
Amit 25 Diabetes
Kavita 42 Hypertension
Rahul 28 Diabetes
Sonia 50 Hypertension
```

Output:

```
Amit - 25
Rahul - 28
```

Private test case 1:
Input:

```
Varun 30 Migraine
Priya 55 Arthritis
Arjun 40 Hypotension
Naina 28 Diabetes
```

Output:

```
Naina - 28
```

1.2 Session 2 Type 1

Problem Statement

Write a Java program that, given a list of books, prints the title of books whose publication year is between 2000 and 2022 (including 2000, 2022). Otherwise the program raises an exception and prints custom message. Complete the program as specified below.

- Class `PublicationYearOutOfBoundsException` extends the Exception class and should have the following member:
  - Constructor `public PublicationYearOutOfBoundsException(String t)` that takes the title of the book as argument. The constructor, initializes the error message as `"Publication year of <book-title> is outside the acceptable range"`.
- Class `Book` has/should have the following members:
  - Private instance variables `String title` and `int publicationYear`
  - Constructor to initialize these variables
  - Method `checkAndGetTitle` should return the title of the book if the `publicationYear` is within the given limits. Otherwise, it should throw `PublicationYearOutOfBoundsException`.
- Class `ExceptionTest` has the `main` method. It takes the `title` and `publicationYear` of four books as input, and invokes the method `checkAndGetTitle` of class `Book` to produce the specified output.

What you have to do

- Define class `PublicationYearOutOfBoundsException`
- Define method `checkAndGetTitle()` in class `Book`

Template Code

```
import java.util.Scanner;
import java.util.ArrayList;

  //Define class PublicationYearOutOfBoundsException

class Book {
    private String title;
    private int publicationYear;
    public Book(String t, int year) {
        title = t;
        publicationYear = year;
    }
    public String checkAndGetTitle() throws PublicationYearOutOfBoundsException {

  //Complete definition of method checkAndGetTitle
```

```
   }
}
public class ExceptionTest {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<Book> bookList = new ArrayList<>();
        for (int i = 0; i < 4; i++) {
            Book b = new Book(sc.next(), sc.nextInt());
            bookList.add(b);
        }
        for (Book b : bookList) {
            try {
                String title = b.checkAndGetTitle();
                System.out.println(title);
            } catch (PublicationYearOutOfBoundsException pe) {
                System.out.println(pe.getMessage());
            }
        }
        sc.close();
    }
}
```

Public test case 1:
Input:

```
Book1 2010
Book2 2005
Book3 2018
Book4 2015
```

Output:

```
Book1
Book2
Book3
Book4
```

Public test case 2:
Input:

```
Book5 1998
Book6 2025
Book7 2015
Book8 2000
```

Output:

```
Publication year of "Book5" is outside the acceptable range
Publication year of "Book6" is outside the acceptable range
Book7
Book8
```

Private test case 1:
Input:

```
Book7 2015
Book8 2000
Book9 2023
Book10 2008
```

Output:

```
Book7
Book8
Publication year of "Book9" is outside the acceptable range
Book10
```

Solution:

```
import java.util.Scanner;
import java.util.ArrayList;
class PublicationYearOutOfBoundsException extends Exception {
    public PublicationYearOutOfBoundsException(String t) {
        super("Publication year of \"" + t
        + "\" is outside the acceptable range");
    }
}
class Book {
    private String title;
    private int publicationYear;
    public Book(String t, int year) {
        title = t;
        publicationYear = year;
    }
    public String checkAndGetTitle() throws
    PublicationYearOutOfBoundsException {
        if (publicationYear < 2000 || publicationYear > 2022)
            throw new PublicationYearOutOfBoundsException(title);
        return title;
    }
}
```

```
public class ExceptionTest {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<Book> bookList = new ArrayList<>();
        for (int i = 0; i < 4; i++) {
            Book b = new Book(sc.next(), sc.nextInt());
            bookList.add(b);
        }
        for (Book b : bookList) {
            try {
                String title = b.checkAndGetTitle();
                System.out.println(title);
            } catch (PublicationYearOutOfBoundsException pe) {
                System.out.println(pe.getMessage());
            }
        }
        sc.close();
    }
}
```

1.3 Session 2 Type 2 Copy 2 Cloning

Problem Statement

Complete the Java program to create two objects `a1` and `a2` of type `Airplane`. `a2` should be created from `a1` using cloning such that any later changes to `a2` do not affect `a1`.

- Class `Airplane` implements `Cloneable` interface and has/should have the following members:
  - Instance variables `String company`, `eng` of type `Engine`, and `String model`
  - Constructor to initialize the instance variables
  - Mutator methods as needed
  - Overridden method `toString()`
  - Implement method `clone()` that achieves deep copy using cloning
- Class `Engine` implements `Cloneable` interface and has/should have the following members:
  - Instance variables `String name` and `int numEngines`
  - Constructor to initialize the instance variables
  - Mutator methods as needed
  - Overridden method `toString()`
  - Implement method `clone()`
- Class `AirplaneCloneTest` contains the `main` method that takes the inputs and invokes appropriate methods to achieve the functionality.

What you have to do

- Implement method `clone()` in class `Airplane`
- Implement method `clone()` in class `Engine`

```
import java.util.Scanner;
class Airplane implements Cloneable{
  private String company;
  private String model;
  private Engine eng;
  public Airplane(String c, String m, Engine e) {
    company = c;
    model = m;
    eng = e;
  }
```

```
  public String toString() {
    return company+": " + model+eng;
  }
  public void setEngine(String n, int num) {
    eng.setName(n);
    eng.setNumEngines(num);
  }
  public void setModel(String m) {
    model = m;
  }

  // Write code to implement the clone() method

}
class Engine implements Cloneable{
  private String name;
  private int numEngines;

  // Write code to implement the clone() method

  public Engine(String n,int num){
    name = n; numEngines = num;
  }
  public void setName(String n) {
    name = n;
  }
  public void setNumEngines(int n) {
    numEngines = n;
  }
  public String toString() {
    return "[" + name +", "+numEngines+"]";
  }
}
public class AirplaneCloneTest{
  public static void main(String[] args) throws CloneNotSupportedException {
    Scanner sc = new Scanner(System.in);
    Airplane a1 = new Airplane(sc.nextLine(),sc.next(),
                               new Engine(sc.next(),sc.nextInt()));
    Airplane a2 = a1.clone();
    sc.nextLine(); //Last escape character
    a2.setModel(sc.next());
    a2.setEngine(sc.next(),sc.nextInt());

    System.out.println(a1);
    System.out.println(a2);
```

```
    sc.close();
  }
}
```

Public test case 1:
Input:

```
Boeing
747 GE 4
737 RR 3
```

Output:

```
Boeing: 747[GE, 4]
Boeing: 737[RR, 3]
```

Public test case 1:
Input:

```
AirBus
A330 GE 3
A380 Williams 4
```

Output:

```
AirBus: A330[GE, 3]
AirBus: A380[Williams, 4]
```

Private test case 1:
Input:

```
TATA
Indica TataMotors 4
Safari TataMotors 3
```

Output:

```
TATA: Indica[TataMotors, 4]
TATA: Safari[TataMotors, 3]
```

Public test case 1:
Input:

```
Mahindra
Scorpio MahindraMahindra 2
Scorpio MahindraMotors 5
```

Output:

```
Mahindra: Scorpio[MahindraMahindra, 2]
Mahindra: Scorpio[MahindraMotors, 5]
```

Solution:

```
import java.util.Scanner;
class Airplane implements Cloneable{
  private String company;
  private String model;
  private Engine eng;
  public Airplane(String c, String m, Engine e) {
    company = c;
    model = m;
    eng = e;
  }
  public String toString() {
    return company+": " + model+eng;
  }
  public void setEngine(String n, int num) {
    eng.setName(n);
    eng.setNumEngines(num);
  }
  public void setModel(String m) {
    model = m;
  }
  public Airplane clone() throws CloneNotSupportedException {
    Airplane newPlane = (Airplane) super.clone();
    Engine newEng = eng.clone();
    newPlane.eng = newEng;
    return newPlane;
  }
}
class Engine implements Cloneable{
  private String name;
  private int numEngines;
  public Engine clone() throws CloneNotSupportedException {
    return (Engine)super.clone();
  }
  public Engine(String n,int num){
    name = n; numEngines = num;
  }
  public void setName(String n) {
    name = n;
  }
  public void setNumEngines(int n) {
    numEngines = n;
```

```
  }
  public String toString() {
    return "[" + name +", "+numEngines+"]";
  }
}
public class AirplaneCloneTest{
  public static void main(String[] args) throws CloneNotSupportedException {
    Scanner sc = new Scanner(System.in);
    Airplane a1 = new Airplane(sc.nextLine(),sc.next(),
                               new Engine(sc.next(),sc.nextInt()));
    Airplane a2 = a1.clone();
    sc.nextLine(); //Last escape character
    a2.setModel(sc.next());
    a2.setEngine(sc.next(),sc.nextInt());

    System.out.println(a1);
    System.out.println(a2);
    sc.close();
  }
}
```
