# Java 2

2nd Semester
1439-1440

# Quiz 1

**estion 1: Consider the following UML Diagram.**

onsider class ***email*** as given below:

```
ublic class email{
  private String title; //title of the email
  private String sender; //sender of the email
  private boolean spam; // attribute is true if email is spam
ublic email (String t,String d)

title=t;
date=d;

/setters and getters for attributes
```

**Complete Class *emailAccount* as indicated in the comments (8 pt):** *Read method description care*

```
ublic class emailAccount{
/1.attributes
  private String id;
  private String password;
  private int nOe ;
  private email[] inbox;

/2.constructor

ublic emailAccount(String id, String pass){
    this.id = id;
    Password = pass;
    nOe = 0;
    inbox = new email [100];
```

10:36 PM

```
public delete (string title)
/*this method will delete the first email with a given titl[illegible] the inbox ,
the remaining emails are shifted to the left after deleti[illegible]
public markSpam (String sender)
/* this method will change spam to True in all emails in t[illegible]box for
specific sender*/
```

```
// 3. add new email to emailAccount
public void newEmail(email e) {
    if (nOe! = length)
  inbox[nOe++] = new email (e.getTitle() , e.getSender() );

}
```

```
// 4. Copy all emails with specific title from current emailAccount to received
emailAccount, the copied emails should be deleted, the method return the total
number of deleted emails.

public int copyEmail(emailAccount e, String t) {
        for ( int i=0; i<nOe ; i++)
        if(inbox[i].getTitle() .equals (t) )
            e.newEmail(inbox[i]); ← by using newEmail method!
        e.inbox[nOe++] = new Email (inbox[i].getTitle(), inbox[i].getSender());

}
```

**2. Each of the following lines in class *mailSystem* is invalid, correct them based on the comments (2pts.)**

```
public class mailSystem{
public static void main (String args[]){
emailAccount a = new emailAccount("myMail@gmail.com","123456");
                                                password
//add new email to emailAccount a
a.newEmail("meeting","Team Maneger");
email e = new email("meeting", "Team Maneger");
a. newEmail (e);
~~~
```

# Mid 1

**Question1:** Consider the class Item and its subclasses (in page 8) to answer the following Multiple Choices (7 Pts.)

-2.5

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | D | B | C | A | A | D | D | C | D | A | A | C | D |

1. What are the lines where `this` can be deleted without affect code execution?

   A. Class `Medicine` line 12 and 13
   B. Class `Medicine` line 7 and 17
   C. Class `Medicine` line 28 and class `Vitamins` line 7
   D. Class `Medicine` line 7 and class `Vitamins` line 12

2. What are the lines where `super` can be deleted without affect code execution?

   A. Class `Medicine` line 11 and class `Vitamins` line 6
   B. Class `Medicine` line 17 and class `Vitamins` line 11
   C. Class `Medicine` line 28 and class `Vitamins` line 11
   D. Class `Vitamins` line 17 and 18

3. If the implementation of `Vitamins` constructer changed to

```
public Vitamins(String name, int p,
String usage, int w)
{   this.year=w;   }
```

The compiler will produce an error?

   A. True
   B. False

4. If I removed method `computeDose(int age)` from `Medicine` and add it to `Vitamins` the compiler will produce an error because:

   A. `getName()` method is not in `Medicine`.
   B. `Vitamins` class is a direct subclass of `Medicine`.
   C. `Medicine` is not abstract.
   D. `Vitamins` class is not a direct subclass of `Item`.

5. If the implementation of `print()` method in class `Vitamins` changed to

```
public void print()
{
System.out.print(super.name+"
"+super.price+" "+super.usage);
System.out.println(" "+this.year);
}
```

The compiler will produce an error?

   A. True
   B. False

6. Can I change the header of `computeDose()` method in class `Medicine` to

```
public final int computeDose(int
age)
```

   A. Yes
   B. No

7. What is the output of following Java program?

```
public class Test
{public static void main (String[] args)
 {
Item arr[] = new Item [2];
arr [0] =new Item ("Augmentin");
arr [1] =new Medicine ("Panadol",10,"
headache");
for(int i = 0; i<2 ; i++)
System.out.print(arr[i].getName()+" ");
 }}
```

A. Augmentin Panadol
B. null Panadol
C. Augmentin null
D. Error

arr[0] Item Augmentin
arr[1]

Item class abstract class
we can't create object from it

8. What is the output of following Java program?

```
public class Test
{ public static void main (String[]
args)
{
  Vitamins obj =new Vitamins
("Centrum",10,"poor Diet",2019);
  System.out.print(obj);
 }}
```

to string has not overrid in vitamins class

A. Something like Vitamins@2a139a55
B. Name:Centrum price: 10 Usage: poor Diet 2019
C. Name:Centrum
D. Name:Centrum price: 10 Usage: poor Diet

9. What is the output of following Java program?

```
public class Test
{public static void main (String[] args)
 {
Medicine Med =new Medicine
("Panadol",10,"headache");
Vitamins obj= new
Vitamins("Centrum",10,"poor Diet",2019);
obj.merge(Med);
System.out.print(Med.getName()+" "+
obj.getName()); }}
```

Centrum + Panadol

A. Panadol Centrum null
B. Panadol Centrum Panadol
C. null Centrum Panadol
D. Panadol Centrum Panadol

10. What is the output of following Java program?

```
public class Test
{ public static void main (String[]
args)
{
 Vitamins obj =new Vitamins
("Centrum",12,"poor Diet" ,2019);
 Medicine med = new Medicine
("Augmentin",100,"Flo");
obj.getDescription(med);
 }}
```

A. Vitamins: 2019
B. medicine: 0
C. medicine: 2019
D. Error

11. What is the output of following Java program?

```
public class Test
{ public static void main (String[] args)
 {
  Item obj =new Vitamins
("Centrum",10,"poor Diet",2019);
  obj.print();
 }}
```

A. Centrum 10 poor Diet 2019
B. Centrum 10 poor Diet
C. Centrum 10 poor Diet 0
D. Error

12. What is the output of following Java program?

```
public class Test
{ public static void main (String[]
args)
 {
  Vitamins obj =new Vitamins
("Centrum",10,"poor Diet",2019);
System.out.println(obj.computeDose(7)
);
 }}
```

A. medicine: Centrum 10
B. medicine: null 10
C. medicine: null 7
D. Error

13. What is the output of following Java program?

```
public class Test
{public static void main (String[] args)
 {
 Item arr[] = new Item [3];
 arr [0] =new Medicine
("Augmentin",100,"Flo");
 arr [1] =new Medicine
("Panadol",10,"headache");
 arr[2]= new Vitamins("Centrum",10,"poor
Diet",2019);
  for(int i = 0; i<3 ; i++)
   if (arr[i] instanceof Medicine)
System.out.print(arr[i].getName()+" ");
   }}
```

i = 0 → true
i = 1 → true
i = 2 → true

A. Centrum
B. Augmentin Panadol
C. Augmentin Panadol Céntrum
D. Error

14. What is the output of following Java

```
public class Test
{public static void main (Str
args)
{
Item arr[] = new Item [3];
arr [0] =new Medicine
("Augmentin",100,"Flo");
arr [1] =new Medicine
("Panadol",10,"headache");
arr[2]= new
Vitamins("Centrum",10,"poor
Diet",2019);
for(int i = 0; i<3 ; i++)
 if (arr[i] instanceof Vitamins)
System.out.print(arr[i].getUsage
");
  }}
```

A. poor Diet
B. Flo headache
C. Flo headache poor Diet
D. Error → we will cause runtime exception

because the refrence of object is Item and we call method defined only in sub Class (type of object)

## Class Item and its Subclasses for Questions #1

```
public abstract class Item
{
    private String name;
    public Item(String s)
    {
        name = s;
    }
    public String getName()
    {
        return name ;
    }
    public void setName(String n)
    {
        name=n ;
    }
    public String toString()
    {
        return "Name:"+name;
    }
    public abstract void print();
    public abstract int computeDose(int age);
}
```

```
public class Medicine extends Item
{
    public int price;
    private String usage;
    public Medicine ()
    {
        this(null,0,null);
    }
    public Medicine(String name, int p, String usage)
    {
        super(name);
        this.price =p;
        this.usage=usage;
    }
    public void print()
    {
        System.out.print(super.getName()+" "+this.price+"
+this.usage+" ");
    }
    public int computeDose(int age)
    {
        System.out.print("medicine: " + super.getName()+" ");
        if (super.getName().charAt(0)=='\' && age >10)
            return 7;
        return 10;
    }
```

```
    public String toString()
    {
        return super.toString()+" price: "+this.price+" Usage: "
+this.getUsage();
    }
    public String getUsage()
    {
        return this.usage;
    }
    public void setUsage(String u)
    {
        this.usage=u;
    }
}
```

```
public class Vitamins extends Medicine
{
    private int year;
    public Vitamins(String name, int p, String usage, int w)
    {
        super(name,p,usage);
        this.year=w;
    }
    public void print()
    {
        super.print();
        System.out.print(this.year);
    }
    public void merge (Item item)
    {
        Medicine M =(Medicine) item;
        super.setName(super.getName()+" "+ M.getName());
        super.setUsage(super.getUsage()+" "+M.getUsage());
        M.setName(null);
        M.setUsage(null);
    }
    public void getDescription(Item item)
    {
    System.out.println(item.getClass().getName()+": "+
((Vitamins)item).year);
    }
}
```

لو حذفنا super راح تنادي print نفسها للأبد

(infinite call method)

Augmentin , 100, 5

Medicine:

Medicine

Vitamin

Plantation
- name: String
- nOp: int
+ Plantation( name:String, maxNumPlants: int)
+ gardenPlant( Plant p): boolean
+ orgnaicPlants(d: Date): String [ ]
1
-plantList
0....*
<<abstract>>
Plant
- plantId: String
- plantName; String
+ Plant (plantId:String, plantName:String)
+ irregate (): int
// setters- getters
<<abstract>>
Flowers
- blooming: int
+ Flowers( id:String, name:String, bloom: int)
// setters-getters
Fruits
- organic: boolean
+ Fruits(name: String, id: String , org: boolean, d: Date
+ irrigate () :int
//setters - getters
-Harvested_Date
1
1
Outdoor
+Outdoor(id:String, name: String, bloom:int)
+Outdoor( p: Outdoor)
+ irrigate ( ): int
Indoor
+Indoor(id:String, name: String, bloom:int)
+Indoor( p: Indoor)
+ irrigate ( ): int
Date
+Year: int
+Month: int
+day: int
+Date (y:int, M: int, D:int)
+equals(date:Date): boolean

## The description of UML:

### *Plant* Class:

- Attributes:
  - *plantId:* plant id.
  - *plantName:* plant name.
- Methods:
  - *Plant( plantId: String , plantName: String):* constructor.
  - *irrigate( )*(ري): abstract method.

### *Fruits* Class:

- Attributes:
  - *organic:* indicates whether the fruit is organic or not.
- Methods:
  - *Fruit(name: String , id: String, org: boolean, d:Date):* constructor.
  - *irrigate( ):* returns the number of time of irrigation for organic and non-organic fruits.

### *Flowers* Class:

- Attributes:
  - *blooming:* the number of days the flower is bloomed.
- Methods:
  - *Flowers( id: String, name: String , bloom: int):* constructor.

### *Indoor* Class:

- Methods:
  - *Indoor( id: String, name: String , bloom: int):* constructor.
  - *Indoor( p:Indoor):* constructor.
  - *irrigate( ):* returns the number of time that required for indoor plant irrigation.

### *Outdoor* Class:

- Methods:
  - *Outdoor( id: String, name: String bloom: int):* constructor.
  - *Oudoor( p:Outdoor):* constructor.
  - *irrigate( ):* returns the number of time that required for Outdoor plan irrigation.

### *Date* Class:

- Methods:
  - *Date (y:int, M:int, D:int):*constructo
  - *equals (date:Date):* this method returns true if the current date is same as the date sent by parameter.

### *Plantation* Class : (مشتل)

- Attributes:
  - *name:* plantation name
  - *nOp:* the number of plants in the plantation.
- Methods:
  - *Plantation( name: String, maxNumPlants: int):* constructor.
  - *gradenPlant(Plant p):* this method adds a plant (p) to the plantation if possible and returns true. Otherwise, the method will return false.
  - *OrganicPlants (d:Date ):* this method returns an array containing **ONLY** the name of all organic fruits that have been harvested in (تحصد في) the specified date. ***Hint:*** *null elements are not allowed in the returned array.*

1. **Write the class header of Indoor class**

```
Class Indor extends Flowers {
```

2. **Implement the constructor of class Fruits**

```
public Fruits (String name, String id, boolean org, Date d)
{
    super(name, id);
    organic = org;
    Harvested_Date[0] = new Date(d.year, d.Month, d.day);
}
```

+0.25

where did you create the Array!

3. **Translate into Java code the following selected method from class plantation**

```
public boolean gardenPlant( Plant p){
{
        if (nOP >= PlantList.length)
            return false;

        if(p instanceof Fruits)
PlantList[nOP++] = new Fruits(p.getName(), p.getId(),
                    ((Fruits) p).getOrganic(), ((Fruits)(p).getDate

        if (p instanceof indoor)
PlantList[nOP++] = new indoor((indoor) p);

        if (p instanceof outdoor)
PlantList[nOP++] = new outdoor((outdoor) p);

        return true
}
```



code the following selected method from class plantation

```
n gardenPlant( Plant p){
=PlanList.length){
se;
```

```
if(p instanceof Indoor)
Plant List[nop++] = new Indor
((Indoor) p);
```

```
public String [] organicPlants (Date date)
{
        int Cont = 0;
        for (int i=0 ; i< nop ; i++) {
            if (PlantList[i] instanceof Fruits)
        if ((Fruits) PlantList[i].equals(date))
                Cont ++; }
if(Cont != 0){
        String [] nameorg = new String [Cont];
        int j=0;
if (PlantList[i] instanceof Fruits)
        for (int i=0 ; i< Cont ; i++)
            if ((Fruits) PlantList[i].equals(date))
        name org [j++]=PlantList[i].getPlantName();
        return name org; } //end if
}
        return null;
```

get

-0.25

-0.25

-0.5

Har + organic

4. **Write Java code Statements for the application class (main method)**

   a. Create one Plantation called garden with the name "Al Mashtal" that contains a maximum of 10 plants.

```
Plantation garden = new Plantation("Al Mashtal", 10);
```

   b. Add indoor flower plant with the values (Id:"3224", name: "Orchid", blooming: 7) to the garden.

```
Plant in = new indoor("3224", "Orchid", 7);
garden.gardenPlant(in);
```

   c. Print the name of all organic fruits in the garden that harvested in 2019-02-20.

```
String [] orgPlant = garden.organicPlants(new Date(2019,02,20));
for (int i=0 ; i< orgPlant.length ; i++)
System.out.Println(orgPlant[i]);
```

**b.** Add indoor flower plant with the values (Id:"3224", name: "Orchid", blooming: 7) to the garden.

```
garden[0] = new Indoor("3224", "Orchid", 7);
```

-0.25

```
garden - garden.plant(new Indor("3224", "orchid", 7);
```

**c.** Print the name of all organic fruits in the garden that harvested in 2019-02-20.

```
String S[] =
garden.organicPlants(new Date(2019,02,20));
for (int i = 0; i < garden.length; i++){
System.out.println(organicPlants(2019-02-20)); }
```

-0.75

```
for (int i = 0 ; i < S.length ; i++)
S.O.P (S[i]);
```



```
public int price;
private String usage;
public Medicine ()

public Medicine(String name, int p, String usage)

public void print()

public int computeDose(int age)

public String toString()

public String getUsa
```

```
public class Vitamins extends Medicine
{
    private int year;
    public Vitamins(String name, int p, String usage,
    {
        super(name,p,usage);
        this.year=w;
    }
    public void print()
    {
        super.print();
        System.out.print(this.year);
    }
    public void merge (Item item)
    {
        Medicine M = (Medicine) item;
        super.setName(super.getName()+ " " + M.getN
        super.setUsage(super.getUsage()+ " " +M.ge
        M.setName(null);
        M.setUsage(null);
    }
    public void getDescription(Item item)
    {
        System.out.println(item.getClass().getName()
((Vitamins)item) .year);
    }
```

# Quiz 2

unchecked runtime ← myEx1

Exc ← myEx2 checked

**Question 1: Consider the following Java program.**

```
class MyEx1 extends RuntimeException{
public MyEx1(String m){
super(m);}}

class MyEx2 extends Exception{
public MyEx2(String m){
super(m);}}

public class TestMyProgram{
static int i = -4;
public static void main(String args[]){
try{
   System.out.println(M2());
   i = 5;
   System.out.println(M3());}
   catch(Exception e){
   System.out.println("catch in main");
   System.out.println("Bye Bye");}
}

public static int M1(int a) throws MyEx2{
if(a % 2 == 0)
   throw new MyEx1 ("It is even");
else if (a <5)
   throw new MyEx2("Odd");
return 100;
}

public static String M2(){
try {
i = M1(i);
if(i >0)
   throw new MyEx2("It is Positive");
return "try";}
catch(MyEx2  exp){
   System.out.println(exp.getMessage() + " catch in M2"); }
catch(MyEx1  exp){
   System.out.println(exp.getMessage()+ " catch in M2"); }
finally{
   System.out.println("finally of M2 "+ i);}
return "Catched"; }

public static int M3() {
int num;
try{
num = Integer.parseInt(M2());}
catch (NumberFormatException exp) {
   System.out.println("not a numeric String ");
   throw new MyEx1("Exception One");}
return num;}
}//end class
```

-4

i = -4 5

i = -4

it is even catch
finally of M2 -4
Catched
It is positive catch
finally of M2.
not numeric String
Catch in main
Bye Bye

Output:

It's even catch in M2
finally of M2 -4
Catched
Its Positive catch in M2
finally of M2 ...
not numeric String
Catch in main
Bye Bye.

num

Catched

It is Positive catch in M2
finally of M2 100
not numeric string

catch in main
Bye Bye.

**2. Consider the code from Question1, state whether the following statements are true or false (2.5 pts)**

**a)** MyEx2 exception is a checked exception. (T / F)

**b)** If we remove `catch (MyEx1 exp)` from M2 method, the compiler will produce a compilation error.
(T / F) because there's Finally.

**Question 2: Mention if the following code fragments are legal or illegal; explain your choice. (2 pts):**

| Code Fragment | Legal/Illegal | Explanation |
|---|---|---|
| `public interface one{`<br>`public void A();}`<br><br>`public interface two extends one{`<br>`public double B();`<br>`private void C();}`<br>`Public void A();` | illegal | interface does not accept private methods |
| `public interface A{`<br>`public void x();}`<br><br>`public interface B{`<br>`public double y();`<br>`public void z();}`<br><br>`public class AB implements A, B{`<br>`public double y(){`<br>`//Implements the body of y`<br>`}`<br>`public void x(){`<br>`//Implements the body of x`<br>`}`<br>`}` | illegal | a class that implements two interfaces is legal But in class AB did not implement method z() that's why it's illegal |

# Mid 2

| KING SAUD UNIVERSITY<br>COLLEGE OF COMPUTER AND INFORMATION SCIENCES<br>COMPUTER SCIENCE DEPARTMENT | | |
|---|---|---|
| **CSC 113: Computer Programming II** | **Midterm 2 (Duration: 1:50)** | **2nd Semester 1439-1440** |

| Student Name (Arabic) | Student ID | Section Number | Serial Number |
|---|---|---|---|
| | | | |

**Question1:** **Consider the following Java program** **(6 Pts.)**

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

class Ex2  extends FileNotFoundException {
public Ex2 (String m) {super(m); } }

public class Mid2 {

public static void main(String args[]) {
      Scanner scan =new Scanner (System.in);
try{
System.out.println("Lets go:");
int x=scan.nextInt();
int y=scan.nextInt();
         String z=scan.next();
System.out.println("Calling the methods:");
isThere(z);
         M1 (x,y);
         String arr[] = new String [2];
arr[0] = z;
System.out.println(arr[1].toString());
System.out.println("See you soon");
      }
catch(InputMismatchException e){
System.out.println("Catch1:InputMismatchException");}
catch(ArithmeticException e){
System.out.println("Catch 2: ArithmeticException");}
catch(RuntimeException e){
System.out.println("Catch 3:RuntimeException");}
catch(Exception e){
System.out.println("Catch 4: " +e.getMessage());}
finally{ System.out.println("Finally of main");}
   }

public static Boolean isThere (String s)throws IOException
   {System.out.println("Testing the file");
if (!s.equals("file.data"))
throw new Ex2("Not there");
System.out.println("File");
return true;}

public static void M1 (intx,int y)throws Exception
   {
try{
try{
int r=x/y;
            }
catch (NumberFormatException e)
         {System.out.println(e);}
      }
finally{ System.out.println("Finally of M1");}
System.out.println("End of method M1");
   }
}
```

**Assume the user's input for the three declared variables x, y, and z will be as shown in the following table, show the program output for each user input and if an exception is thrown at what line:**

| X | Y | Z | Output | The exception is thrown at line number |
|---|---|---|---|---|
| 1 | two | file.data | Lets go:<br>Catch1:InputMismatchException<br>Finally of main | 14 |
| 2 | 4 | Text.data | Lets go:<br><br>Calling the methods:<br>Testing the file<br>Catch 4: Not there<br>Finally of main | 38 |
| 1 | 0 | file.data | Lets go:<br><br>Calling the methods:<br>Testing the file<br>File<br>Finally of M1<br>Catch 2: ArithmeticException<br>Finally of main | 46 |
| 12 | -2 | file.data | Lets go:<br><br>Calling the methods:<br>Testing the file<br>File<br>Finally of M1<br>End of method M1<br>Catch 3:RuntimeException<br>Finally of main | 21 |

**Question2:Answer the following Multiple Choices** **(4 Pts.)**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| A | A | C | B | C | B | B | A |

1. Considering the code from Q1, removing `throws Exception` from the header of M1 will Not affect the code?

   A. True
   B. False

2. An *Interface*is a reference type, so we can write a method that accept a parameter of type interface.

   A. True
   B. False

3. What is the output of following Java program?

```
Public class Test{
Public staticvoid main (String[]
args){try
{
System.out.printf("1");
int sum = 9 / 0;
System.out.printf("2");
   }
catch(ArithmeticException e)
{ IOException obj = new IOException ("Oops");
            try
            {
                throw obj;
            }
            catch(Exception e1)
            {
                System.out.printf("8");
            }
}
finally
{ System.out.printf("3"); }
System.out.printf("4");
}
}
```

   A. Compilation error
   B. Runtime error
   C. 1834
   D. 134

4. What is the output of following Java program?

```
import java.io.*;
public class Files
{
Public static void main(String
args[])
{
File obj = new File("/java/system");
System.out.print(obj.getName());
}
}
```

   A. java
   B. system
   C. java/system
   D. /java/system

5. Which of these exceptions is thrown in cases when the file specified for reading is not found?

   A. IOException
   B. FileException
   C. FileNotFoundException
   D. FileInputException

6. After an exception is handled, program control returns to the throw point to complete its normal execution?

   A. True
   B. False

| | |
|---|---|
| 7. Binary files are human readable in text editors?<br><br>A. `True`<br>B. `False` | 8. `public class one extends two implements three,four`<br>is a correct class header ?<br>A. `True`<br>B. `False` |

7. Binary files are human readable in text editors?

A. True
B. False

8. public class one extends two implements three,four
is a correct class header ?

A. True
B. False

**(10 Pts.)**

**Question3:**

**Consider the UML and the corresponding classes' descriptions.**

**The description of UML:**

**Professor Class:**

9.75

- Attributes:
  - *name:*Name of the professor.
  - *ID:*Employment ID of the professor.
  - *nOp:* the number of publications for a specificprofessor.
- Methods:
  - *Professor(name:String, ID:int, size:int):*constructor.
  - *addPublication(p: Publication):* this method adds the receivedpublication*p* to the professor's publication list if possible.
  - *readFile (fName: String):* this method reads the publications, object by object from thereceived binary file *fName*, and then inserts them into the publication list*pList*.
  - *writeFile(fName: String, keyword: String):* this method writes into the received text file *fName* the publications information. Each publication should be written in twoseparate String lines as follows:
    - The first line should contain the publication type, which is specified as Book or Paper.
    - The second line should include the title and the year of the publicationonly.

    Please note that this method should **ONLY** write into the file the publications that follows the below conditions:

    - Book publications: should contain the received keyword in its title.
    - Paper publications:the keyword appeared in its title or keyword list at least 3 times.
  - *calculatePoints(fName: String):* This method will perform two tasks:
    - Write the most recentlypublished books of the professor in the received binary file. The written book publications' informationshould include onlytwo attributes, the publishing year and the ISBN.
    - In addition, the method should calculate the number of recently published papers and books by the professor; and compute and return the deserved promotion points accordingly. The professor should get 3 points on every book published and 2 points on every paper publishedrecently.
  - *recentPublications ():* this method returns an array containing **ONLY** the object of all publications that havebeen published inthe last three years.

***Publication*Class:**

- Attributes:
  - *title:*the title of the publication.
  - *year:* the year of the publication.
- Methods:
  - *Publication(title: String, year: int):*constructor

***PaperClass:***

- Attributes:
  - *issue* the issue number of the journal.
  - *keywords*: a list of 5 keywords that describe the publication.
- Methods:
  - *Paper(title: String, year: int, issue: int, keywords: [] String)*:constructor.
  - *searchKeyword (word: String)*: this methodchecks whether a keyword is contained either in the title of the paper or in the list of the keywords, and then returns the total number of its occurrence.

***BookClass:***

- Attributes:
  - *ISBN*: the ISBN number of the book.
- Methods:
  - *Book(title: String, year: int, isbn: String)*:constructor.
  - *searchKeyword (word: String)*: this method searches for a keyword in the book title and returns true if the keyword is found, otherwise, it returns false.

**...nslate into Java code the following selected methods from class Professor.Consider all ...ters as available.**

**...ing methods.**

7. Binary files are human readable in text editors?

A. True
B. False

8. public class one extends two implements three, four
is a correct class header ?

A. True
B. False

**(10 Pts.)**

**Question3:**

**Consider the UML and the corresponding classes' descriptions.**

throws

**The description of UML:**

**Professor Class:**

- Attributes:
  - *name:*Name of the professor.
  - *ID:*Employment ID of the professor.
  - *nOp:* the number of publications for a specificprofessor.

- Methods:
  - *Professor(name:String, ID:int, size:int):*constructor.
  - *addPublication(p: Publication):* this method adds the receivedpublication*p* to the professor's publication list if possible.
  - *readFile (fName: String):* this method reads the publications, object by object from thereceived binary file *fName*, and then inserts them into the publication list*pList*.
  - *writeFile(fName: String, keyword: String):* this method writes into the received text file *fName* the publications information. Each publication should be written in twoseparate String lines as follows:
    - The first line should contain the publication type, which is specified as Book or Paper.
    - The second line should include the title and the year of the publicationonly.

    Please note that this method should **ONLY** write into the file the publications that follows the below conditions:

    - Book publications: should contain the received keyword in its title.
    - Paper publications:the keyword appeared in its title or keyword list at least 3 times.
  - *calculatePoints(fName: String):* This method will perform two tasks:
    - Write the most recentlypublished books of the professor in the received binary file. The written book publications' informationshould include onlytwo attributes, the publishing year and the ISBN.
    - In addition, the method should calculate the number of recently published papers and books by the professor; and compute and return the deserved promotion points accordingly. The professor should get 3 points on every book published and 2 points on every paper publishedrecently.
  - *recentPublications ():* this method returns an array containing **ONLY** the object of all publications that havebeen published inthe last three years.

***Publication*Class:**

- Attributes:
  - *title:*the title of the publication.
  - *year:* the year of the publication.

- Methods:
  - *Publication(title: String, year: int):*constructor

***Paper*Class:**

- Attributes:
  - *issue:*the issue number of the journal.
  - *keywords:* a list of 5 keywords that describe the publication.

- Methods:
  - *Paper(title: String, year: int, issue: int, keywords: [] String):*constructor.
  - *searchKeyword (word: String):* this methodchecks whether a keyword is contained either in the title of the paper or in the list of the keywords, and then returns the total number of its occurrence.

***Book*Class:**

- Attributes:
  - *ISBN:* the ISBN number of the book.

- Methods:
  - *Book(title: String, year: int, isbn: String):*constructor.
  - *searchKeyword (word: String):* this method searches for a keyword in the book title and returns true if the keyword is found, otherwise, it returns false.

**Translate into Java code the following selected methods from class Professor.Consider all the getters and setters as available.**

**Note:**

- **It is important that you make use of existing methods.**
- **Handle any exception caused by incorrect file processing.**

**a.** Implement the *readFile* method of class professor.

```
import java.io.*;
public void readFile (String fName) {throws IOException, ClassNotFoundException}
File f=new File(fName);
ObjectInputStream o = new ObjectInputStream(new FileInput
Stream(f)); publication p;
try {
  while(true){
  p=(publication) o.readObject();
  addPublication(p); }
  catch(EOFException e) { o.close(); } }
}
o.close(); }
```

**b.** Implement the *writeFile* method of class professor.

```
import java.io.*;
public void writeFile(String fName, String keyword) throws IOException {
File f = new File(fName); PrintWriter pw = new PrintWriter(f);
for(int i=0; i<nOP; i++) {Publication P = PList[i];
if(P instance of Book) {if(P.getTitle().contains(keyword)) {
pw.println("Book"); pw.println(P.getTitle()+" "+P.getYear());}}
if(P instance of Paper) {if(((Paper)P).searchKeywords(keyword) > 2){
pw.println("Paper"); pw.println(P.getTitle() + " " + P.getYear()); }}
pw.close();}
```

**c.** Implement the *calculatePoints* method of class professor.

```
import java.io.*;
public int calculatePoints(String fName) throws IOException {
Publication[] P = recentPublications();
DataOutputStream o = new DataOutputStream(new FileOutputStream(new File(fName)));
int PC = 0; int BC = 0;
for(int i=0; i<P.length; i++) { Publication P1 = P[i];
if(P1 instance of Book) BC = BC + 3;
else if(P1 instance of Paper) PC = PC + 2;}
o.close();
return PC + BC;
for(int i=0; i<P.length; i++) {
Publication P2 = P[i];
if(P2 instance of Book) {
o.write((Book)P).getISBN()); ;}
```