CITY UNIVERSITY OF HONG KONG

Course code & title : CS2312 Problem Solving and Programming

Session : Semester A 2022-23 **Midterm**

Time allowed : 90 Minutes

This paper has **12** pages (including this page).
Page 5-6 are your scratch paper.

This paper consists of **4** questions.

Question	1	2	3	4	Total
Marks					
Max	20	10	20	25	75

Please provide your answers in pages **7-12**.
You should submit pages **7-12** only.

Remarks:

1. For all written code required by the questions, you should give precise JAVA code with proper programming styles. Marks may be deducted for redundant or unnecessary code.
2. Apply best practices and skills of **object-oriented programming**.
3. Pay attention to **code design and naming**. **Program execution speed** is **NOT** a major concern**.**
   No need to handle input errors or exceptions unless explicitly mentioned in the question.
4. If your answer is incomplete, you can still get partial marks based on the quality of the answer. Note: Do not remove or change any given code.
5. Whitespaces and line breaks in the output are unimportant.

*This is a **closed-book** examination.*

No materials or aids are allowed during the whole examination.
If any unauthorized materials or aids are found on a candidate during the examination, the candidate will be subject to disciplinary action.

Question 1. (20 marks)

The following program models the **companies** who launch **products** (e.g. cell phones, watches, calculators) and the **salespersons** who sell the products and earn commission points from the companies.

- Each product is launched by one company only. A salesperson earns 1 point every time when he sells a product.
- **main()** is given below. Its outputs are given in the underlined comments.
  The **Company** class is also given below. It is extremely simplified, but is sufficient for this program.
- Your tasks: write the **Product** class and the **SalesPerson** class

```
public static void main(String[] args) {
    Company a = new Company();
    Company b = new Company();
    Company c = new Company();

    Product p1 = a.launch();
    Product p2 = b.launch();
    Product p3 = c.launch();
    Product p4 = c.launch();

    SalesPerson helena = new SalesPerson();
    SalesPerson paul = new SalesPerson();

    helena.sell(p1); //earn one commission point from company a
    helena.sell(p3); //earn one commission point from company c
    helena.sell(p3); //earn one commission point from company c
    helena.sell(p4); //earn one commission point from company c

    helena.showEarnFrom(a); //Output: 1 points
    helena.showEarnFrom(b); //Output: 0 points
    helena.showEarnFrom(c); //Output: 3 points
    paul.showEarnFrom(a); //Output: 0 points
    paul.showEarnFrom(b); //Output: 0 points
    paul.showEarnFrom(c); //Output: 0 points
}
```

```
public class Company
{
    public Product launch() {
        return new Product(this);
    }
}
```

Question 2 (10 marks) What are the outputs from statements I-V below?

```
public class A {
   public int value=500;

   public int getValue() {
       return value;
   }

   public int algorithm() {
       return this.getValue()-this.value;
   }
}
```

```
public class B extends A {
   public int value=300;

   public int getValue() {
       return value;
   }

   public int algorithm() {
       return super.algorithm()*2;
   }
}
```

```
public class Main {

   public static void main(String [] args)  {

       A x = new B();
       System.out.println(x.value);             //Statement I
       System.out.println(((B)x).value);        //Statement II

       System.out.println(x.getValue());        //Statement III
       System.out.println(((B)x).getValue());   //Statement IV

       System.out.println(x.algorithm());       //Statement V
   }
}
```

Question 3. (20 marks)

We are to analyze some numeric data stored a file. For simplicity, only the average is reported, as shown below:

Test cases:	*Expected Outputs:*

Case 1: The file *marks.txt* has 2 marks: `40` `80`.	`Average is: 60`
	`Report finished without error.`
	`File closed.`

Case 2: The file *marks.txt* exists but is empty.	`Content is empty!`
	`File closed.`

Case 3: The file *marks.txt* does not exist.	`Cannot open marks file!`

(a) Mary has written the program below. However, she did not make use of Exception Handling correctly. What will be the outputs from her program? Choose the answers in the answer sheet. (9 marks)

(b) Mary's program needs to be corrected and improved. Rewrite it with proper code. (11 marks)
Note: To write less, you may write the line numbers of below in your answer instead of copying the code. The program outputs should be the same as the *Expected Outputs*.

```
[1]  public static void main(String[] args) {
[2]      ArrayList<Integer> allMarks = new ArrayList<>();
[3]
[4]      try {
[5]
[6]          //Open file
[7]          Scanner f = new Scanner(new File("marks.txt"));
[8]
[9]          //Read marks into arraylist
[10]         while (f.hasNext()) {
[11]               try {
[12]                 int mark = f.nextInt();
[13]                 allMarks.add(mark);
[14]               } catch (InputMismatchException e) {
[15]                 System.out.println("Wrong data.  Program will stop!");
[16]               }
[17]         }
[18]
[19]         //Reporting
[20]         analyzeMarks(allMarks);
[21]
[22]         //Close file
[23]         f.close();
[24]         System.out.println("File closed.");
[25]
[26]     } catch (FileNotFoundException e) {
[27]         System.out.println("Cannot open marks file!");
[28]     }
[29]
[30]     System.out.println("Report finished without error.");
[31] }
[32] //-----------------------------------------------------------------
[33] private static void analyzeMarks(ArrayList<Integer> marks) {
[34]     try {
[35]         if (marks.size() == 0)
[36]               throw new ContentIsEmpty();
[37]
[38]         int sum = 0;
[39]
[40]         for (Integer m: marks)
[41]               sum += m;
[42]         System.out.println("Average is: " + sum/marks.size());
[43]
[44]     } catch (ContentIsEmpty e) {
[45]         System.out.println(e.getMessage());
[46]     }
[47] }
```

```
public class ContentIsEmpty extends Exception {
    public ContentIsEmpty() {super("Content is empty!");}
    public ContentIsEmpty(String msg) {super(msg);}
}
```

<u>Question 4. (25 marks)</u>

The following program creates objects of folders, subfolders, and files in a hard disk, in which the root of the drive is also treated as a folder. These objects are created as instances of the classes `Folder` and `File`, which are subclasses of the abstract class `Item`. The `FileSystem` class is a **Singleton**. It keeps track of all files and folders using an `ArrayList` of Items.

Study the explanation and outputs in the comments below. The outputs are given in the <u>underlined comments</u>:
- The `getMyPath()` method of `Item` is used to obtain the path of a folder or a file;
- The `search()` method of `FileSystem` is used to search for a file or a folder with a given name. If two or more hits are found, all results are shown (the ordering is unimportant). For each found folder, show the path; for each found file, show the path and the file size.

<u>Your task</u>: Finish all missing code in the answer paper.

<u>Note:</u> <u>Do not use the **`instanceof`** operator</u>. Otherwise you may get <u>0 mark</u> for this question!

<u>Reminder</u>: Read ALL given code carefully! Don't remove or change any given code!
Apply `@Override` where appropriate.

```
public static void main(String[] args) {

   Folder root = new Folder("C:");                // Create the root object for drive C; it has no parent folder
   Folder d1 = new Folder(root,"personal");       // Under the folder root, create the personal folder as d1
   Folder d2 = new Folder(root,"study");
   Folder d3 = new Folder(d2,"cs");               // Under the folder d2, create the cs folder as d3
   Folder d4 = new Folder(d3,"lab01");
   Folder d5 = new Folder(d3,"lab02");

   File x1 = new File(d1,"job.docx", 5008);       // Under the folder d1, create file job.docx (5008 bytes) as x1
   File x2 = new File(d4,"main.java",834);
   File x3 = new File(d4,"food.java",705);
   File x4 = new File(d5,"main.java",1234);

   System.out.println(d5.getMyPath());            // Output: C:\study\cs\lab02
   System.out.println(x1.getMyPath());            // Output: C:\personal\job.docx

   FileSystem fs=FileSystem.getInstance();

   fs.search("main.java");        // Output: C:\study\cs\lab01\main.java (834 bytes) C:\study\cs\lab02\main.java (1234 bytes)
   fs.search("lab02");            // Output: C:\study\cs\lab02

}
```

--- End of Questions ---

[You may use this blank paper as your scratch paper]

[You may use this blank paper as your scratch paper]

Q1.

```
public class Product{

}
```

```
import java.util.ArrayList;

public class SalesPerson {

}
```

Q2. What are the outputs from statements I-V?

I : ________ II : ________ (3 marks for I and II)

III : ________ IV : ________ (3 marks for III and IV)

V : ________ (4 marks)

Optional: You may add a drawing for explanation of V in the space below.
Extra marks may be given for your drawing <u>if your answer for V is wrong</u>.

Q3. (a) For each case, choose your answer from the choices: A, B, or C. (9 marks)

Case 1: The file *marks.txt* contains 2 marks:

```
40
80
```

. Your answer: ________

A.
```
Average is: 60
Report finished without error.
```

B.
```
Average is: 60
File closed.
Report finished without error.
```

C.
```
Average is: 60
```

Case 2: The file *marks.txt* exists but is empty. Your answer: ________

A.
```
Content is empty!
Report finished without error.
```

B.
```
Content is empty!
File closed.
Report finished without error.
```

C.
```
Content is empty!
```

Case 3: The file *marks.txt* does not exist. Your answer: ________

A.
```
Cannot open marks file!
Report finished without error.
```

B.
```
Cannot open marks file!
Content is empty!
File closed.
Report finished without error.
```

C.
```
Cannot open marks file!
```

Q3. (b) (11 marks)

```
public static void main(String[] args) {
       ArrayList<Integer> allMarks = new ArrayList<>();


}
-----------------------------------------------------------------
private static void analyzeMarks(ArrayList<Integer> marks)
{


}
```

```
public class ContentIsEmpty extends Exception {
    public ContentIsEmpty() {super("Content is empty!");}
    public ContentIsEmpty(String msg) {super(msg);}
}
```

Q4

```
import java.util.ArrayList;

public class FileSystem {













    private ArrayList<Item> allItems;

    public void search(String name) { Item.search(allItems,name); }
    public void addItem(Item item) { allItems.add(item); }

}
```

```
import java.util.ArrayList;

public abstract class Item {

    private String name;
    private Folder parent;

    public Item(Folder parent, String name) {
        this.parent = parent;  this.name = name;  FileSystem.getInstance().addItem(this);
    }

    protected Folder getParent() { return parent; }
    protected String getName() { return name; }
```

```
public class Folder extends Item {













}
```

```
public class File extends Item {













}
```

[You may add your answers for any question on this page.]

--- End of Paper ---