COSC2531 Programming Fundamentals

Assignment 2: PlayStore - Mobile Applications Management system

Introduction

This is a group assignment and worth 20% of your final grade (5% demonstration, 15% final submission). The grading is based on both demonstration and final code submission. Each group may consist of 1-3 students. You must demonstrate your assignment in your timetabled practical class prior to your final submission.

Objective

The main objective of this assignment is to familiarize you with object oriented design, programming, testing, and refactoring. Object oriented programming helps to solve complex problems by coming up with a number of domain classes and associations. However identifying meaningful classes and interactions requires a fair amount of design experience. Such experience cannot be gained by classroom-based teaching alone but must be gained through project experience.

The different stages of this assignment are designed to gradually introduce different concepts such as inheritance, abstract classes, method overloading, method overriding, and polymorphism.

Assessment

Demonstration

The demonstration is worth 5% of the final grade (25% of the weight of Assignment 2). The whole assignment group must be present to demonstrate the assignment in a timetabled practical class (tute-lab) prior to the final submission. If members come from different classes, please choose just one of the classes to make the demonstration. Subject to on-time final submission and plagiarism detection, you will receive the preliminary marks for your demonstration. The code should follow the style guidelines and have proper comments. You need to demonstrate it in any tute-lab scheduled during week 11 or week 12, but you can do it only once.

You are to demonstrate in the lab that the majority of your team members are enrolled in. Don't expect that all of the marking will happen in the last week (as many people do). **If we run out of time in week 12 your demonstration will not be assessed**; we advise you to get it marked in week 11 if possible.

If you do not demonstrate within week 12 you will receive marks only on the final submission code.

Final submission

The final submission is worth 15% of the final grade (75% of the weight of Assignment 2), due on week 12 (12 Oct, 2018, 23:59 pm). Your final submission through Canvas should be a zip file that includes the following files –

1. The java files of your assignment. Do not submit the .class file.
2. A .txt file that contains the name and student ID of each member of your group.
3. (Optional) If you are unable to complete the code, please also submit a word or pdf document file with the brief description of problems encountered and lessons learnt. If you have not received full marks during the demonstration and then made any change in your code afterwards, submit a word or pdf document file with the brief description of changes as well.

Assessment criteria

Assessment Task	**Marks**
Group A	4 marks
Group B – F	2.5 marks (0.5 mark each)
Functionality 1	1 marks
Functionality 2	3 marks
Functionality 3	1 marks
Functionality 4 - 6	3 marks (1 mark each)
Functionality 7	2 marks
Functionality 8	1 marks
Others 1. Correctness / Test cases & results 2. Code quality / comments	3.5 marks
3. Modularity / Use of classes, inheritance, polymorphism, methods & arguments 4. Clear Logic / Strategy explained 5. Reflection / Lessons learnt	

Assignment overview

A PlayStore is a standalone digital marketplace that allows users to browse and download mobile phone applications. The PlayStore also serves as a digital multimedia store offering music, magazines, books, movies and television programs. Applications and multimedia items in the PlayStore are either free or can be bought for a price.

The program you create will allow the creation of a store, filling it with products, creating users and simulating their interaction with the store (purchasing products, adding comments etc).

Assessment tasks

Your program should consist of multiple class files where you can demonstrate your knowledge of inheritance, polymorphism, method overriding, abstract classes, etc. You need to write classes, add methods and variables to complete the following tasks performed by the admin of the PlayStore. ***There are two sample/starter classes (PlayStoreMain.java and PlayStore.java) provided.***

Section 1: The classes and their attributes

Group A

Class `Content`

The mobile phone apps and the multimedia items are `Content` of the PlayStore. Each `Content` (either application or multimedia) is associated with the following information: an ID, application name, number of downloads, price, and reviews. Reviews is a collection of `Comment` objects (see Task B for details). **A object of type `Content` cannot be created (only subclasses).**

Class `Game`

Game is a type of Content. In addition to the data that a content class have, a Game object has a Boolean variable called isMultiPlayer, and an OS type of object that presents the minimum operating system requirement (See Task C for details on the OS class) for the game. A false value for the variable isMultiPlayer means it's a one player game. A Game type of object can be initialized as

```
Game g1 = new Game("g1", "Pokémon", 5, false, androidV4);
```

Here 10 is the price of the game in dollars, it is a one player game, and androidV4 is the OS type of object (details in Task C). Initially the number of downloads is zero, and the reviews are empty.

Class Reading

Another type of Content is Reading. In addition to the data that the Content class has, a Reading object also has: publisher, genre, and the number of pages. **A object of type Reading cannot be created.**

Class Book

A type of Reading is Book, where the additional data is a collection of strings for the authors' name. A Book object can be initialized as

```
String[] authors1 = {"L. Tolstoy"};
Book b1 = new Book ("r1", "War and Peace", 12,
                    "The Russian Messenger", "Novel",
                    1225, authors1);
```

Here the book title "War and Peace" is the name of the book, 12 is the price of the book in dollars, "The Russian Messenger" is the publisher, "Novel" is the genre, 1225 is the number of pages, "L. Tolstoy" is the author.

Class Magazine

Another type of Reading is Magazine, where the additional data is the title of the main feature. A magazine does not contain any author's name. A Magazine object can be initialized as

```
Magazine m1 = new Magazine("r3", "Forbes", 8, "Forbes Media",
               "Business", 50, "World's richest under 30");
```

Here the name of the magazine "Forbes" is stored as the application name, 8 is the price in dollars, "Forbes Media" is the publisher, "Business" is the genre, 50 is the number of pages, and "World's richest under 30" is the title of the main feature.

Group B

Class Comment

Write a Comment class that has the following data: a User type of object, which is the user who wrote the comment (See Task D for details on User), a string for the comment, and a collection of Comment objects as the reply of the comment. A Comment object can be initialized as

```
Comment comment1 = new Comment(u1, "This is a fantastic game!");
```

Class OS

Write a OS class that contains the type of mobile operating system (e.g., Android or iOS), and the version number. For simplicity, assume the version number is an integer. An OS type of object can be initialized as:

```
        OS androidV4 = new OS("Android", 4);
```

Class User

Write a User class, where each user has an ID, a name, a phone number, a balance, and an OS object. A User can be initialized as:

```
        User u1 = new User("u1", "John Doe", "0412000", 2000, androidV4);
```

Class PlayStore

You have been provided a PlayStore class outline.

The PlayStore class will need to have two attributes: a collection of Content and a collection of User objects (you are encouraged to use Java collections like ArrayList and HashMap). Note that each content can be uniquely identified based on content ID (please see the use of java collections HashMap below for details on this requirement).

An instance of the PlayStore class named store is created in the main method of PlayStoreMain. The interaction with this store is simulated within the main method (see the PlayStoreMain.java class).

Section 2: Functionalities of the classes

User functionalities

1. Method becomePremium. A user can become a Premium user for a cost of $100. A premium user gets 20% discount on the price of each of the contents she buys *after* becoming premium.
2. Method buyContent, where the parameter is a Content type of object. When a user buys any content, the price of that content needs to be deducted from the balance of that user. Do necessary checks before the deduction. You need to consider whether the user is a premium user or not in this step. The number of downloads of the content also increases by one when a user buys it.
   a. Exceptions must be thrown when an attempt is made to buy a content without having sufficient balance. The exception thrown must indicate the cause of error, show an appropriate error message, allowing the caller to respond appropriately and recover if possible.

   b. If the content of buyContent method is a Game type of object, you need to first check the compatibility of the OS of the user, and the minimum OS requirement of the game. If the operating system of the user is “Android”, and the OS requirement of the game is “iOS” (i.e., the operating systems are not the same), an appropriate exception must be thrown. If the version number of the user’s OS is less than the version number of the required OS of the game, an appropriate exception must be thrown. The thrown exceptions must indicate the cause of error, show an appropriate error message, allowing the caller to respond appropriately and recover if possible.

      As an example, let the OS of the game is “Android”, 5 and the OS of the user is “Android”, 4. Although the operating systems are the same, the user’s version number is less than the minimum required version number of the game, so the user cannot buy this game.

c. A user may buy multiple content. Write a method `showContentBought` in the `User` class to show the list of names of all the contents that the user has bought. You may add additional attributes in the `User` class if necessary.

d. Note that when you add exceptions the method calls will need to be surrounded by try/catch blocks.

`Content` and `Comment` functionalities

3. Write a method of the `Content` class, where a review (which is a `Comment` type of object) from a user can be added to a content type of object.
4. Write a method of the `Comment` class to add a reply (which is a `Comment` type of object) to a specific comment.

   As an example of adding reviews and replies from the main method, see the following code snippet (which can also be found in the main method provided in the Canvas)

```
Comment comment1 =
       new Comment(u1,"This is a fantastic game!");
g1.addReview(comment1);

Comment reply1 =
       new Comment(u2, "I never liked this game!");
comment1.addReply(reply1);

Comment reply2 =
       new Comment(u1, "Why not??");
reply1.addReply(reply2);

Comment comment2 =
       new Comment(u2, "The game crashes frequently");
g1.addReview(comment2);
```

5. Write a method `showReview` in the `Content` class to show all the reviews of a `Content` object (e.g. a particular game or book). If a review has replies to its comments, show them as well.

   Bonus point: If you can show the replies of the comments with an indentation at the beginning of each reply, you will receive a bonus 1 mark. Sample output is shown below…

```
John Doe (u1): This is a fantastic game!
      Jane Doe (u2): I never liked this game!
            John Doe (u1): Why not??
Dave Roe (u3): The game crashes frequently.
```

`PlayStore` and `Admin` functionalities

6. Write a method `showContent` of the `PlayStore` class to show a list of all available contents. Also write a method for each type of contents to show the list of contents of that

type (e.g., show all games, show all books, show all magazines). Do you need to write a method for each type, or you can use the same method for this task? (Hint: You may find Java `getClass()` method in `java.lang.Object` useful).

7. Write a method of the `PlayStore` class to show the list of all `Reading` type of objects with a given genre. The parameter of this method should be a `String`, representing the genre (for example, “Novel”).

Use of Java Collections

8. You are encouraged to use collections such as `ArrayList` and `HashMap`. `ArrayList` implements an array which can grow indefinitely. `HashMap`s allow an association to be created between keys and objects. Using such classes also reduces the amount of code required as they provide methods for retrieving required objects easily.

```
ArrayList<Comment> reviews = new ArrayList<Comment>();
reviews.add(someComment);
```

To extract the 4th element

```
Comment c4 = reviews.get(3); // index starts at 0
```

You can use `HashMap` for storing objects, which have unique primary keys. For example,

```
HashMap<String, Content> contents = new HashMap<String, Content>();
```

To add a content we use :

```
contents.put(new Game(…));
```

To extract the content, use:

```
content content1 = contents.get("G101");
// returns null if no such content is found
```

Input and output

Your program is not required to do any input at all, although you may choose to add some to aid in your testing.

The only part of the program that should perform any output should be inside the class PlayStoreMain. You may use print statements in part of your testing in other classes, but these should be removed/disabled when you submit your assignment.