COSC2531 Programming Fundamentals
Semester 1 2018
Assignment 2: PlayStore - Mobile Applications Management system

Introduction
This is a group assignment and worth 20% of your final grade. The grading is based on both demonstration and final code submission. Each group must consist of **3-4 students**. You need to demonstrate your assignment in your timetabled practical class prior to your final submission.

Objective
The main objective of this assignment is to familiarize students 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 10% of the final grade (half 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. 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 be clean and have proper comments. You can choose to demonstrate either during week 11 or week 12 tute-lab, but you can do it just once.
If you demonstrate in week 11 tute-lab, you will receive a bonus 1 mark. If you do not demonstrate by the end of week 12 you will receive marks only on the final submission code.
Final submission: The final submission is worth 10% of the final grade (half of the weight of Assignment 2), due on week 12 (27 May, 2018, 23:59 pm). **Only one group member needs to make the final submission.**
Your final submission through Canvas should be a zip file that includes the following files –

1. All the java files of your assignment. Please 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 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 document file with the brief description of changes as well.

Assessment criteria

Assessment Task	**Marks**
Class A (A.1 – A.4)	4 marks
Class B	1 mark
Class C - E	1.5 mark (3X0.5)
Functionality 1	1 marks
Functionality 2	3 marks
Functionality 3	1 marks
Functionality 4 - 6	3 marks (3X1)
Functionality 7	2 marks
Functionality 8	1 marks
Others 1. Correctness / Test cases & results 2. Code quality / comments 3. Modularity/Use of classes, proper access modifiers, inheritance, polymorphism, methods & arguments 4. Logic / Strategy clearly explained 5. Reflection / Lessons learnt	2.5 marks

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 playStore are either free or can be bought for a price.

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. Where the specification is subject to interpretation, you may make any reasonable assumptions but you are required to justify those using comments.

There is a sample main method in PlayStore.java file along with its expected outputs are provided in the Canvas course shell for testing. You may write your own main method for testing. If you want to demonstrate with your own written main method, make sure you include checking all the functionalities with proper comments. All attributes and methods (except 'main' method) are non-static.

Section 1: The classes and their attributes – write the required classes

Class A

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 review. **A content type of object cannot be created.** Review is an array of 'Comment' type of objects (See Class B for details).

Class A.1

A type of content is 'Game'. In addition to the data that a content class have, a game type of 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 OS class) for the game. If the value of the variable 'isMultiPlayer' is false, that means it's a one player game. A game type of object can be initialized as

Game g1 = new Game ("G101", "Pokemon", 10, false, os1);

Here "G101" is the content ID, "Pokemon" is the application name, 10 is the price of the game in dollars, it is a one player game, and os1 is the 'OS' type of object (details in Task C). Initially the number of downloads is zero, and the review is empty. If the price is not mentioned in the initialization, like the following –

Game g1 = new Game ("G101", "Pokemon", false, os1);

That means the game is free, i.e., the price of that game is zero.

Class A.2

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

Class A.3

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

String[] author = {"L. Tolstoy"};

Book b1 = new Book ("R21", "War and Peace", 12, "The Russian Messenger", "Novel", 1225, author);

Here the book title "War and Peace" is the name of the content (as the application name of the content class), 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. **Note that, a book may have multiple**

authors. Similar to the Game class, a Book object may not have its price in the initialization if it is free (i.e., the price of that Book object will be zero).

Class A.4
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' type of object can be initialized as -

Magazine m1 = new Magazine("M21", "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. **Similar to the Game class, a Magazine object may not have its price in the initialization if it is free (i.e., the price of that Magazine will be zero).**

Class B
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 an array of 'Comment' type of objects as the reply of the comment. A 'Comment' type of object can be initialized as -

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

Here, 'u1' is an 'User' type of object, and *"This is a fantastic game!"* is the string as comment. Initially, the reply is empty.

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

OS os1 = new OS("Android", 4);

Class D
Write a 'User' class, where each user has an ID, a name, a phone number, a balance, and an 'OS' type of object. Assume that the phone number is a string, as that will allow you to store a zero at the beginning. A 'User' can be initialized as -

User u1 = new User("u1", "John Doe", "0400000000", 20, os1);

or as -

User u1 = new User("u1", "John Doe", "0400000000", os1);

If the User object is initialized in the second way, the balance of that user is zero.

Class E
You have been provided a 'PlayStore' class in the Canvas that contains the main method. The 'PlayStore' class has a collection of 'Content' and a list of 'User' objects (you are encouraged to use Java collections like ArrayList and Hashtable). Note that each content can be uniquely identified based on content ID (Please see the use of java collections Hashtable for details on this requirement).

An instance of the PlayStore class, called 'Admin' is initialized from the main method. For this assignment, only the admin will carry out all operations on behalf of the users.

Section 2: Functionalities of the classes
Users' functionalities

1. Write a method of the User class "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. The balance of a user cannot become negative.

Exceptions must be thrown when an attempt is made to become premium without having sufficient balance. The exception thrown must indicate the cause of error, show an appropriate error message, and allowing the caller to proceed to the next statements of the program.

2. Write a method of the User class "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 increase 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, and allowing the caller to proceed to the next statements of the program.
   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, and allowing the caller to proceed to the next statements of the program.
      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.

3. A user may buy multiple contents. Write a method "AllContentsBought" in the User class to show the list of names of all the contents that the user bought. You may add additional attributes in the User class if necessary.

Contents' and Comments' functionalities

4. 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.
5. 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 cmnt = new Comment(u1, "This is a fantastic game!");
g1.addReviews(cmnt); //g1 is a Game type of object, see description of Class A.1
Comment r1 = new Comment(u2, "I never liked this game!");
cmnt.addReply(r1);
Comment r2 = new Comment(u1, "Really??");
r1.addReply(r2);
Comment cmnt2 = new Comment(u3, "The game crashes frequently.");
g1.addReviews(cmnt2);
```

6. Write a method "ShowReview" in the "Contents" class to show all the reviews of a particular content object. If a review has replies to its comments, show them as well.
   Show the replies of the comments with an indentation at the beginning of each reply. In case you are unable to show them with proper indentations, you will receive partial marks for showing the comments. Sample output is shown bellow, (also provided in the Canvas shell)

John Doe (u1): This is a fantastic game!
Jane Joe (u2): I never liked this game!
John Doe (u1): Really??
David Roe (u3): The game crashes frequently.

PlayStores' and Admin's functionalities

7. Write a method "showAllContents" 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?

   Also, you need to provide methods in the PlayStore class to add "Content" objects and "User" objects. You **don't** need to handle deletion of objects.

8. 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").

Other requirements:
Properly use the access modifiers of the attributes and the methods of the classes. Think if you require them to be private/public/protected/with no access modifier. If an attribute is private, do you need to write an accesor method to get the value of that attribute (e.g., getID() for 'Content' class)?

Use of Java Collections

You are encouraged to use collections such as ArrayList and HaspMap. ArrayList implements an array, which can grow indefinitely. HashMaps allow an association to be created between keys and objects. Using such classes also reduce the amount of code required as they provide methods for retrieving required objects easily. As examples -

```
ArrayList<Comment> exampleArrList = new ArrayList<Comment >();
Comment cmnt = new Comment(u1, "This is a fantastic game!");
exampleArrList.add(cmnt));
```

To extract the 4th element as example,

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

The above code segment is just an example on how to use ArrayList. You need to use a method in the Comment class to add reviews to a content type of object as described in Section 2, functionality 5.

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

```
HashMap<String, content> allContents = new HashMap<String, content>();
```

To add a content we use:

```
Game g1 = new Game ("G101", "Pokemon", 10, false, os1);
allContents.put("G101",g1));
```

To extract the content, use:

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

Sample main method (this sample main method in PlayStore.java and its output is provided in Canvas):
NOTE THAT, the main method provided in Canvas will NOT compile unless you write all the required classes, along with their attributes and methods.

```
public static void main(String[] args)
   {
      PlayStore admin = new PlayStore();
      //adding new readings
      String[] authors ={"L. Tolstoy"};
      Book b1 = new Book("R1", "War and Peace", 12, "The Russian Messenger", "Novel", 1225,
authors);
      String[] authors2 ={"F. Scott Fitzgerald"};
      Book b2 = new Book("R2", "The great gatsby", 10, "Charles Scribner's Sons", "Novel", 180,
authors2);
      String[] authors3 ={"Thomas H. Cormen", "Charles E. Leiserson", "Ronald L. Rivest", "Clifford
Stein"};
      Book b3 = new Book("R3", "Introduction to algorithms", 100, "MIT Press", "Computer
Science", 1312, authors3);
      Magazine m1 = new Magazine("R4", "Forbes", 8, "Forbes Media", "Business", 50, "World's
richest under 30");

      admin.addContents(b1);
      admin.addContents(b2);
      admin.addContents(b3);
      admin.addContents(m1);

      //adding new games
      OS os1 = new OS("Android", 4);
      OS os2 = new OS("iOS", 10);
      OS os3 = new OS("Android", 3);
      Game g1 = new Game("g1", "Pokemon", 5, false, os1);
      Game g2 = new Game("g2", "Pokemon", 5, false, os2);
      Game g3 = new Game("g3", "MineCraft", 2, true, os1);

      admin.addContents(g1);
      admin.addContents(g2);
      admin.addContents(g3);

      //adding new users
      User u1 = new User("u1", "John Doe", "0412000", 20, os1);
      User u2 = new User("u2", "Jane Doe", "0412001", 120, os1);
      User u3 = new User("u3", "Dave Roe", "0412002", 100, os2);
      User u4 = new User("u4", "Diane Roe", "0412003", 50, os3);
      admin.addUsers (u1);
      admin.addUsers(u1);
      admin.addUsers(u1);
      admin.addUsers(u1);


      //Users are buying contents
      u1.buyContent(b1);
      u1.buyContent(b3);
      u4.buyContent(g1);
      u1.buyContent(m1);
```

```
    //some users becoming premium and then buying contents
    u4.becomePremium();
    u4.buyContent(m1);
    u2.becomePremium();
    u2.buyContent(g2);
    u2.buyContent(g1);

    //showing contents bought by the user u2
    u2.showContentsBought();

    //showing all contents in the PlayStore
    admin.showAllContents();


    //showing all reading type of objects with the genre “Novel”
    admin.showReadingOfGenre("Novel");

        //Student to do: call a method to show all games. What should be the parameters of that
        //method?   See Section 2, functionality 7


    Comment cmnt = new Comment(u1, "This is a fantastic game!");
    g1.addReviews(cmnt);
    Comment r1 = new Comment(u2, "I never liked this game!");
    cmnt.addReply(r1);
    Comment r2 = new Comment(u1, "Really??");
    r1.addReply(r2);

    Comment cmnt2 = new Comment(u3, "The game crashes frequently.");
    g1.addReviews(cmnt2);

     //showing all reviews including the replies on object g1
    g1.printAllReview();
}
```