ENSE 370 Software Systems Design

Assignment A00

Java Refresher

(Due date: Friday February 5th 2021, 23:59)

Instructions:

1. Login to Snoopy (**snoopy.engg.uregina.ca** / **142.3.105.92**).
2. Create your program using any text editor of your choice (e.g. **vi / emacs**)
3. Name your files using the following convention:

Class definition files	ATM.java Person.java Operator.java Customer.java
Main program file	*"A"+number+username+".java"* (e.g. **A00jon123.java** if your username is jon123)

4. Compile your program and test that it works correctly
   (e.g. **javac ATM.java Person.java Operator.java Customer.java A00jon123.java**)
   (and **java A00jon123**)
5. Submit all your java source code files (total 5 files)
   (Type **~ense370/bin/submit A00 ATM.java Person.java Operator.java Customer.java A00jon123.java**)

In this assignment, you will create a Java program to represent an ATM system that interacts with an operator and a customer. You will need to build Java classes, including nested classes and inherited classes, create instances, and invoke class methods.

ATM.java

1. In a file "ATM.java", create a public class called **ATM** that has the following data members:
   a. An integer variable *cash* representing how much cash the ATM has, and
   b. A boolean variable *inService* representing whether the ATM is in service or not.

2. In the ATM class in the same file, create the following member methods:
   a. A default constructor taking no parameters, and sets *cash* to 0 and *inService* to **false**.
   b. An overloaded constructor taking two parameters, an integer parameter x and a boolean variable y, and sets *cash* to x and *inService* to y.
   c. **int** queryCash() : returns the amount of *cash* inside the ATM,
   d. **void** increaseCash(**int** x) : increase its current *cash* by an amount x,
   e. **void** reduceCash(**int** x) : reduces its current *cash* by an amount x,
   f. **boolean** getServiceStatus() **:** returns the value of the *inService* variable, and
   g. **void** changeServiceStatus() **:** toggles the *inService* variable from true to false, and vice versa. Print out the new service status.

   Don't forget to make all your member methods **public**.

3. In the ATM class in the same file, create the following member classes (i.e. nested classes):
   a. **class** CashDispenser :
      - no data members
      - only one member method:
        void dispenseCash(**int** x) : this method calls the ATM's *reduceCash*() method to reduce the ATM's *cash* by x and then prints "x dollars has been dispensed" to the screen.
   b. **class** ReceiptPrinter :
      - no data members
      - only one member method:
        void printReceipt() : this function prints "Receipt has been printed".
   c. **class** CardReader :
      - no data members
      - only one member method:
        void readCard() : this function prints "Card has been read".
   d. **class** KeypadDisplay :
      - no data members :
      - member functions :
        void displayPINverification () : this function prints "PIN has been verified".

   Don't forget to make all your member methods **public**.

4. In the ATM class in the same file, create the following instances of the nested classes (i.e. as data members):
   a. A "CashDispenser" object *dispenser,*
   b. A "ReceiptPrinter" object *printer,*

c. A "CardReader" object *reader,* and
d. A "KeypadDisplay" object *display*.

Person.java

5. In a separate file "Person.java", create a public class called **Person** that has the following data member and member methods:
   a. A String variable *name* representing the name of the person.
   b. A default constructor taking no parameters, and sets *name* to a default name.
   c. String getName() : returns *name*.
   d. **void** setName(String nameString) : sets the name to nameString and prints "The name is set to (nameString)" to the screen.

Operator.java

6. In a separate file "Operator.java", create the public class **Operator** inherited from the **Person** class (i.e. derived class) with the following data member and member methods:
   - no data members
   - only one member method:

     void topUpATM(ATM atm) : this method does the following to the object *atm*:
     i. Print the current *inService* status of the ATM to the screen.
     ii. Print the current *cash* amount in the ATM.
     iii. Check that the ATM is off. If the ATM is on (i.e. in service), turn it off.
     iv. If the ATM has less than $5,000 in cash, add $5,000 to the ATM.
     v. Print out the new amount of cash in the ATM.
     vi. Turn on the ATM.

Customer.java

7. In a separate file "Customer.java", create the public class **Customer** inherited from the **Person** class (i.e. derived class):
   - no data members
   - only one member method:

     void withdrawCash(ATM atm, **int** amount) : this method does the following to *atm*:
     i. Print the *inService* status of the ATM to the screen.
     ii. If the ATM is off, print "ATM is not in service" to the screen.
     iii. If the ATM has less than amount in cash, print "ATM has insufficient cash" to the screen.
     iv. Otherwise, do
        1. Call the *readCard*() method from the ATM's CardReader object.
        2. Call the *displayPINverification*() method from the ATM's KeypadDisplay object.

3. Call the *dispenseCash*() from the ATM's CashDispenser object
4. Call the *printReceipt()* from the ATM's ReceiptPrinter object, and
5. Print "(amount) successfully withdrawn from ATM" to the screen.

Main Program File

8. In your main program file, create a public class that has the same name as your main program filename. Create a *main*() method that does the following:
   a. Ask the user for an integer between 0 and 10,000. Read in this value.
   b. Create two instances of an ATM, one with zero cash and not in service, and another with the amount that the user entered and in service.
   c. Create one instance of an Operator and make him do the following:
      i. Ask the user for the name of an Operator
      ii. Set the Operator's name to the name that the user has entered,
      iii. Print "Processing ATM 1"
      iv. Call *topUpATM*() on the first instance of the ATM.
      v. Print "Processing ATM 2"
      vi. Call *topUpATM*() on the second instance of the ATM.
   d. Create one instance of a Customer and make him do the following
      i. Ask the user for the name of a Customer
      ii. Set the Customer's name to the name that the user has entered,
      iii. Ask the user for an amount to withdraw.
      iv. Call *withdrawCash*() from an ATM of your choice, for the amount specified by the user.

Your output should look as follows:

```
$ javac ATM.java Person.java Operator.java Customer.java
A00jon123.java
$ java A00jon123
Enter a number between 0 and 10000
8000
Enter a name for the Operator
John
The name is set to John
Processing ATM 1
current inService is false
current cash is 0
ATM now has 5000 dollars
inService is now true
Processing ATM 2
```

```
current inService is true
current cash is 8000
inService is now false
ATM now has 8000 dollars
inService is now true
Enter a name for the Customer
Mary
The name is set to Mary
Enter an amount to withdraw
2000
current inService is true
Card has been read
PIN has been verified
2000 dollars has been dispensed
Receipt has been printed
2000 successfully withdrawn from ATM
$
```

The End