Edwin Chan
CPSC 233 T01/T05

February 4, 2016 Handout

Banking

Part 1: Instance methods and variables

1. Class declaration
    - We will make two classes: *Banking* and *Account*
    - *Main()* entry point to program is in *Banking*.

2. Basic constructor for *Account* class.
    - We can now make an account.

3. We want to keep track of the *owner*, *balance*, and *accountName*.

4. We will change our constructor so that it stores a name.
    - Change the account we made in Banking, so that it includes a name (eg. *"Bob").*

5. We want to be able to deposit and withdraw money with the account.
    - Add two instance methods, *deposit*() and *withdraw*()

6. Let's deposit $100 now.

7. I need to check my account balance.
    - Add a get method to check my account balance.

Part 2: Class methods and variables

1. Let's create a Savings account.
    - We need a constructor which can change the accountName.

2. Let's add $50 to the Savings account. (instance method)

3. How can I check the total balance of all my accounts?
    - Add a class variable (static) that stores the total balance.
    - Every time I change the balance of an account, update the total balance as well.

4. I need to transfer $20 from my chqAccount to my savAccount.
    - Let's make a class method (static) which takes in two accounts, and transfer money between them.

Edwin Chan
CPSC 233 T01/T05

Part 3: More constructors

1. Add another constructor which also takes an initialBalance.

2. Modify the other two constructors so that they reuse this new constructor, by using the *this* keyword.

Part 4: Garbage collection

1. Similar to part 2, let's make a *numberOfAccounts* static (class) variable so I know how many accounts there are.
    - Every time I make a new account (constructor), I should add one to *numberOfAccounts*.

2. How can I remove an account?
    - Garbage collection, *System.gc()*, frees up (eg. removes) any object that has no references to it.
    - Removing a reference:
        i. chqAccount and savAccount are *references* to two instances of the Account class
        ii. These references are like house addresses. If we point them to *null* instead, the house doesn't have an address anymore, and nobody can find it.
        iii. *savAccount = null;*
    - Since savAccount doesn't have a reference anymore, I can run *System.gc()* to remove the account. System.gc() will try to run the *finalize() method* in the Account class.
    - In the account class:
        i. *protected void finalize() throws Throwable{*
            super.finalize();
            numberOfAccounts--;
            }
        ii. This code will run whenever the object is garbage collected.

3. Now I can add a static (class) method to check the number of accounts I have.