Types
I – Getting Started

Java Fundamentals

Lab 8: Types I – Getting started

Objective

The primary objective for this lab is as follows:

- To create and use references.
- To consolidate on passing by value and passing by reference

Part 1 – Creating and using reference types

Step by step

1. Back in the **labs** project which you created in Lab1, add a new package called **lab08**.
    a. Please refer to Lab1's instructions if you need help.
2. Create a new class called **Program** in this package with a main() method.
3. Create another class called Account with following private fields:
   `int` id;
   `string` owner;
   `double` balance,

   and methods:
   ```
   a. void Deposit(double amount) { }
   b. void Withdraw(double amount) { }
   ```
4. Make sure no one can deposit a negative amount of money!
5. Also make sure no one can withdraw money they don't have.
6. Create a constructor for the account class to set its state.
7. Create a method called **getDetails**() to return details of the account to console.
8. In the main()method create an instance of the account class.
9. Invoke its deposit() and withdraw() methods.
10. Call the account's getDetails() method and print the result to make sure your code has had the expected effect.

Part 2 – Passing reference types to a method

In this section you'll examine passing reference types between methods. This is an important topic which has security implications.

Step by step

1. In the Account class, create a new method as:
   public void **AddInterest**()

   Write code in this method to add 2.5% interest to the balance.
2. Add code in main() to create an account called ***myAccount***, with a balance of £100.
3. Call the addInterest method of the myAccount instance.
4. After the call, display the details of myAccount (using its getDetails()).
   Did the balance change?
5. Create another Account reference called partnerAccount and set it to the myAccount instance as:
   Account partnerAccount = myAccount;
6. Call the partnerAccount.addInterest() method.
7. Call myAccount.getDetails() method. Did the balance change?

Let's do another experiment

1. Create a method in the **Program** class (just under the main())
   static void **processAccount**(**Account acc**){
   acc.addInterest();
   }
2. Back in Main(), create an instance of Account called *myAccount* and call the processAccount method, passing myAccount as parameter.
3. Call myAccount.getDetails() method. Did the balance change?
4. Let's do another important experiment.
5. Create another method under the main() method as:
```
private static void incInt(int x) {
      x++;
}
```
6. Add code to the main() method to create an integer like
   int k=100
7. Call incInt and pass k to the method.
8. After the call, print the value of k.

Did k change after the call?

All primitive data types (built into the language) are **Passed by Copy** (also known as **passed by value**). Therefore, changing x did not change k and changing the name of k to x will not alter anything!

All reference types (like myAccount) are also passed by copy, but what is passed is their address in memory. Therefore, when you call a method like **processAccount**(**Account acc**) and pass it an account reference (such as myAccount) you are actually passing its address! The **acc** reference will be pointing to (referencing) myAccount. Any change made by **acc** will directly affect myAccount.

**** End ****