

Lab 10: Introduction to Inheritance

Objective

To learn about inheritance with concrete and abstract classes

Background Information

Refer to the lecture notes for more details.

Material

- **Dr Java**

Lab Instructions:

Complete the following exercises:

1. <http://www.seas.upenn.edu/~palsetia/java/Botworld/betterbot/assignment.html>
2. <http://www.seas.upenn.edu/~palsetia/java/Inheritance/Account/account.html>

Post-Lab Questions

1. Answer the following questions related to betterbot exercise:
 - a. Consider the following statements:

```
Bot b = new BetterBot(new BotWorld());  
System.out.println(b.toString());
```

What is the outcome when the print statement is executed if:

- i) `toString()` is present only in the `Bot` class
- ii) `toString()` is overridden in the `BetterBot` class

- b. Is the following statement below valid?

```
BetterBot b = new Bot(new BotWorld());
```

- c. Consider the following statement:

```
Bot b = new BetterBot(new BotWorld());  
b.turnRight();
```

Does the above code execute? If not, is it a compile-time or a runtime error?

2. Answer the following question related to the account exercise:
 - a. Describe your implementation. Distinguish between concrete and abstract classes if any. You can also illustrate your design with a diagram.
 - b. How difficult would it be to extend your scheme so each account also stored its minimum monthly balance? Propose any design changes if needed to accommodate this feature.
 - c. What ramifications does minimum monthly balance (from Q2) have for using inherited `withdraw(..)` (which may be overridden `withdraw(..)` in sub classes) versus just changing the balance instance variable directly?
3. When would you use inheritance over composition (has-a relationship)?
4. Why would use an abstract class, and give an example of an instance where using an abstract class would be appropriate (do not give examples from lecture notes or lab examples)?

Turn-in the necessary files needed for each lab exercise to Blackboard.