CMSC 143: Introduction to Object-Oriented Programming with Robots

Lab 11: Programming with Class
Due December 7, 2009

In this lab, we will improve upon the small game we created in lecture. In the first part of the lab, you are asked to create class diagrams of the existing classes in the game. A class diagram describes the attributes and methods of the class as well as type information. For example, consider the the following Duck class:

```
class Duck:

  DUCKS = 0

  def __init__(self, name):
    self.name = name
    self.wings = 2
    self.body = Scribbler()
    Duck.DUCKS = Duck.DUCKS + 1

  def quack(self, voice):
    setVoice(voice)
    speak(self.name + " says quack")

  def waddle(self):
    self.body.move(1, -0.5)
    self.body.move(1, 0.5)
    self.body.stop()

  def __str__(self):
    return self.name + " with " + self.wings + " wings"
```

Name	Duck
Class Attributes	DUCKS: int
Object Attributes	name: string wings: int body: Scribbler
Methods	quack(voice: string) waddle() __str__(): string

In the second part you should design **at least two** new classes that will be part of the game. For example, you could create a treasure object that rewards the player when collected, or create a new type of enemy.

You **should first** come up with class diagrams for your new classes before starting to write any python code. Finally, you should integrate your new classes into the game.

Your classes should be properly documented (each class and method should have a pydoc string). Also, there should be a `howDoIPlay()` function that prints out directions on how to play the game.

Learning Objectives

- Create Class Diagrams
- Design Classes
- Implement Classes
- Create a Game

Deliverables

1. `cmsc143_lab11_LASTNAME_FIRSTNAME.pdf` – Your class diagrams.
2. `cmsc143_lab11_LASTNAME_FIRSTNAME.py` – Your program.