HST 952

Computing for Biomedical Scientists

Lecture 2

Object Oriented Programming

- In this course, we will examine imperative, object-oriented programming (OOP) using Java
- Imperative OOP: objects issue commands
- Object: an entity (car, dog, house, person, etc.)
- In OOP, data and the methods for manipulating the data are grouped together in an object

Object Oriented Programming

- To create an OO program we need to identify
  - all the objects that we want to manipulate
  - the properties that these objects have
  - how these objects relate to/interact with each other
- This process is called *data modeling*

Example problem: scheduling mid-term exams
for four courses that have some of the same students
so that no two students have an overlapping exam

Object Oriented Programming

- Objects are defined using classes -- an object is an instance of a class
- Each class specifies attributes (properties) of an object and the object's behavior (via methods)
- An object can perform actions by invoking methods defined in its class

Object Oriented Programming

- To perform a task, find an object and send it a message (a message is a request to perform a method)
- If no appropriate object is available, create one using an already defined class
- If no class is available that does what you want, write a new class

A simple Java class

```
import java.io.*;
/* The SimpleProgram class implements an application
 *  that displays “Hello world!” to the standard output
 */
public class SimpleProgram {
   public static void main(String[] args)
   {
      // Display the words “Hello world!”
      System.out.println(“Hello world!”);
   }
}
```

Must be saved in a file called SimpleProgram.java

Object Oriented Programming

OOP follows three main design principles:

1. Encapsulation (information hiding)
   - Users of an object see only what is absolutely necessary for using that object
   - As an abstraction mechanism, encapsulation frees us from the details of a particular implementation of an object
   - We can focus on what an object can do (its *interface*) rather than how it does it

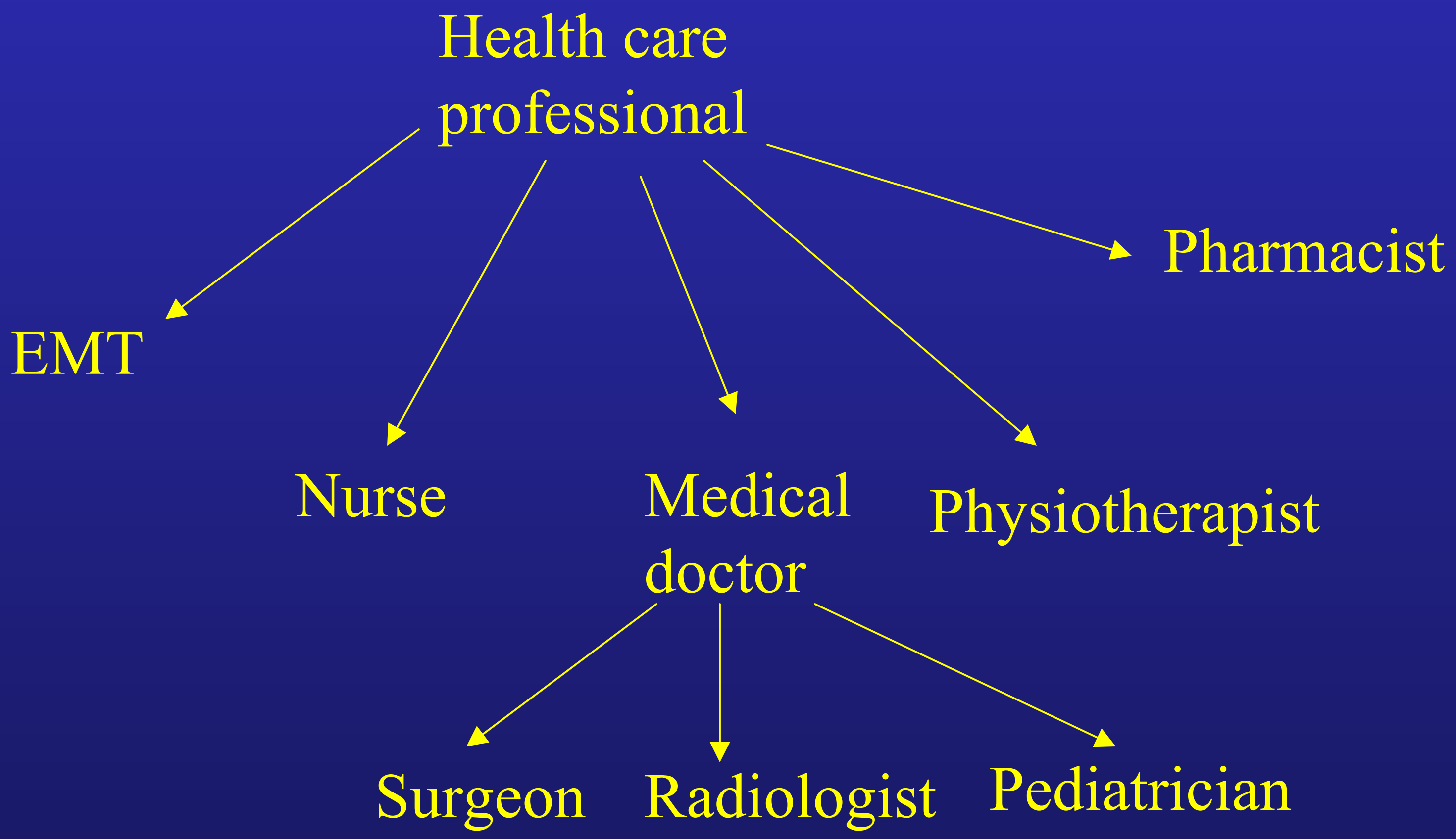
Object Oriented Programming

2. Polymorphism (e.g., method overloading)
   - an instruction (via a single method name) can be issued using different types of objects
   - different actions are performed depending on the objects used

Object Oriented Programming

3. Inheritance - means by which classes that have common or overlapping properties can have those properties specified just once
   - can base a new class on an existing class with more general properties: e.g., base “student” on “person”
   - common or base class is the *superclass*
   - new (derived) class is the *subclass*
   - subclass is usually a specialization of the superclass

Object Oriented Programming



Health care professional

EMT

Nurse

Medical doctor

Physiotherapist

Pharmacist

Surgeon

Radiologist

Pediatrician

Object Oriented Programming

Problem: Given a list of shapes, draw them

Non-OO procedural programming approach (e.g. C, Pascal):

```
for each shape, s in the list
    if s is a square
        specify square drawing code
    else if s is a rectangle
        specify rectangle drawing code
    else if s is a circle
        specify circle drawing code
```

Object Oriented Programming

Problem: Given a list of shapes, draw them

Imperative OOP approach (e.g. Java, C++):

```
for each shape, s
    s.draw()
```

- There is a base shape class with a draw method
- Each subclass of shape implements its own version of the draw method so it knows how to draw itself

Object Oriented Programming

If we add a new shape to the list using the procedural approach, need to add another conditional statement and code for drawing it

If we add a new shape in the OOP approach, we only need to implement its draw method

OO Modeling Problem

A bakery shop sells 5 different types of pastries and orders common ingredients used to make all pastry types each week. While all the pastries require flour, butter, sugar, and eggs, each of the five types of pastries uses one additional ingredient that the other four don’t. All pastries are wrapped and the wrapper bar-coded according to the pastry type. The store would like you to create an automated re-ordering program for the special ingredient: When the store has only 10 of a particular type of pastry left, an electronic order for its special ingredient should be generated to the particular store that sells this ingredient .

Next Class

Read Chapter 2 sections 2.1-2.4

Questionnaire 2