Assignment 2

SUMMARY
In this assignment you will write a program to play the game Connect Four.

DEADLINE
This assignment is due on Friday, April 4 at 11:00 pm.

DESCRIPTION
Connect Four is a popular game played by dropping checkers into a vertical grid with six rows and seven columns. The checkers can be dropped in any of the seven columns (permitted they are not full). Once dropped, the checker falls to the lowest unoccupied space. The winner is the first player to occupy four consecutive spaces in a straight line, either horizontal, vertical, or diagonal.

SPECIFICATIONS
In this project you will write a program to represent the game Connect Four using a two-dimensional array of integers to represent the grid and a Picture object to display the game. You will create a class called ConnectFour to represent the game. The class PlayConnectFour (provided) will allow the user to play the game. When run, this program will display the game board and ask the user to specify a column. The user is always player 1 (red checkers) and goes first. After the user enters a column the opponent will select a column for their piece. For this project you will create a simple naive opponent that selects a column at random. Then the program will update and repaint the board and ask the user to select another column. This will repeat until the game ends in a win, loss, or draw.

To help with this project a Referee class is provided. The referee will look at your grid and tell you whether the game is a win, loss, draw, or is not finished.
Dynamic Binding

• A polymorphic method
  – A method that has multiple meanings
  – Created when a subclass overrides a method of the superclass

• Late binding or dynamic binding
  – The appropriate version of a polymorphic method is decided at execution time

java.util.ArrayList

• An ArrayList object is an array that can grow or shrink as needed.
  – For instance, we usually don’t know the number of students that will be in a course.
    • If we create an array for more than we have, we waste space.
    • If we try to add a student past the end of the array,
      – We get an exception.
• Use an ArrayList when you don’t know how many of something you need.
ArrayList Methods

• Look in the Java API for ArrayList
  – Open the package `java.util`
  – Click on the class `ArrayList`.

• What method(s):
  – lets you add an object to the ArrayList?
    `add(Object o)`
    `add(int index, Object o)`
  – lets you get an object from the ArrayList?
    `get(int index)`
  – tells you how many things are in the ArrayList?
    `size()`
  – lets you remove an object from an index in the ArrayList?
    `remove(int index)`

An ArrayList is a List

• Look at the API for ArrayList

ArrayList Exercise

• In the Course class,
  – Modify `students` to be an `ArrayList`
    `List students = new ArrayList();`
  – Change all the methods that use an array to use an `ArrayList`.
    – Cast back to Student when you pull the object out of the list
      ```java
      public Student getStudent(int index)
      {
        return (Student) this.students.get(index);
      }
      ```
Collections Store Objects

• Why do we need to cast the Student object back to Student when we pull it back out of a list?
  – A list is a collection of (any type of) objects.
  • We need to put it back into a Student object.

Collections Store Objects

• Casting from Object to Student is also called **downcasting**.
• Going from Student to Object is called **upcasting** and doesn’t require any casting.
• In other words, we need a cast when we need the more specific type.