Delta College

CST 283
Programming Assignment 6

Winter 2024
Instructor: T. Klingler

Objective

This program will provide an opportunity to solve a problem using class inheritance.

Overview & Instructions

Write a program that will manage a sports recruiting operation. Your goal is to examine athletes from three sports and select those with special attributes for further consideration. This implies use of four classes: one for general player attributes and methods, and then one for each targeted sport. Utilize inheritance to define the three specialized classes from the one general class.

Class	Purpose	Data Members
`Player`	Store general information about one athlete.	Name, height, weight, GPA
`FootballPlayer`	Manage information related to a potential **football** recruit.	Attributes of Player plus • 40 yard dash time
`VolleyballPlayer`	Manage information related to a potential **volleyball** recruit.	Attributes of Player plus • Service aces per game • Kills per game
`HockeyPlayer`	Manage information related to a potential **hockey** recruit.	Attributes of Player plus • Goals per season • Penalty minutes per season • Plus-minus number

Include set/get methods as needed, but include at least one constructor per class. Your program is mainly designed to identify players for serious recruiting if they have all of the attributes from a list. The following table describes what players should be identified:

Class	Target Attributes
Any `Player`	Must have a GPA of at least 2.3 (this method should reside in the `Player` class)
`FootballPlayer`	Height over 73 inches and weight over 190 lb 40 yard dash time under 4.7 sec
`VolleyballPlayer`	Height at least 70 inches At least 6 service aces per game At least 9 kills per game
`HockeyPlayer`	At least 15 goals per season Less than 12 penalty minutes per season Plus-minus number greater than 8

The methods for specifically assessing a target candidate for the individual sports should reside within the respective class.

Your program should be driven by a list of athletes (found in file players.txt). Each line of data in this file represents a player.

The format of the file begins with:

- Game code (**F**, **V**, or **H**), name, height (inches), weight (pounds), GPA

Beyond these, additional data items will be included for each type of athlete:

- Football: 40 yard dash time
- Volleyball: services aces per game, kills per game
- Hockey: goals per season, penalty minutes per season, plus-minus number

For processing, define basic array or `ArrayList` of **`Player`** objects. Your program should read the file above, determine what type of athlete the data represents,read the subsequent related attributes of the athlete, instantiate the correct object type (**`football`**, **`volleyball`**, or **`hockey`**), and add it to the array.

Next, sort the array of **`Player`** objects by *descending GPA*. Your output should simply be a sequence of all names followed by their sport name, and whether or not you they are being targeted for recruitment.

For determination if they are to be recruited, invoke a ***abstract*** method **`isCandidate()`** using polymorphism to determine if a player is a candidate for further recruitment. This implies a declaration of **`isCandidate()`** in **`Player`** as **`abstract`** followed by individually declaring the method in each of the three sports classes. The specific sports recruiting requirements can be then implemented via the three respective methods. Remember that all athletes must reach a certain GPA, so be sure the common **`Player`** method checking for GPA is included as part of the specific **`isCandidate()`** actions.

Include a **simple JavaFX user interface** that includes a **text area** and a **button**. When the button is clicked, the file should be processed and the list of candidates described above could be produced as string data that can be passed in to be displayed in the text area.

Deliverables

Demonstrate the development steps of your program with at least two version commits to the assignment Git repository.

Deliver the following to the eLearning system **Assignment Dropbox** as your final product:

- **Upload**
    - your **source code** (.java) file(s); preferably zipped in more than one file
    - the **data file** used for testing