Data Structures and Algorithms

Comp 200

Fall 2022

Department of Computer Science

Forman Christian College University

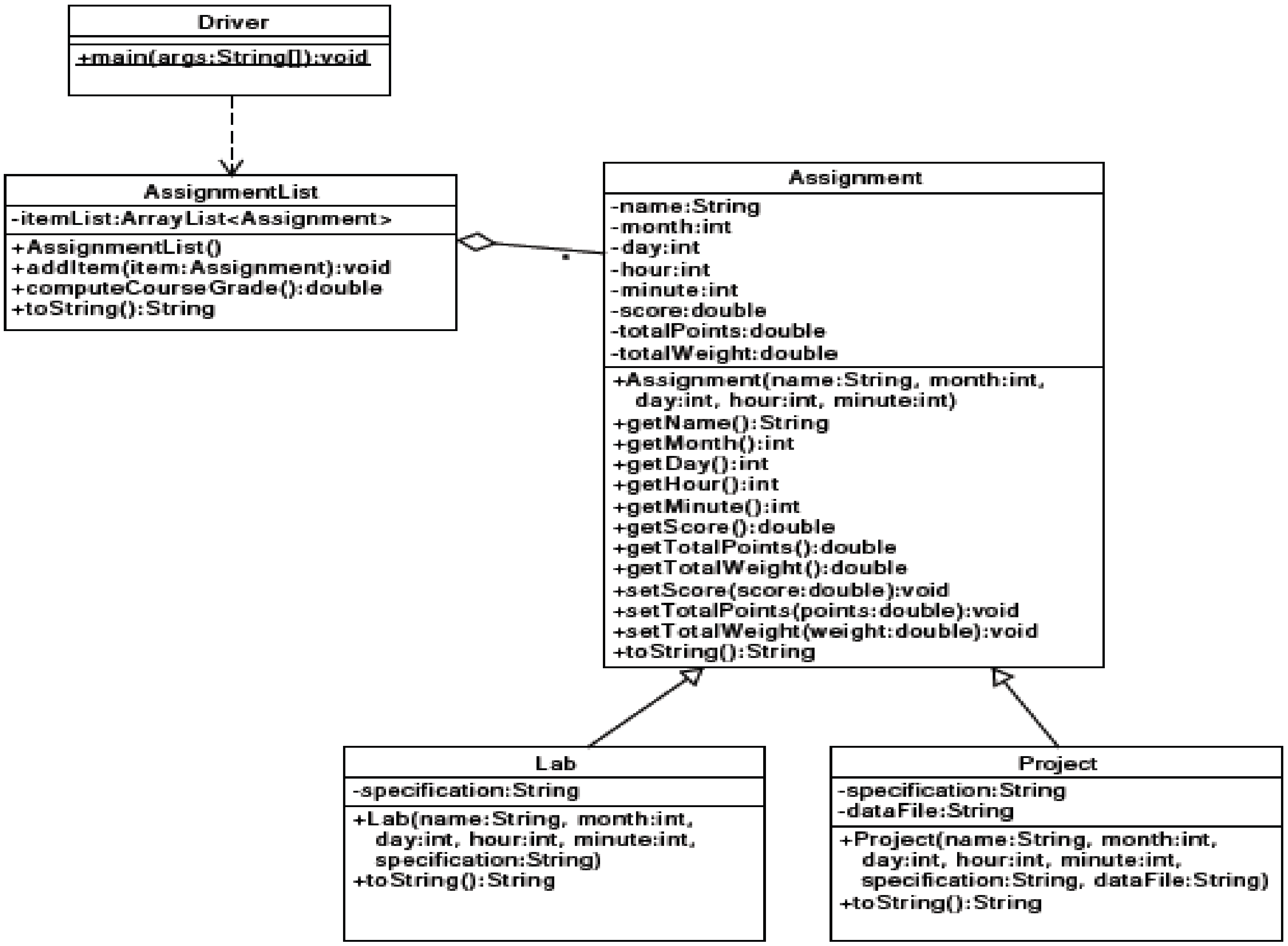
Lab 1
Review of Programming II – Aggregation, Inheritance, Polymorphism and UML

Question #	**Total Marks**
Question 1	10

Lab Problem

In this lab, we consider relationships between objects. Think about the records that we keep for every item used to compute your grade in this course. Each of these Assignments has a name, a date, a score, a total point count and a weight.
We will use class inheritance in this lab to capture the fact that different types of Assignments have some properties in common, but differ in other properties. For example, a Lab might have a specification document, and a Project might have both a specification and a data file. In addition, we will use class aggregation to represent a list of Assignments.
By the end of this laboratory exercise, you should be able to:
1. Read and understand UML diagrams involving class inheritance
2. Implement classes from a given specification
3. Create inheritance relationships
4. Understand the difference between is-a and has-a relationships
5. Develop testing procedures for your own code

Below is the UML representation of a set of classes that represent items that are used to compute your course grade. Your task is to implement this set of classes.


Driver
+main(args:String[]):void
AssignmentList
-itemList:ArrayList<Assignment>
+AssignmentList()
+addItem(item:Assignment):void
+computeCourseGrade():double
+toString():String
*
Assignment
-name:String
-month:int
-day:int
-hour:int
-minute:int
-score:double
-totalPoints:double
-totalWeight:double
+Assignment(name:String, month:int, day:int, hour:int, minute:int)
+getName():String
+getMonth():int
+getDay():int
+getHour():int
+getMinute():int
+getScore():double
+getTotalPoints():double
+getTotalWeight():double
+setScore(score:double):void
+setTotalPoints(points:double):void
+setTotalWeight(weight:double):void
+toString():String
Lab
-specification:String
+Lab(name:String, month:int, day:int, hour:int, minute:int, specification:String)
+toString():String
Project
-specification:String
-dataFile:String
+Project(name:String, month:int, day:int, hour:int, minute:int, specification:String, dataFile:String)
+toString():String

The line from Lab to Assignment indicates that a Lab is-a Assignment. You should use inheritance to capture this relationship. Likewise, for the Project and Assignment classes.
The line from AssignmentList to Assignment indicates that AssignmentList has-a Assignment. Observe that there is an itemList attribute (instance variable) in the middle section of the AssignmentList box. Other than including this attribute, there is nothing special to be done for a has-a relationship. The “*" on this relationship indicates that the AssignmentList can have any number of Assignment objects.
How much a particular Assignment contributes to your course grade is determined by three properties:

1. score is the number of points that you have received on the assignment,
2. totalPoints is the number of points that are possible for the assignment, and
3. totalWeight is the weight of this Assignment relative to other assignments.
4. AssignmentList.computeCourseGrade() returns your score for the class based on the list of assignments. In particular:

$$return = \frac{\sum_{i=0}^{n-1}(totalWeight_i \times score_i/totalPoints_i)}{\sum_{i=0}^{n-1} totalWeight_i}$$

where n is the total number of Assignments in the list, and $totalWeight_i$ refers to the totalWeight of the ith Assignment. If n = 0, then the returned course grade must be zero.

5. Test your class well by writing main function. There should be enough objects in the list to verify your program. You can also store the data in file and reuse it for every execution.