Help – Project Week 04

Our project is to modify an example program that uses Inheritance and an abstract superclass. The textbook example of the Employees program zyDE 10.13.2 uses the concept of a base class being abstract because we just need a framework of data and methods to build child classes from. The base class called EmployeePerson is the foundation for a Manager as well as a Staff member but the authors did not need to define generic objects of type EmployeePerson so that is why EmployeePerson was declared abstract.

There are some important concepts in the example program and additional concepts we deploy in our project:

1. The "protected" keyword used to declare instance variables allows us to reference the variable names directly in the child class definition. The topic of "protected" is covered in section 10.2 and access information is shown in table 10.2.1.
2. The keyword super followed by a dot and then a method name (super.somePrivateParentMethod()) is how you access a "private" method in a parent/superclass – see section 10.3 – Calling a base class method. You can call/invoke a base class' constructor with super() or super(one or more arguments separated by commas) which is not explicitly shown in the chapter. We use super even though the instance variables in CoursePerson are defined as "pretected" so our program will work whether the base class' instance variables are declared "protected" or the more strict declaration of "private".
3. When a class contains an abstract method, the class itself must be declared abstract - see the definition of abstract class at the top of section 10.8.
4. In our project for Week 04, we declare and define our CourseTeacher and CourseStudent object variables with the constructors that take all the attribute values. The Employee example program uses a setData() method as a secondary call in the main() method to define the object variables.
5. When you make an array of a Class type (such as CourseStudent), you first declare the array of the Class type followed by "= new " and then the Class type again but include the size of the array in the square brackets (such as "[5]"). The array declaration statement does not create the element objects… that must be done element by element.

   See the following Website for more info:

   https://www.javatpoint.com/how-to-create-array-of-objects-in-java

   Example from above URL:

   ```
   //create an array of product object
   Product[] obj = new Product[5] ;
   //create & initialize actual product objects using constructor
   obj[0] = new Product(23907,"Dell Laptop");
   obj[1] = new Product(91240,"HP 630");
   obj[2] = new Product(29823,"LG OLED TV");
   obj[3] = new Product(11908,"MI Note Pro Max 9");
   obj[4] = new Product(43590,"Kingston USB");
   ```

6. When you create the NetBeans project, you were told to make the project name and package name the same but to change the main class name. See the example below (Fall 2022 shown).
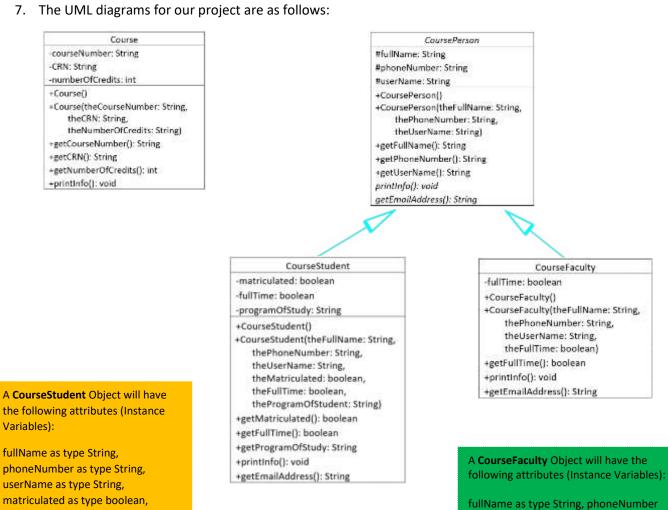

7. The UML diagrams for our project are as follows:

Course

-courseNumber: String
-CRN: String
-numberOfCredits: int
+Course()
+Course(theCourseNumber: String,
    theCRN: String,
    theNumberOfCredits: String)
+getCourseNumber(): String
+getCRN(): String
+getNumberOfCredits(): int
+printInfo(): void

CoursePerson

#fullName: String
#phoneNumber: String
#userName: String
+CoursePerson()
+CoursePerson(theFullName: String,
    thePhoneNumber: String,
    theUserName: String)
+getFullName(): String
+getPhoneNumber(): String
+getUserName(): String
printInfo(): void
getEmailAddress(): String

CourseStudent

-matriculated: boolean
-fullTime: boolean
-programOfStudy: String
+CourseStudent()
+CourseStudent(theFullName: String,
    thePhoneNumber: String,
    theUserName: String,
    theMatriculated: boolean,
    theFullTime: boolean,
    theProgramOfStudent: String)
+getMatriculated(): boolean
+getFullTime(): boolean
+getProgramOfStudy: String
+printInfo(): void
+getEmailAddress(): String

CourseFaculty

-fullTime: boolean
+CourseFaculty()
+CourseFaculty(theFullName: String,
    thePhoneNumber: String,
    theUserName: String,
    theFullTime: boolean)
+getFullTime(): boolean
+printInfo(): void
+getEmailAddress(): String

A **CourseStudent** Object will have the following attributes (Instance Variables):

fullName as type String, phoneNumber as type String, userName as type String, matriculated as type boolean, fullTime as type boolean, and programOfStudy as type String

A **CourseFaculty** Object will have the following attributes (Instance Variables):

fullName as type String, phoneNumber as type String, userName as type String, and fullTime as type boolean

Sample Output of Completed Project:

```
run:
Course Info:

Course #: CST141, CRN: 95537, Number of Credits: 4

Teacher Info:

Name: Prof. H., Phone #: 631-548-2691, Username: hassile, Full Time: true, Email: hassile@sunysuffolk.edu

Students in course:

Student #0
Name: Sally, Phone #: 631-555-1111, Username: sally01, Matriculated: true, Full Time: true, Program of
Study: CST, Email: sally01@mail.sunysuffolk.edu

Student #1
Name: Mohammed, Phone #: 631-555-2222, Username: moham56, Matriculated: true, Full Time: true, Program of
Study: CST, Email: moham56@mail.sunysuffolk.edu

Student #2
Name: Alex, Phone #: 631-555-3333, Username: alex93, Matriculated: false, Full Time: false, Program of
Study: N/A, Email: alex93@mail.sunysuffolk.edu

Student #3
Name: Barbara, Phone #: 631-555-4444, Username: barba28, Matriculated: true, Full Time: false, Program of
Study: GenStud, Email: barba28@mail.sunysuffolk.edu

Student #4
Name: Monique, Phone #: 631-555-5555, Username: moniq71, Matriculated: true, Full Time: true, Program of
Study: CST, Email: moniq71@mail.sunysuffolk.edu
BUILD SUCCESSFUL (total time: 0 seconds)
```