CPSC 411-03
Exercise #3

[Swift DSL]

In this exercise, you will create Swift DSL for adding Faculty Members to a given Department. You will perform the following two tasks to accomplish it.

1. Implement a class named *Faculty* which is derived from the *CampusMember* class. The class is used to model faculty members on campus.
    a. There are two types of faculty members: Tenured or PartTime faculty members. To this end, you will add a property named *facultyType* to the Faculty class. Since facultyType property could have only two possible values, it should be declared as an *enum* type.
    b. Add the following method to establish the bi-directional (*workFor*) relationship between Faculty and Department.

        ```
        func hiredBy(m: Department) -> Void
        ```

        The implement of the method should mimic that of *Student.major(m:).*
2. You will implement the following DSL by applying the techniques we have used for creating Student-DSL in the class. The DSL should look like:

    ```
    department("Computer Science") {
        faculty(fn: "Shawn", ln: "Wang", id: 3647784).type(.Tenured)
        faculty(fn: "James", ln: "Shen", id: 7748488).type(.PartTime)
    }
    ```

The second part of this exercise is to refine the implementation for Student-DSL and Faculty-DSL to create the following hybrid DSL syntax where both Students and Faculties can be added to one Department:

```
department("Computer Science") {
    faculty(fn: "Shawn", ln: "Wang", id: 3647784).type(.Tenured)
    faculty(fn: "James", ln: "Shen", id: 7748488).type(.PartTime)
    student(fn: "Chris", ln: "Sampson", id: 3645484).yearGrad(2024)
    student(fn: "Jenny", ln: "Dang", id: 6748488).yearGrad(2025)
}
```

Hint: How about creating *campusmember* class and deriving both student and faculty classes from it? Then use the campusmember to implement @resultBuilder annotated class.