BÀI 4. PROTOCOL VÀ DELEGATE

✓ **Mục tiêu:**

- *Hiểu được tầm quan trọng của tính kế thừa*
- *Nắm được khái niệm, ý nghĩa, tầm quan trọng của protocol và delegate*

Bài tập 4.1. Áp dụng tính kế thừa

Đề bài: Ở bài tập 3.1, ta thấy **SINH VIÊN** và **GIẢNG VIÊN** đều là **NGƯỜI**, đều có các thuộc tính **Mã**, **Tên** và **Tuổi**, khi khởi tạo đều lấy ngẫu nhiên tên. Áp dụng kiến thức đã học về tính kế thừa, ta tiến hành cải tiến bài tập 3.1:
Yêu cầu thực hiên:

- Tạo lớp đối tượng **Nguoi** kế thừa từ lớp **NSObject**, gồm các thuộc tính: ma, ten, tuoi.
- Khai báo phương thức init thực hiện lấy tên ngẫu nhiên.
- Chỉnh lại lớp **SinhVien** và lớp **GiangVien** kế thừa từ lớp **Nguoi**.
- **Bỏ** thuộc tính **ma, ten, tuoi** tại lớp **SinhVien** và **GiangVien**.
- Tại phương thức **init** ở lớp SinhVien và GiangVien:
  - **Bỏ** đoạn lấy tên ngẫu nhiên.
  - Gọi phương thức khởi tạo của lớp cha.
- Chạy thử ứng dụng và so sánh kết quả trả về.

Mục tiêu:

- Hiểu và áp dụng tính kế thừa của đối tượng vào thực tế.
- Thấy được tầm quan trọng của tính kế thừa.

Gợi ý thực hiện:

- Để gọi phương thức khởi tạo của lớp cha ta dùng lệnh: `self = [super init];`

Bài tập 4.2. Tạo Protocol và Delegate

Đề bài: Kế thừa bài tập 4.1, được biết trong lớp có qui định về thời gian vào lớp, giảng viên và sinh viên đều phải thực hiện quy định này cụ thể như sau:

- Sinh viên: phải vào lớp trước lúc 6h45 để trực nhật.
- Giảng viên: có thể vào lớp lúc 7h.

Ngoài ra, giảng viên còn có thêm quy định về tính lương, tuy nhiên quy định này khá phức tạp và thường xuyên thay đổi nên phần này cần quản lý linh động. Cụ thể như sau:

Yêu cầu thực hiện:

- Tạo protocol đặt tên là "**QuyDinhChungProtocol**"
- Khai báo phương thức **vaoLop** cho **QuyDinhChungProtocol**. Bất kỳ lớp nào áp dụng quy định này đều phải triển khai phương thức này.
- Các lớp SinhVien và GiangVien đều phải áp dụng QuyDinhChungProtocol.
- Phương thức **vaoLop** khi được triển khai tại lớp **SinhVien** sẽ in ra màn hình log: "**Sinh viên phải vào lớp trước lúc 6h45 để trực nhật**".
- Phương thức **vaoLop** khi được triển khai tại lớp **GiangVien** sẽ in ra màn hình log: "**Giảng viên có thể vào lớp lúc 7h**".
- Tạo protocol đặt tên là "**QuyDinhLuongProtocol**"
- Tạo phương thức **tinhLuongChoGiangVien** cho protocol "**QuyDinhLuongProtocol**", có thông số truyền vào là một đối tượng giảng viên.
- Tại lớp **GiangVien**:
    - Tạo **delegate** cho lớp **GiangVien**, đối tượng quản lý delegate này phải áp dụng **QuyDinhLuongProtocol**.
    - Khai báo và triển khai phương thức **tinhLuongGiangVien** như sau:
        - Nếu đối tượng delegate của GiangVien không rỗng và đối tượng delegate này có triển khai phương thức **tinhLuongChoGiangVien** của protocol QuyDinhLuongProtocol, thì gọi thực thi phương thức **tinhLuongChoGiangVien** của đối tượng delegate.
        - Ngược lại, in ra màn hình: "Lương căn bản là:5,000,000".
- Tại phương thức **viewDidLoad** của tập tin **ViewController.m**, thực hiện:
    - Từ các đối tượng giảng viên gv và sinh viên sv đã được khởi tạo ở bài tập 3.1, gọi phương thức **vaoLop** thực hiện.
    - Gọi phương thức **tinhLuongGiangVien** của đối tượng giảng viên gv thực hiện.
- Chạy thử chương trình xem kết quả ở màn hình log.
- Vào lại phương thức **viewDidLoad**:
    - Gán **delegate** cho đối tượng giảng viên bằng **self**.
    - Gọi lại phương thức **tinhLuongGiangVien** cho đối tượng giảng viên.
- Áp dụng protocol **QuyDinhLuongProtocol** cho lớp **ViewController**.
- Triển khai phương thức **tinhLuongChoGiangVien** của protocol **QuyDinhLuongProtocol**:
    - Nếu giảng viên trên 30 tuổi thì lương tăng thêm 2,000,000
    - Nếu giảng viên trên 40 tuổi thì lương tăng thêm 4,000,000

  - Nếu giảng viên trên 50 tuổi thì lương tăng thêm 6,000,000
  - Còn lại thì vẫn giữ mức lương căn bản là 5,000,000
  - In ra tuổi và lương của giảng viên như sau: "Tuổi của giảng viên là …, lương của giảng viên là …".
- Chạy lại chương trình và xem kết quả ở màn hình log.

Mục tiêu:

- Biết cách tạo và hiểu được cách thức hoạt động của protocol.
- Thấy được ý nghĩa của việc kết hợp giữa protocol và delegate.
- Hiểu được nguyên tắc hoạt động và biết cách sử dụng protocol kết hợp delegate để làm tiền đề cho các bài sau.

Gợi ý thực hiện:

- Để tạo phương thức cho protocol mà bất kỳ lớp nào áp dụng quy định này đều phải triển khai phương thức này thì ta sử dụng **@require**.
- Để format kiểu số theo định dạng lương 5,000,000 như ở trên ta làm như sau:

```
//Khởi tạo đối tượng NSNumberFormatter để định dạng lương hiển thị
kiểu 5,000,000
NSNumberFormatter *numberFormat = [[NSNumberFormatter alloc] init];
//xét kiểu hiển thị là decimal(số thập phân)
[numberFormat setNumberStyle: NSNumberFormatterDecimalStyle];
//Vì đối tượng numberFormat cần một đối tượng kiểu NSNumber để chuyển
đổi về kiểu NSString
//nên ta khởi tạo một đối tượng kiểu NSNumber để lưu trữ giá trị của
lương. LUONG là một hằng số lưu trữ giá trị của lương
NSNumber *luongNumber = [NSNumber numberWithFloat: LUONG];
//Chuyển đổi lương từ số sang chuỗi theo định dạng số thập phân đã xét
ở trên
NSString *luongString = [numberFormat stringFromNumber: luongNumber];
```

- Để kiểm tra xem đối tượng delegate có triển khai phương thức nào đó hay không ta có thể sử dụng phương thức respondsToSelector. Phương thức này trả về giá trị kiểu BOOL:

```
[self.delegate respondsToSelector:@selector(tinhLuongChoGiangVien:)]
```