1. Chia sẻ sự hiểu biết của bạn về Lập trình hướng đối tượng(OOP). Từ sự hiểu biết của bạn về OOP, hãy viết một chương trình tính tổng số tiền của một khách hàng. Khách hàng này có các tài khoản: thanh toán, tiết kiệm.

* Lập trình hướng đối tượng (OOP) là một kỹ thuật lập trình mà cho phép lập trình viên tạo ra các đối tượng trong code trừu tượng hoá các đối tượng thực tế trong cuộc sống.
* Hai khái niệm cơ bản chính của OOP là lớp (class) và đối tượng (object)
  + Class là một tập hợp các đối tượng (object) có cùng thuộc tính và phương thức
  + Định nghĩa Class thì ta định nghĩa ra các thuộc tính (properties) và các phương thức (methods)
  + Còn đối tượng (object) là một thể hiện của class
* Lập trình hướng đối tượng có 4 tính chất:
  + Tính đóng gói: các dữ liệu liên quan sẽ được đóng gói thành một lớp. Nên giúp che giấu dữ liệu với các đối tượng bên ngoài. Vì vậy sẽ chia thành các thuộc tính và phương thức qua cách truy cập: private, public
  + Tính kế thừa: ta có thể xây dựng lớp mới đơn giản hơn khi dựa vào lớp có sẵn. Lớp con sẽ mang đầy đủ các phương thức và thuộc tính của lớp cha
  + Tính đa hình: một hành động hay phương thức có thể thực hiện bởi nhiều các khác nhau.
    - Đa hình là hai hay nhiều lớp có phương thức giống nhau nhưng thực thi theo những cách khác nhau.
    - Ví dụ như lớp “chim” và lớp “mèo” đều có phương thức “kêu” nhưng mà tiếng kêu của chúng phát ra lại khác nhau
  + Tính trừu tượng: giúp ta khái quát hoá một lớp. Vì chỉ có khai báo các thuộc tính và phương thức chứ không định nghĩa cụ thể từng cái ra. Lớp trừu tượng không tạo được ra các thể hiện của nó. Các lớp con của lớp trừu tượng phải định nghĩa lại hết tất cả các phương thức và thuộc tính của lớp trừu tượng
    - Ví dụ: ta có lớp “động vật” có một phương thức là “phát ra tiếng kêu”
      * Ta có lớp “chim” kế thừa từ lớp “động vật” và ta phải định nghĩa lại phương thức “phát ra tiếng kêu” như “kêu chíp chíp”
      * Còn lớp “mèo” cũng vậy, định nghĩa lại phương thức “phát ra tiếng kêu” là “kêu meo meo”.

1. Chia sẻ sự hiểu biết của bạn về các “Type” trong Swift.

* Các kiểu dữ liệu cơ bản:
  + Integer
  + Float
  + Double
  + Bool
  + String
  + Optional: đại diện cho một biến có thể giữ giá trị hoặc không có giá trị
* Type aliases: ta có thể sử dụng để đặt tên cho 1 kiểu dữ liệu bất kì với điều kiện là kiểu dữ liệu đó phải tồn tại trong trình biên dịch
* Collection types
  + Array collection: gọi là mảng. Lưu trữ nhiều dữ liệu có cùng kiểu và sắp xếp thứ tự
  + Set collection: lưu trữ nhiều dữ liệu có cùng kiểu nhưng k có sắp xếp theo thứ tự
  + Dictionary collection: dùng để lưu trữ dữ liệu được đánh dấu bằng cặp key🡪value. Trong đó các key k đc trùng lặp, còn value có thể trùng lặp
* Value types: Mỗi instance giữ một bản copy duy nhất của dữ liệu, việc thay đổi bản copy sẽ k làm thay đổi bản còn lại. nếu mà là reference thí sẽ thay đổi
  + Ví dụ: Text, letter

    Description automatically generated
* Reference types: các instance chia sẻ từ một bản copy của dữ liệu và type sử dụng được định nghĩa **class**, giữ tham chiếu đến đối tượng. Dễ hiểu là việc thay đổi một instance sẽ ảnh hưởng tới những cái khác khi chúng reference chung với đối tượng
  + Ví dụ: Text

    Description automatically generated

1. Protocol

* Protocol là 1 kiểu interface mang tính chất trừu tượng
* Với Protocol ta có thể khai báo được thuộc tính
* Khai báo phương thức trong Protocol thì khai báo như hàm thông thương nhưng không cài đặt
* Protocol implement đc vào class, struct, enum
* Protocol có hàm khởi tạo init() nhưng là để các class/struct/enum nào implement nó thì bắt buộc phải khởi tạo. Bằng từ khoá “required”
* Có thể Implement được vào nhiều hơn 2
* Có thể xem như một kiểu dữ liệu
* Khi khai báo 1 protocol kế thừa từ một class thì protocol đó chỉ được dùng cho class thôi, struct và enum
* Cú pháp: Graphical user interface, text, application

  Description automatically generated

1. Thử implement một cấu trúc dữ liệu “Stack” hoặc “Queue
2. Chia sẻ sự hiểu biết của bạn về “AutoLayout”

* Autolayout là một tập hợp các quy tắc ràng buộc vị trí của view, nhằm mục đích tự động tính toán kích thước và vị trí của tất cả các views nằm trong hệ thống view được phân cấp của ta
* Để một giao diện hiển thị đúng với thiết kế trên các thiết bị có kích thước khác nhau thì autolayout sẽ đảm nhiệm việc đó
* Trong auto layout, các attributes được xác định một vị trí của view có thể bị ràng buộc. Thông thường nó bao gồm 4 cạnh (leading - Điểm đầu, trailling - Điểm cuối, top - Điểm trên, bottom - điểm dưới) cũng như width, height và thuộc tính vertical and horizontal centers. Text item thì còn có thêm một hoặc nhiều thuộc tính baseline - đường cơ sở

1. Chia sẻ sự hiểu biết của bạn về “UIScrollView”. Hãy thử giải thích bằng cách nào mà một scroll view có thể cuộn

* UIScrollView được sử dụng để hiện thị các nội dung có kích thước lớn hơn màn hình.
  + Frame trong UIScrollView là thuộc tính bao gồm toạ độ và kích thước của nó và xác định vị trí cố định của nó trên view chứa nó
  + Content Size là thuộc tính đại diện cho kích thước nội dung mà UIScrollView chứa nó
  + Content Offset là thuốc tính nhằm xác định được phần nội dung nào sẽ hiển thị được của toàn bộ nội dung mà UIScrollView chứa
* Nó cuộn được vì nội dung của UIScrollview to hơn kích thước của frame, nếu mà bé hơn thì k cuộn được