# Module 1

# Bootcamp Preparation

# Yêu cầu 1 ( If và switch case)

# Bài 1: Viết một chương trình để nhập lương nhân viên, tính thuế thu nhập và lương ròng (số tiền lương thực sự mà nhân viên đó nhận được). Với các thông số giả sử như sau (không theo luật lương, chỉ là con số giả sử để dễ tính toán):

# 20% thuế thu nhập nếu lương là 15 triệu.

# 10% thuế thu nhập nếu lương từ 7 đến 15 triệu.

# 5% thuế thu nhập nếu lương dưới 7 triệu

# Bài 2: Viết chương trình kiểm tra một ký tự trong bảng chữ cái tiếng anh là nguyên âm hay phụ âm. Ký tự là bất kỳ được nhập từ bàn phím. Các ký tự nguyên âm bao gồm: 'o', 'u', 'i', 'a', 'e' hoặc 'O', 'U', 'I', 'A', 'E'

# Yêu cầu 2 ( Loop )

# Bài 1: Vẽ tam giác vuông cân rỗng có chiều cao h. Ví dụ: h = 5

# \*

# \* \*

# \* \*

# \* \*

# \* \* \* \* \*

# Bài 2: Vẽ tam giác cân đặc có chiều cao h do người dùng nhập vào. Ví dụ: h = 5

# 

# \*

# \* \* \*

# \* \* \* \* \*

# \* \* \* \* \* \* \*

# \* \* \* \* \* \* \* \* \*

# Bài 3: Vẽ tam giác cân rỗng có chiều cao h do người dùng nhập vào. Ví dụ: h = 5

# \*

# \* \*

# \* \*

# \* \*

# \* \* \* \* \* \* \* \* \*

# Yêu cầu 3 (Mảng và Loop)

# Bài 1: Cho mảng a chỉ chứa các số nguyên

# Xây dựng mảng b là prefix sums của mảng a cụ thể theo công thức:

# b[0] = a[0]

# b[1] = a[0] + a[1]

# b[2] = a[0] + a[1] + a[2]

# ...

# b[n - 1] = a[0] + a[1] + ... + a[n - 1]

# với n là độ dài của mảng a.

# *Ví dụ:* Cho a = [1, 2, 3], thì kết quả prefixSums(a) = [1, 3, 6].

# Mảng b được tạo ra: [1, 1 + 2, 1 + 2 + 3] = [1, 3, 6]

# Bài 2: Cho 2 mảng A và B chứa số nguyên. Tìm ra một mảng đã được sắp xếp chứa các phần tử xuất hiện trong mảng B nhưng lại không xuất hiện A

# Chú ý các phần tử giống nhau chỉ tính là 1 lần xuất hiện

# Ví dụ: A=[7,2,5,3,5,3] và B=[7,2,5,4,6,3,5,3] thì kết quả missingValue(A, B) = [4, 6]

# Bài 3: Cho một danh sách các phân số, hãy tìm chỉ số của phân số lớn nhất (đếm từ 0)

# Giả định rằng không có các phân số bằng nhau trong tập đầu vào

# Ví dụ: numerators = [5, 2, 5] và denominators = [6, 3, 4], thì kết quả maxFraction(numerators, denominators) = 2.

# 5/4 là phân số lớn nhất, có chỉ số là 2.

# Bài 4: Giả sử menu của chúng ta có các loại thức uống như sau:

# Cafe

# Cam vắt

# Nước ép cà rốt

# Nước ép cà chua

# Nước lọc

# Nước dừa

# Viết một chương trình gọi thức uống đơn giản.

# Gợi ý: Trước tiên in ra màn hinh danh sách các loại thức uống cho người dùng chọn lựa

# Đánh dấu theo thứ tự 1 (cafe), 2 (cam vắt), 3(nước ép cà rốt), 4(nước ép cà chua), 5(nước lọc), 6(nước dừa), 7(thoát gọi món)

# Khi người dùng lựa chọn 1 trong số các thức uống, chương trình sẽ hiện đơn giá của thức uống đó và cho phép người dùng nhập số lượng. Cuối cùng hiển thị tổng số tiền người dùng cần phải trả.

# Sử dụng vòng lặp do while lặp lại cho người dùng đặt thức uống, kết thúc quá trình đặt thức uống khi người nhấn số 7

# Yêu cầu 4 (String)

# Bài 1: Viết một hàm chuẩn hóa xâu ký tự: biến đổi xâu ký tự thành xâu sao cho trong xâu không có 2 dấu cách liền nhau và bắt đầu mỗi từ phải in hoa. Ví dụ tRuong TaN HAI => Truong Tan Hai.

# Và một hàm tìm từ dài nhất trong 1 xâu ký tự. Hàm main sử dụng các hàm này để nhập xâu ký tự từ bàn phím, in ra xâu trước và sau khi chuẩn hóa, 1 từ dài nhất trong xâu đó.

# Bài 2: Cho xâu kí tự. Hãy đếm số lượng kí tự khác nhau trong xâu đó

# Ví dụ: s = "cabca", thì kết quả differentSymbolsNaive(s) = 3.

# Có 3 kí tự khác nhau là a, b và c.

# Bài 3: Cho hai xâu kí tự, tìm số lượng kí tự chung giữa chúng.

# Ví dụ: s1 = "aabcc" và s2 = "adcaa", thì kết quả commonCharacterCount(s1, s2) = 3.

# 2 xâu s1 và s2 có 3 kí tự chung: 2 kí tự 'a' và 1 kí tự 'c'.

# Yêu cầu 5 (Mảng 2 chiều)

# Bài 1: Viết hàm tính tổng các số chẵn trong ma trận có kích thước m\*n. Với m, n là các số nhập từ bàn phím.

# Bài 2: Viết hàm liệt kê các số nguyên tố trong mảng 2 chiều, đếm các số nguyên tố có trong mảng đó.

# Bài 3: Viết hàm tìm giá trị lớn nhất, nhỏ nhất trong mảng 2 chiều

# Yêu cầu 6 (OOP)

# Xây dựng hệ thống quản lý học viên CODEGYM

# Đối với mỗi học viên sẽ có các thông tin:

# Mã học viên

# Tên

# Lớp

# Email

# Ngày sinh

# Module

# Xây dựng hệ thống quản lý học viên và kết quả sau kỳ thi với các yêu cầu như sau:

# 1. Hiển thị tất cả học viên có trong danh sách.

# 2. Thêm mới học viên

# 3. Sửa thông tin học viên

# 4. Xóa học viên.

# Chi tiết các yêu cầu:

# 1. Đối với chức năng hiển thị học viên có trong danh sách sẽ được hiển thị dưới dạng bảng trên màn hình với giao diện tương tự dưới đây

# 

# 2. Khi click vào button thêm mới học viên chương trình sẽ hiển thị hộp thoại prompt() cho người dùng nhập thông tin của học viên. Sau khi người dùng nhập đầy đủ thông tin thì học viên sẽ được thêm vào danh sách.

# 3. Khi người dùng click vào button sửa thông tin học viên, chương trình sẽ hiển thị hộp thoại prompt() cho người dùng nhập vào mã học viên cần sửa chữa.

# Nếu mã học viên tồn tại thì sẽ cho người dùng sửa thông tin của học viên đó. Sau khi sửa xong thông tin được thay đổi lại trên màn hình danh sách.

# Còn nếu mã không tồn tại thì sẽ hiển thị hộp thoại thông báo cho người dùng “ Mã học viên không tồn tại”.

# 4. Khi người dùng click vào button xóa thông tin, chương trình sẽ hiển thị hộp thoại prompt() cho người dùng nhập vào mã học viên cần sửa chữa

# Nếu mã học viên tồn tại thì sẽ cho người dùng chọn xóa hoặc không. Nếu người dùng chọn xóa thì học viên sẽ mất đi trên màn hình danh sách còn nếu chọn không thì giữ sẽ giữ lại danh sách cũ.

# Ngoài ra nếu mã không tồn tại thì sẽ hiển thị hộp thoại thông báo cho người dùng “ Mã học viên không tồn tại”.

# Yêu cầu nâng cao:

# Đối với chức năng thêm mới:

# Mã học viên là duy nhất không trùng lặp và có định dạng HV-XXXX ( với XXXX là 4 chữ số )

# Tên học viên chỉ được tối đa 50 ký tự.

# Module chỉ được nhập từ 1 -> 6

# Ngày sinh đúng định dạng dd/mm/yyyy ( ngày / tháng / năm )

# Yêu cầu 7.

# Xây dựng giao diện như hình bên dưới:

# 