ITESM Campus Guadalajara
Fundamentos de Programación
Examen Parcial 2

"De Acuerdo con el Código de Honor del estudiante del Tecnológico de Monterrey, mi actuación en este examen estará guiada por la integridad académica."

Nombre: _____ Matrícula: _____ Grupo: 2

1. Crea un nuevo archivo y nómbralo **EP2_Matricula.py**. Por favor, elige las funciones a desarrollar e impleméntalas. De las dos opciones que tienen algunos incisos sólo puedes escoger una. No se necesita menú, sólo las funciones. Elige bien tus funciones y cerciórate de no exceder de 110 puntos.
    a. **(25 puntos)** Escribe la función **coincidencias**, la cual recibe dos cadenas, la función devuelve cuántas letras coinciden de manera continua en las dos cadenas, es decir, se debe dejar de contar hasta que encuentre la primera letra que sea diferente.
        i. Si la función recibe las palabras pez y pecera, devuelve 2
        ii. Si la función recibe las palabras computadora y computación, devuelve 7.
        iii. Si la función recibe las palabras pequeño y largo, devuelve 0.

        Restricción: Esta función debe utilizar un ciclo While. Asume que las cadenas no son vacías, pero si pueden tener diferente número de caracteres.

    b. Elige una de las dos opciones:

    En un estacionamiento de un solo piso, cada lugar está identificado por una letra y un número. La letra corresponde a la fila (a-j) y el número, al correspondiente de esa fila, que puede ir del 1 al 15. Así, en ese estacionamiento tendremos, por ejemplo, lugares a1, b15 ó j7.

    El sistema que maneja los sensores del estacionamiento, mantiene una lista anidada de unos y ceros para identificar los lugares ocupados (1) de los vacíos (0). Cada lista interna representa una fila del estacionamiento.

    Elige una de las siguientes opciones:

    Opción 1: 30 puntos. Ayudemos a los desarrolladores del sistema de estacionamiento con una función llamada **vacíos** que reciba la lista anidada que genera el sistema de sensores y nos devuelva una lista de strings que contiene la nomenclatura de los lugares que se encuentran vacíos. Por ejemplo: [“a5”, “b2”, “d10”, “e2”, “e6”, “f4”, “j1”, “j2”, “j15”] pudiera ser el resultado de la función si sólo quedaran 9 lugares vacíos. Asume que la lista anidada recibida es una lista anidada bien formada de acuerdo con las características descritas.

    Opción 2: 20 puntos. Crear la función **cuantosVacios** que recibe la lista anidada generada por el sistema de sensores. La función te devolverá el número de lugares de estacionamientos vacíos que se tienen.

    c. Elige una de las siguientes opciones:

    Opción 1: 35 puntos: Escribe la función divide, la cual recibe una lista de datos numéricos. La función deberá devolver una lista anidada donde el primer elemento es una lista con los 3 números más pequeños de la lista original y el segundo elemento es la lista con los elementos restantes. Asume que la lista que llega como parámetro tiene al menos 4 elementos.

    Opción 2: 25 puntos: Escribe la función **los_3_menores**, la cual recibe una lista de datos numéricos y te regresa otra lista con los 3 elementos más pequeños de la lista. Asume que la lista que llega como parámetro tiene al menos 4 elementos.

    Restricción: En ambas opciones se deberá implementar con For.

    d. Elige una de las dos opciones:

    Opción 1: (25 puntos) Escribe una función **valida_con_rango**, la cual recibe un string que representa el mensaje para solicitar el dato al usuario y dos números, uno que representa el límite inferior y otro que representa el límite superior de un rango. La función deberá solicitar el dato al usuario (utilizando el mensaje que llega como parámetro) y validar que el dato esté dentro del rango solicitado, además de atender los posibles errores que se puedan presentar por el ingreso del dato a través del teclado. La función debe

devolver el dato validado. Se debe repetir el ingreso del dato hasta que el dato recibido cumpla con el requisito.

Opción 2: (10 puntos) Escribe la función **valida_real**, la cual recibe un string que representa el mensaje para solicitar el dato al usuario. La función deberá solicitar un número al usuario (utilizando el mensaje que llega como parámetro) y validar que el dato introducido por el usuario corresponde a un dato que puede ser representado como un número real, de no ser así, se debe repetir el ingreso del dato a través del teclado. La función deberá devolver el dato recibido cuando éste cumpla con los requisitos.

e. Elige una de las dos opciones:

Tenemos el archivo log.csv, el cual es un archivo de texto que nos ofrece un servidor con algunos datos sobre las conexiones y usuarios. Al ser un csv, los datos están separados por comas.

Los datos y el orden que nos ofrece el archivo es:
usuario,tipo_de_conexión,dirección_ip,mes,dia,hora_inicio_conexión,hora_final_conexión,(tiempo_de_conexión)

Un ejemplo de una línea del archivo es:
pperez,ftp,10.25.25.234,Feb,20,13:24,13:36,(00:12)

Opción 1: 30 puntos. Crea una función llamada **minutos_usuario** que reciba un string que representa el nombre de un usuario. La función debe leer el archivo e imprimir a pantalla un mensaje que nos indique la cantidad de minutos totales que este usuario se mantuvo conectado en el periodo de tiempo que abarca el reporte.
Nota: Si necesitas crear alguna función auxiliar para resolver este problema, puedes hacerlo.

Opción 2: 20 puntos. Crea una función llamada **conexiones**, que recibe un string que representa el nombre de un usuario. La función debe leer el archivo e imprimir a pantalla un mensaje que nos indique la cantidad de veces que se conectó un usuario en el periodo de tiempo que abarca el reporte.