Diagrama

Descripción generada automáticamente con confianza media

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE NUEVO LAREDO

DEPARTAMENTO DE INGENIERIA EN SISTEMAS COMPUTACIONALES

Logotipo

Descripción generada automáticamente con confianza baja

**PRESENTA:**

PRACTICAS DE LA UNIDAD 1

**ALUMNO:**

ALAN PEÑA ORTIZ 19100234

ALAN ABIUD CASTRO CRUZ 19100159

CARLOS DANIEL MORIN RABAGO 19100222

**MATERIA:**

PROGRAMACION MULTIPARADIGMA

**VII SEMESTRE**

**PROFESOR:**

ING. LUIS DANIEL CASTILLO GARCIA

NUEVO LAREDO, TAMPS

FECHA DE ENTREGA: 25 SEP 2022

***Práctica 1:*** ***Funciones con n parámetros***

Escribir un programa que contenga una función que reciba n parámetros de tipo numérico y calcule el producto total.

*Solución*: Definimos una función para el producto total que reciba, mediante un for, variables numéricas para poder realizar el cálculo del producto total con diferentes asserts para verificar la función y no muestre una excepción.

*Sentencia de código:*

def productoTotal(\*numeros):

res = 1

for i in numeros:

res = res \* i

return res

assert productoTotal(1,2,3) == 6

assert productoTotal(1,2,3,0) == 0

print('Pruebas pasadas :)')

*Comprobación*:

Captura de pantalla de computadora

Descripción generada automáticamente

*Explicación del resultado*: Se realiza una funcion que recibe n cantidad de números, dentro de esta funcion se realiza un ciclo que recorre cada uno de estos números que junto al uso de una variable auxiliar que vaya almacenando el resultado, se iran multiplicando sucesivamente en cada iteración.

Esto se retornara a manera de resultado; con el uso de los *assert,* podemos verificar que el programa funcione correctamente.

***Práctica 2:*** ***Manejo y manipulación de elementos de una lista.***

Escribir un programa que almacene el abecedario en una lista, elimine de la lista las letras que ocupen posiciones múltiplos de 3, y muestre por pantalla la lista resultante.

*Solución*: Creamos una lista del abecedario utilizando un range para producir una serie de números y los mapeamos con la función chr para convertir los números enteros en ASCII. Creamos otra función que define las posiciones con múltiplos de 3 de nuestra lista sea falsa y al final verificamos con assert que el resultado sea correcto sin sobrepasar el limite del rango ya establecido.

*Sentencia de código:*

abecedario = [chr(i) for i in range(65,91)]

abecedarioLimpio = [a for i,a in enumerate(abecedario) if i % 3 != 0]

assert len(abecedarioLimpio) == 17

assert len(abecedarioLimpio) < len(abecedario)

print(abecedarioLimpio)

Captura de pantalla de computadora

Descripción generada automáticamente*Comprobación:*

*Explicación del resultado:* Mediante un print, imprimimos nuestra lista ya enumerada para que nos entregue las letras del abecedario que no pertenezcan a los múltiplos del 3 y así analizar que nuestras funciones fueron correctas.

***Práctica 3:*** ***Entrada de datos y manipulación.***

Escribir un programa que permita al usuario capturar su nombre completo e imprima su nombre de manera inversa letra por letra.

*Solución*: Con un input, se le pedirá al usuario que de su nombre para de ahí hacer una lista con su nombre y utilizando el método reverse, nos permitirá en un for dar ese resultado al revés y se muestre su nombre inverso en la terminal letra por letra.

*Sentencia de código:*

nombre = input('Dame tu nombre: ')

nombre = list(nombre)

nombre.reverse()

for i in nombre:

print(i)

*Comprobación:*

Captura de pantalla de computadora

Descripción generada automáticamente

*Explicación del resultado:* Imprimimos la lista inversa para que muestre el nombre del usuario al revés.

***Práctica 4:*** ***Entrada de datos y estructuración.***

Revisar su retícula para escribir un programa que cree un diccionario vacío para que el usuario capture las materias y créditos de su semestre preferido (inferior a 8vo) al final imprimir en el formato “{asignatura}” tiene “{créditos}” créditos. Y la suma de todos los créditos del semestre.

*Solución*: Crearemos distintos menús para ingresar distintos valores, nuestro primer menú detallará si el usuario desea registrar una materia o salir del programa, si decide registrar una materia, utilizaremos inputs para que el usuario agregue la materia que desea y sus créditos.

Una vez que el usuario agrego su materia, volveremos al primer menú para que escoja si desea agregar otra materia o salir del programa. Si selecciona ingresar materia de nuevo, podrá volver a ingresar sus datos según lo que quiera registrar; si selecciona salir, se le mostrará en pantalla las materias que el mismo usuario ingreso con sus respectivos créditos y al final sumará estos créditos para darnos un total.

*Sentencia de código:*

def imprimirMenu() :

print("\n \nSeleccione una opcion del siguiente menu:")

print("[0] - Salir")

print("[1] - Ingresar materia")

return int(input("Valor seleccionado: "));

def agregarMateria() :

mNombre = input("\nIngrese el nombre de la materia: ");

mCreditos = input("Ingrese la cantidad de creditos para la materia: ");

miReticula[mNombre] = int(mCreditos);

print("");

miReticula = {};

selectorDeOpcion = 1;

while(selectorDeOpcion != 0) :

selectorDeOpcion = imprimirMenu();

if selectorDeOpcion == 1 :

agregarMateria();

print("\n++--Programa finalizado--++")

if not miReticula :

print("No se econtraron materias registradas.");

else :

print("Resumen de materias registradas:");

sumaDeCreditos = 0;

for materia in miReticula :

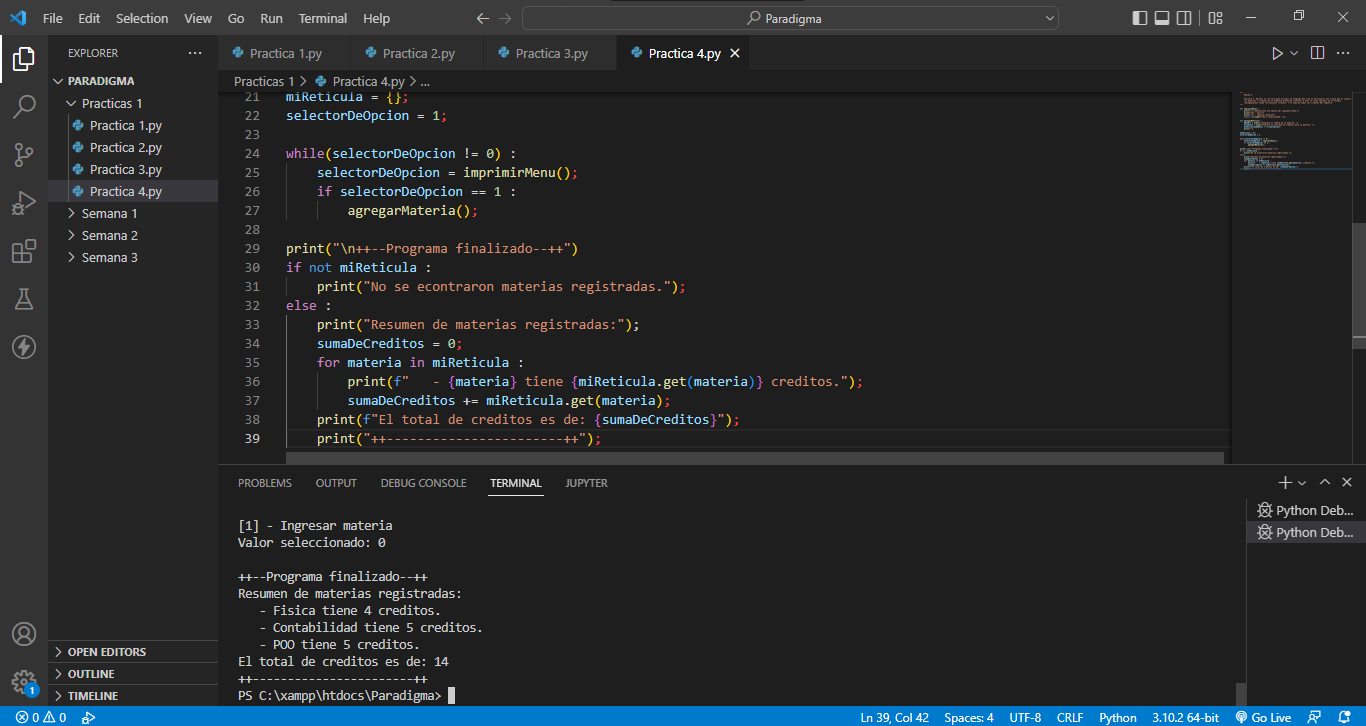
print(f" - {materia} tiene {miReticula.get(materia)} creditos.");

sumaDeCreditos += miReticula.get(materia);

print(f"El total de creditos es de: {sumaDeCreditos}");

print("++-----------------------++");

*Comprobación:*



*Explicación del resultado:* Cuando salimos de nuestra aplicación, se le mostrará al usuario los datos que el mismo ingreso utilizando funciones y parámetros que almacenarán dicha información. Si salimos de la aplicación y el usuario no registro nada, mediante un if se mostrará el mensaje que no se registró ninguna materia.

***Práctica 5: Manejo de información***

Escribir una función que reciba n parámetros de llave valor e imprima la información en formato “{llave}”: “{Valor}”.

*Solución*: Definimos un formato que recibirá las cadenas de nuestro programa utilizando key y value para que sean asignadas según su llave valor y se imprimirá estos parámetros en nuestra terminal con sus llaves correspondientes.

*Sentencia de código:*

def formato(\*\*params):

for key, value in params.items():

print(f' "{key}":"{value}" ')

formato(Nombre = "Alan", Apellido = "Ortiz")

formato(Nombre = "Alan", Apellido = "Castro")

*Comprobación:*

Captura de pantalla de computadora

Descripción generada automáticamente

*Explicación del resultado:* Imprimimos estos parámetros definidos para visualizar que las llaves valor se muestren en la terminal.

***Práctica 6: Razonamiento y prueba de código***

Escribir un programa que reciba un numero entre 0 y 20 e imprimir el número en letra, no utilizar condicionales, máximo 5 líneas de código.

*Solución*: Definimos un diccionario que almacene del 1 al 20 los números y su nombre, después le preguntamos al usuario mediante un input que número desea imprimir para que este imprima el mismo número pero en letra.

*Sentencia de código:*

baseDeDatos = {1: "uno", 2: "dos", 3: "tres", 4: "cuatro", 5: "cinco", 6: "seis", 7: "siete", 8: "ocho", 9: "nueve", 10: "diez", 11: "once", 12: "doce", 13: "trece", 14: "catorce", 15: "quince", 16: "dieciseis", 17: "diecisiete", 18: "dieciocho", 19: "diecinueve", 20: "veinte"};

lectura = int(input("Ingresa un numero del 1 al 20: "));

if lectura > 20 or lectura < 1 :

print("El numero no se encuentra en el rango del 1 al 20.");

else :

print(baseDeDatos.get(lectura));

Captura de pantalla de computadora

Descripción generada automáticamente*Comprobación:*

*Explicación del resultado:* Nos mostrará el número en letra cada vez que el usuario inserte un número del 1 al 20, si esté ingresa un número que no corresponde al rango que se asigno en nuestro programa, mediante un if haremos una validación de su número y si sobrepasa después del 20, se mostrará un mensaje detallando que el número que ingreso no está en el rango que establecimos en nuestro diccionario.

***Práctica 7: Formateo y conversiones***

Escribir un programa que muestre un menú con 2 opciones la primera opción “1.- Imprimir YYYY/MM/DD” la segunda “2.- Imprimir MM/DD/YYYY” una vez seleccionada la opción imprimir la fecha del día de hoy en el formato seleccionado.

*Solución*: Haremos un import al módulo datetime para conseguir la fecha de hoy y para que podamos trabajar sobre lo que pide la práctica en nuestro código. En un menú, el usuario tendrá la opción de escoger como imprimir la fecha de hoy, si en orden de año, mes y día, o en el orden de mes, día y año.

*Sentencia de código:*

from datetime import date

opcion = int(input('1.- Imprimir YYYY/MM/DD\n2.- Imprimir MM/DD/YYYY\n'))

print(date.today().strftime("%Y/%m/%d" if opcion == 1 else "%m/%d/%Y"))

*Comprobación:*

Captura de pantalla de computadora

Descripción generada automáticamente

*Explicación del resultado:* En nuestro print usaremos la fecha de hoy y con strftime podremos convertir esa fecha en un string para que pueda retornarnos ese valor, dentro del print tendremos un if donde verificará que opción escogió el usuario para imprimir la fecha en el orden deseado.

***Práctica 8: Resumen y multi-solución***

8.1.– Definir una clase usuario que contenga como atributos:

Usuario

Contraseña

Rol

Nombre

CURP

Ciudad

8.2.- Realizar un programa que contenga el siguiente menú

1.- Registro

2.- Inicio de sesión

3.- Salida

La opción de registro solicitará al usuario registrarse solicitando la información de los atributos la clase exceptuando el atributo Rol que por defecto será rol cliente, no se permitirán usuarios con CURP repetido en caso de mostrar mensaje de “El usuario ya existe”

La opción de inicio de sesión permitirá al usuario introducir sus credenciales al ser correctas desplegar en pantalla la información del usuario de lo contrario mostrar mensaje de “datos incorrectos “

8.3.- Declarar un usuario con rol “Administrador” el cual al momento de iniciar sesión despliegue la información de todos los usuarios registrados al momento.

*Solución*: Crearemos dos clases en esta práctica, en una clase, se llamará Usuario y tendrá definidos los valores que utilizaremos en nuestro programa, donde el cliente pueda ingresar su usuario, contraseña, nombre, CURP y la ciudad donde vive, este se almacenará en un diccionario para utilizarlo en la otra clase.

En nuestra otra clase, haremos un import de los datos definidos en Usuario y crearemos distintos menús para que el cliente pueda interactuar con su información, en nuestro programa el cliente podrá escoger si desea registrarse, iniciar sesión o salir del programa.

Si escoge registrar usuario, mediante inputs podrá ingresar su información que será necesaria para la clase Usuario (usuario, contraseña, nombre, CURP y ciudad, su rol como cliente ya será asignado automáticamente al registrarse al programa). Una vez registrado tendrá de nuevo la opción de registrar otro usuario, iniciar sesión o salir del programa.

Si escoge iniciar sesión, ingresará su usuario y contraseña correspondiente y podrá ver todos sus datos almacenados en el programa por nuestro diccionario.

Si aquí desea terminar de usar el programa, el usuario podrá escoger la opción de salir y automáticamente el programa dejará de ejecutarse.

*Sentencia de código (Practica8\_1):*

class Usuario:

def \_\_init\_\_(self, username, contrasena, nombre, curp, ciudad, rol = "Cliente"):

self.username = username

self.contrasena = contrasena

self.rol = rol

self.nombre = nombre

self.curp = curp

self.ciudad = ciudad

def \_\_str\_\_(self):

return f"Username: {self.username} Rol: {self.rol} Nombre: {self.nombre} CURP: {self.curp} Ciudad: {self.ciudad}"

*Sentencia de código (Practica8\_2):*

from Practica8\_1 import Usuario

import os

listaUsuarios = {}

def registrarUsuario():

os.system('cls')

print('------------Registro de usuario------------')

username = input('Username: ')

contrasena = input('Password: ')

nombre = input('Nombre real: ')

ciudad = input('Ciudad de nacimiento: ')

curp = input('Curp: ')

nuevoUsuario = Usuario(username, contrasena, nombre, curp, ciudad)

for value in listaUsuarios.values():

if value.curp == curp:

print ('Usuario ya existe en los registros.')

menu()

listaUsuarios[nuevoUsuario.username] = nuevoUsuario

menu()

def iniciarSesion():

os.system('cls')

print('------------Inicio de sesion------------')

username = input('Username: ')

password = input('Password: ')

if username in listaUsuarios:

usuario = listaUsuarios[username]

if usuario.contrasena == password:

if usuario.rol == 'Administrador':

print('----------------Lista de usuarios----------------')

print ("{:<25} {:<25} {:<25} {:<25} {:<25}".format('Username','Nombre','CURP','Rol', 'Ciudad'))

for user in listaUsuarios:

data = listaUsuarios[user]

print ("{:<25} {:<25} {:<25} {:<25} {:<25}".format(data.username, data.nombre, data.curp, data.rol, data.ciudad), end='\n')

menu()

else:

print('----------------Datos de tu usuario----------------')

print(usuario,end='\n')

menu()

else:

print("Datos incorrectos", end = '\n')

menu()

else:

print("Cuenta inexistente", end='\n')

menu()

def menu():

opcion = int(input('1.- Registro\n2.- Inicio de sesion\n3.- Salida\n'))

if opcion not in [1,2,3]:

menu()

else:

if opcion == 1:

registrarUsuario()

elif opcion == 2:

iniciarSesion()

elif opcion == 3:

os.system('cls')

print('Gracias por usar el programa')

exit()

if \_\_name\_\_ == '\_\_main\_\_':

administrador = Usuario('admin', '123', 'Administrador', 'ADMIN001HTSXRLA3', 'Laredo', 'Administrador')

listaUsuarios[administrador.username] = administrador

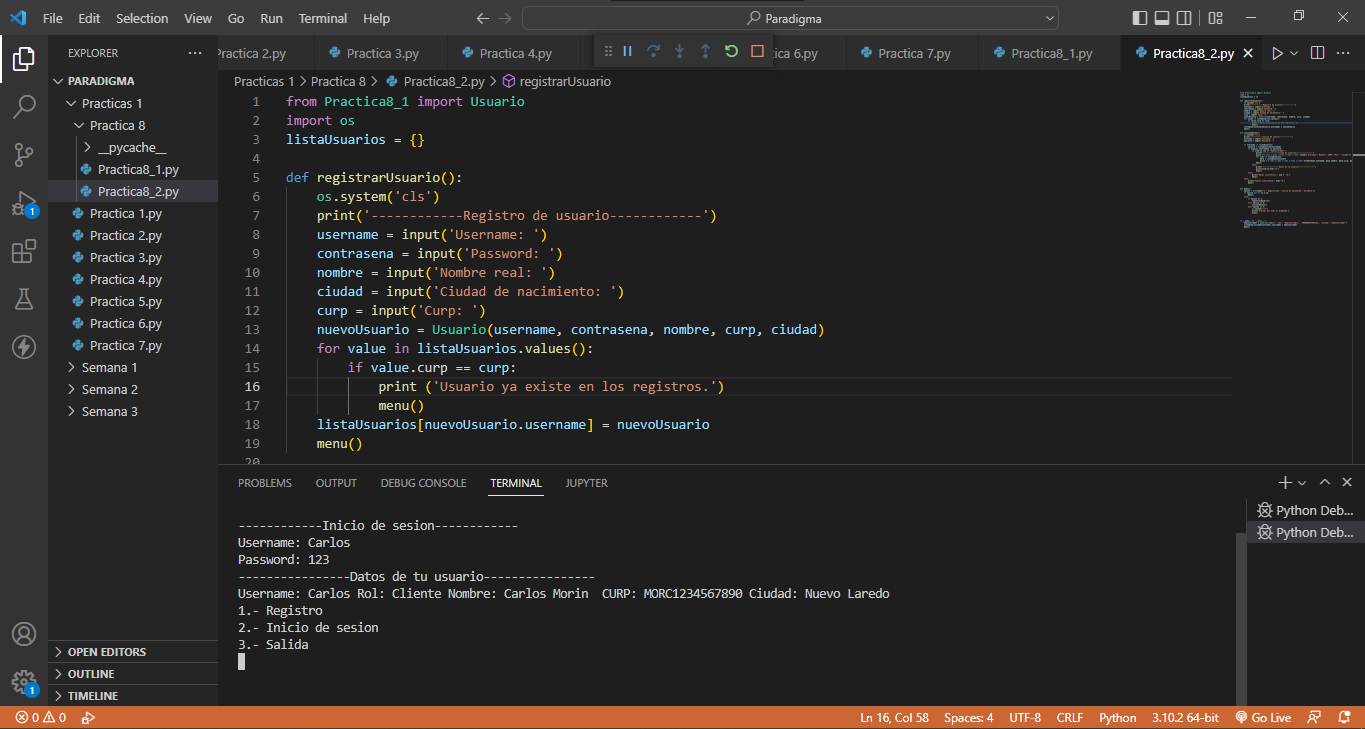
menu()

Captura de pantalla de computadora

Descripción generada automáticamente*Comprobación:*

*Texto

Descripción generada automáticamente*

**

*Captura de pantalla de computadora

Descripción generada automáticamente*

*Explicación del resultado:* Una vez que el cliente registre los usuarios que desea, este podrá verlos en nuestro programa al iniciar sesión según los datos correspondientes con toda la información que el mismo ingreso en el primer menú, habrá excepciones si este ingrese una contraseña incorrecta o si no existe el usuario registrado.

**Enlace a repositorio donde están todas las prácticas:**

<https://github.com/lanazirot/python-teclaredo.git>

Conclusiones y comentarios:

Durante estas prácticas, pudimos analizar diferentes escenarios donde utilizamos la lógica de la programación en el lenguaje Python, desde crear listas, diccionarios, bucles, excepciones, entre otras cosas.

La implementación y desarrollo de la lógica para solucionar problemas mediante algoritmos presentados, es fundamental para cualquier desarrollador de software, y la varianza entre distintos lenguajes es mínima, convirtiéndose en esta la parte más importante para un programador. Ya nos hemos familiarizados a este tipo de actividades de programación en diferentes lenguajes como C# o Javascript, cada uno maneja de diferente forma la solución y uno requerirá un poco más de definición a cada valor que agreguemos o método que utilizamos en las prácticas. Con Python es más sencillo definir métodos y funciones que nos servirá de mucha ayuda para calcular valores o almacenar información que deseamos. Estas prácticas nos servirán para la retroalimentación de como programar y también para conocer Python y sus ventajas que nos brinda para que en un futuro no tengamos dificultad de realizar programas como ingenieros en sistemas computacionales.