HITO 2 DEL 1º TRIMESTRE DE PROGRAMACIÓN

Alejandro Cortés Díaz

CampusFP

Diseño de aplicaciones multiplataforma – programación  20/11/2024

Índice

[**Fase 1.** 2](#_Toc182491266)

[**Fase 2.** 3](#_Toc182491267)

[**Archivo donde ejecuto;** 3](#_Toc182491268)

[**Archivo conexión padre:** 4](#_Toc182491269)

[**Archivo conexión hijo;** 4](#_Toc182491270)

[**Menu Main;** 4](#_Toc182491271)

[**Menu cliente;** 6](#_Toc182491272)

[**Menu producto;** 7](#_Toc182491273)

[**DAW CLIENTE;** 7](#_Toc182491274)

[**DAW producto;** 14](#_Toc182491275)

[**Enlace a GitHub** 24](#_Toc182491276)

[**Bibliografía** 25](#_Toc182491277)

# **Fase 1.**

En este primer paso, se van a definir los pasos necesarios para resolver el registro de un nuevo cliente y cómo realizar una compra, así como diseñar para cada uno de ellos el diagrama de flujo correspondiente. En este primer paso, no es necesario desarrollar la aplicación en ningún lenguaje de programación. Se trata en todo caso, de analizar la organización y métodos de resolución de la aplicación.

En este código se pueden realizar las siguientes operaciones;

Con respecto a cliente;

* Registro de cliente via función ingresar\_cliente()
* Visualizar todos los clientes registrados via función select\_clientes()
* Seleccionar búsqueda de un solo cliente via función select\_cliente()
* Seguimiento de compras de un cliente a través de la id del mismo via función seguimiento\_cliente()

Con respecto al producto;

* Realizar compra de producto via función compra\_producto()
* Seleccionar todos los productos disponibles via función selet\_productos()
* Sacar el precio de un solo producto del interés del usuario por medio de la id via función select\_producto()

# **Fase 2.**

A continuación, nos dedicamos a la implementación de la aplicación. En este punto, sí es necesario utilizar Python para su desarrollo, usando las estructuras de datos que consideres más adecuadas. Debemos tener presente que no se nos pide realizar el aplicativo visual, únicamente la funcionalidad de cada caso. La selección de cada una de las funcionalidades se realizará mediante el uso de un menú.

# **Archivo donde ejecuto;**

Ejecuto el archivo con los archivos “Conexión”, “Menu\_Main (el menú principal)”, y llamo a conexión y cursor de estos mismos archivos, así como, de Menu\_Main, la función menú, cuya utilidad se explica en su correspondiente captura

# IMPORTO CONEXION

import Conexion as Cx

# IMPORTO MAIN MENU

import Menu\_Main as Mm

# ASIGNO CONEXION A CONEXION PARA UTILIZAR SUS FUNCIONES CON LA BASE DE DATOS

conexion = Cx.conexion

# ASIGNO CURSOR A CURSOR PARA HACER EXECUTES

cursor = Cx.cursor

# LLAMO A MENU MAIN PARA PODER USAR CURSOR Y CONEXION EN EL

Mm.Menu(conexion, cursor)

# **Archivo conexión padre:**

En este archivo inicio sesión en mysql

import mysql.connector

def conectar (BaseDatos):

 # Establecer conexión con la base de datos

        conexion = mysql.connector.connect(

            host="localhost",       # Dirección del servidor (localhost para base de datos local)

            user="root",         # Usuario de la base de datos

            password="curso",  # Contraseña del usuario

            database=BaseDatos    # Nombre de la base de datos

        )

        return conexion

# **Archivo conexión hijo;**

Via el archivo conexión, importando el anterior con la sesión iniciada, especifico la conexión a la base de datos del hito, y llamo a cursor para ejecutar sentencias

#IMPORTO DESDE CONEXIÓN PADRE LAS FUNCIONES DE CONEXION

import Conexion\_Padre as bdd

#USO CONEXIÓN PARA CONECTARME A BD

conexion=bdd.conectar("SUPERMERCADOPython")

#EMPLEO CURSOR PARA EJECUTAR

cursor= conexion.cursor()

# **Menu Main;**

Aquí muestro las opciones posibles de las respectivas tablas de las que se permite acceso en el archivo, “Producto” y “Cliente”. Por ello importo los menús de las tablas correspondientes

#IMPORTO LOS DOS MENUS PARA ENLAZARLOS AL MAIN

import Menu\_Producto as Mp

import Menu\_Cliente as Mc

#DEF MENU

def Menu (conexion,cursor):

    try:

        while True:

            #MUESTRO OPCIONES Y ASIGNO CADA UNA A CADA CASO POSIBLE

            opcion=int(input("Menú de opciones\n1/Tabla Producto\n2/Tabla Cliente\n3/Salir\n"))

            match opcion:

                case 1:

                    print("Marchando al menu de Producto")

                    Mp.Menu(conexion,cursor)

                case 2:

                    print("Marchando al menu de Cliente")

                    Mc.Menu(conexion,cursor)

                case 3:

                    print("Saliendo del programa")

                    cursor.close()

                    conexion.close()

                    break

    except ValueError as e:

        #ERROR SI TIPO DE DATO ERRONEO

        print("No aplica ese valor, error{e}")

    except Exception:

        #ERROR SI FALLA ALGUNA COSA DISTINTA

        print("Error al ejecutar la funcion")

# **Menu cliente;**

En este menú la finalidad es la misma, entregar las distintas opciones. En este caso; Ingresar cliente, seleccionar todos los clientes, seleccionar clientes de manera individual, y hacer un seguimiento de los pedidos del cliente

# IMPORTO CLIENTE PARA USARLO EN EL MENU

import DAW\_Cliente as dwC

# DEF MENU

def Menu(conexion, cursor):

    try:

        while True:

            # MUESTRO OPCIONES

            opcion = int(input("Menú de opciones \n1/Insertar nuevo cliente\n2/Seleccionar todos los clientes\n3/Seleccionar un cliente en específico\n4/Seguimiento de cliente \n5/Salir al menu principal"))

            # ASIGNO UNA OPCION A CADA FUNCION CON MATCH CASE CORRESPONDIENTE

            match opcion:

                case 1:

                    print("Insertar cliente")

                    dwC.ingresar\_cliente(conexion, cursor)

                case 2:

                    print("Seleccionar a todos los clientes")

                    dwC.select\_clientes(cursor)

                case 3:

                    print("Seleccionar cliente")

                    dwC.select\_cliente(cursor)

                case 4:

                    print("Seguimiento de cliente")

                    dwC.seguimiento\_cliente(cursor)

                case 5:

                    print("Marchando al menú principal")

                    break

    #ERROR SI TIPO DE DATO NO ES CORRECTO

    except ValueError as e:

        print(f"No aplica ese valor, error {e}")

    except Exception:

        #ERROR SI SE DA OTRO TIPO DE FALLO

        print("Error al ejecutar la función")

# **Menu producto;**

Lo mismo aplica a este menú, las funcionalidades a elegir son; Comprar producto, seleccionar todos los productos, y seleccionar un único producto

# IMPORTO DOW PRODUCTO PARA USARLO EN LOS DISTINTOS CASES

import DAW\_Producto as dwp

# DEFINO MENU

def Menu(conexion, cursor):

    try:

        while True:

            # OPCIONES

            opcion = int(input("Menú de opciones \n1/Comprar producto\n2/Seleccionar todos los productos\n3/Seleccionar un producto en particular\n4/Salir"))

            # SEGUN LA OPCION UNA FUNCION PARA LA NECESIDAD CORRESPONDIENTE

            match opcion:

                case 1:

                    print("Comprar producto")

                    dwp.compra\_producto(conexion, cursor)

                case 2:

                    print("Seleccionar todos los productos")

                    dwp.select\_productos(cursor)

                case 3:

                    print("Seleccionar producto")

                    dwp.select\_producto(cursor)

                case 4:

                    print("Marchando al menu principal")

                    break

    except ValueError as e:

        # ERROR SI EL TIPO DE DATO NO ES VALIDO

        print(f"No aplica ese valor, error {e}")

    except Exception:

        # ERROR PARA CUALQUIER OTRO TIPO DE FALLO

        print("Error al ejecutar la funcion")

# **DAW CLIENTE;**

La función ingresar\_cliente; Introduce los datos solicitados del cliente en la tabla cliente (idcliente es auto\_increment), puesto que idcliente es auto\_increment, lo recojo en base a los datos solicitados para construir su perfil, y le comunico el id que se le ha asignado desde la base de datos. Después, introduzco todos esos datos del perfil del cliente en un archivo txt, previamente manejados por un array.

La función select\_clientes; Emplea una consulta sql para seleccionar todos los clientes ordenándolos por su id específico, los datos asociados a esa id

La función select\_cliente; Pide el idcliente para mostrar los datos asociados al mismo (nombre,ciudad, y teléfono)

La función seguimiento\_cliente; Realiza una consulta sql, y en base a un idcliente ya solicitado. Une las tablas correspondientes para mostrar todos los datos del cliente, el idpedido del pedido o pedidos que haya realizado, y de cada pedido, muestra; El nombre, el precio del producto, y la cantidad comprada de este.

#DEF FUNCION PARA INTRODUCIR CLIENTE A BASE DE DATOS

def ingresar\_cliente(conexion, cursor):

    try:

        # SOLICITO ATRIBUTOS DE LA TABLA CLIENTE

        Nombre = input("Ingrese su nombre: ")

        ciudad = input("Dime la ciudad del nuevo cliente: ")

        tlf = input("Ingrese su número de teléfono: ")

        # LOS INSERTO EN SENTENCIA SQL Y EJECUTO

        consulta = "INSERT INTO cliente(Nombre, ciudad, tlf) VALUES ( %s, %s, %s)"

        cursor.execute(consulta, ( Nombre, ciudad, tlf))

        conexion.commit()

        #EXTRAIGO EL ID EN BASE A LOS DATOS INTRODUCIDOS PARA SACAR EL ID Y MOSTRÁRSELO POSTERIORMENTE

        consulta="select idcliente from cliente where Nombre=%s and ciudad=%s and tlf=%s"

        cursor.execute(consulta,(Nombre,ciudad,tlf))

        resultado=cursor.fetchall()

        for linea in resultado:

                idcliente=linea

                print(f"Posees id:{idcliente}")

        # GUARDO LA INSERCION EN UN ARRAY, Y ESTE LO PRINTEO EN UN ARCHIVO

        array = [ idcliente,Nombre, ciudad, tlf]

        with open("Clientes.txt", "a") as archivo:

            archivo.write("Datos insertados:\n\n")

            for dato in array:

                archivo.write(f"{dato}\n")

        print(f"Se insertó en la base de datos el cliente: {Nombre}\n con el idcliente:{idcliente}\n de la ciudad: {ciudad}\n con el número de télefono:{tlf}")

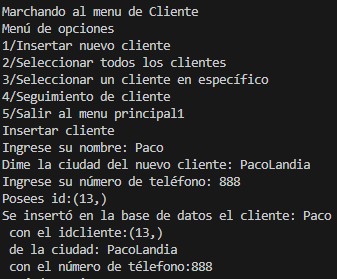
        #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

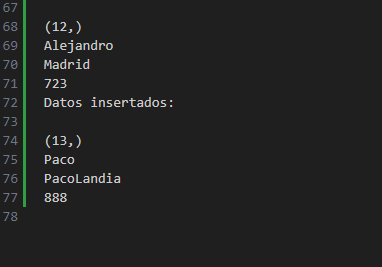
    except ValueError as e:

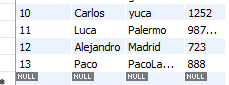
        print(f"Tipo de dato incorrecto, error {e}")

    except Exception as unknown:

        print(f"No pudo ejecutarse la función, error {unknown}")







# DEF FUNCION PARA SELECCIONAR TODOS LOS CLIENTES RECOGIDOS (SOLO HACE FALTA CURSOR PORQUE ES SENTENCIA DE TIPO SELECT)

def select\_clientes(cursor):

    try:

        # CONSULTO A TODOS LOS CLIENTES DE LA TABLA ORDENANDOLOS POR EL ID A TRAVES DE SENTENCIA SQL

        consulta = "SELECT \* FROM cliente ORDER BY idcliente"

        cursor.execute(consulta)

        # PRINTEO A LOS CLIENTES RECOGIDOS

        resultado = cursor.fetchall()

        for fila in resultado:

            print(f"IDCliente: {fila[0]}, Nombre: {fila[1]}, Ciudad: {fila[2]}, Teléfono: {fila[3]}")

        #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

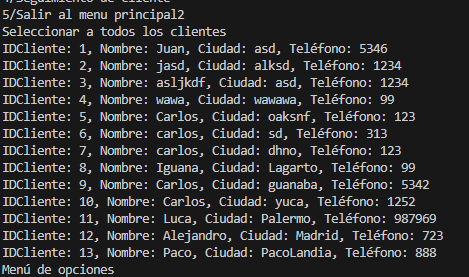
    except ValueError as e:

        print(f"Tipo de dato incorrecto, error {e}")

    except Exception as unknown:

        print(f"No pudo ejecutarse la función, error {unknown}")

# FUNCION PARA SELECCIONAR CLIENTE INDIVIDUAL (SENTENCIA SELECT POR LO QUE NO HACE FALTA CONEXION)



def select\_cliente(cursor):

    try:

        # PIDO ID PARA SACAR SUS DATOS

        idcliente = int(input("Dime el ID del cliente que quieres seleccionar: "))

        # EXTRAIGO CADA DATO DEL CLIENTE CON EL ID DADO

        consulta = "SELECT \* FROM cliente WHERE idcliente = %s ORDER BY idcliente"

        cursor.execute(consulta, (idcliente,))

        # LOS IMPRIMO

        resultado = cursor.fetchall()

        for fila in resultado:

            print(f"IDCliente: {fila[0]}, Nombre: {fila[1]}, Ciudad: {fila[2]}, Teléfono: {fila[3]}")

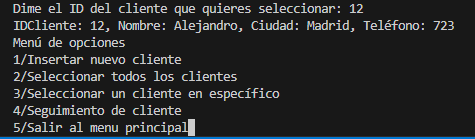
        #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

    except ValueError as e:

        print(f"Tipo de dato incorrecto, error {e}")

    except Exception as unknown:

        print(f"No pudo ejecutarse la función, error {unknown}")



# DEF FUNCION PARA MOSTRAR LOS DATOS DE LAS COMPRAS O PEDIDOS DE UN SOLO CLIENTE

def seguimiento\_cliente(cursor):

    try:

        # PIDO SU ID

        idcliente = int(input("Introduce el ID del cliente para realizar un seguimiento: "))

        # INNER JOINS DE LAS TABLAS DE LAS QUE SELECCIONO TODOS LOS DATOS SOBRE EL CLIENTE Y SUS PEDIDOS

        consulta = """

        SELECT c.idcliente, c.Nombre, c.ciudad, c.tlf, p.idpedido, prod.nombre, d.precio, d.unidades

        FROM cliente c

        JOIN pedido p ON c.idcliente = p.idcliente

        JOIN detalle d ON d.idpedido = p.idpedido

        JOIN producto prod ON prod.idproducto = d.idproducto

        WHERE c.idcliente = %s

        ORDER BY p.idpedido

        """

        #EJECUTO

        cursor.execute(consulta, (idcliente,))

        # LOS PRINTEA

        resultado = cursor.fetchall()

        for fila in resultado:

            print(f"ID Cliente: {fila[0]}")

            print(f"Nombre: {fila[1]}")

            print(f"Ciudad: {fila[2]}")

            print(f"Teléfono: {fila[3]}")

            print(f"Pedido ID: {fila[4]}")

            print(f"Producto: {fila[5]}, Precio: {fila[6]}, Cantidad: {fila[7]}\n")

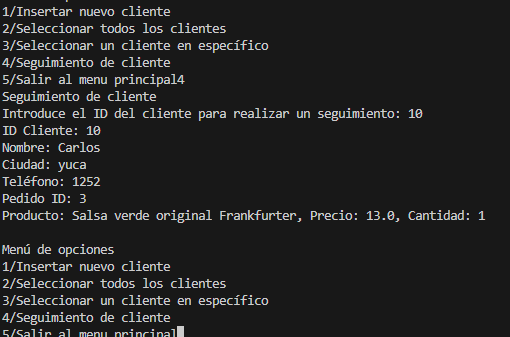
        #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

    except ValueError as e:

        print(f"Error en la entrada de datos: {e}")

    except Exception as unknown:

        print(f"No pudo ejecutarse la función, error: {unknown}")



# **DAW producto;**

La función compra\_producto; Primero comprueba si el cliente ya está registrado en la base de datos, seleccionando los datos del cliente usando el id solicitado para su identificación, lo introduzco en resultado, y en el caso de que sea TRUE, entonces muestra un mensaje de bienvenida, si fuera FALSE, por un error al buscar un cliente inexistente, mostraría un mensaje de error, haciendo referencia a la no existencia de este cliente.

En el caso de que el cliente no se encuentre en la base de datos, le posibilita registrarse directamente introduciendo sus datos y procediendo con ellos, tal y como en la tabla cliente.

Posteriormente, pido el nombre y el id del cliente, para asignarle a esos datos los de compra posteriores, le imprimo toda la lista de productos, con su id, nombre, y precio.

Automáticamente, ingresa su id en la tabla pedido. Luego, pido los datos de unidades y el idproducto que desea comprar, todo esto para, por un lado, servirme de sus respuestas en las variables para rellenar los campos pertinentes de la tabla detalle, puesto que está realizando una compra, y , por otro lado, para introducir esos datos en un txt llamado factura, donde se podrían observar, los datos del movimiento.

Después, le pregunto si desea continuar comprando, de lo contario, termina la función de compra.

Mi función select\_producto; Sirve para saber el precio de un producto en particular, primero le imprimo la lista de los productos, sin el precio, luego le pregunto, de qué producto desea saber el precio, y a continuación procede la impresión con la lista, y el dato deseado.

La función select\_productos; Simplemente via sentencia SQL, solicita todos los datos de la tabla productos, y los ejecuta, luego, por medio de un bucle for, imprime las líneas de todos los productos de la propia tabla.

#DEF FUNNCION PARA LA COMPRA DE PRODUCTO

def compra\_producto (conexion,cursor):

    try:

        while True:

            try:

                #PRIMERO ME ASEGURO DE QUE EXISTA EL CLIENTE QUE VA  A COMPRAR

                cliente=input("Se encuentra en la base de datos? Si/No")

                if cliente.lower() == "si":

                    print("Inicio de sesión")

                    idcliente=int(input("Introduzca su id de cliente"))

                    #SELECCIONO HIPOTETICO CLIENTE CON LA ID QUE ME DAN

                    consulta = "SELECT \* FROM cliente WHERE idcliente = %s"

                    cursor.execute(consulta, (idcliente,))

                    resultado = cursor.fetchone()

                    #SI SE PUEDE REALIZAR LA CONSULTA, LUEGO EXISTE CLIENTE

                    if resultado:

                        print(f"Bienvenido!")

                        break

                    #SINO,

                    else:

                        print("El ID de cliente no existe en la base de datos.")

                #PERMITO REGISTRO SI NO ESTA REGISTRADO EN LA DB

                else:

                    Nombre=input("Ingrese su nombre: ")

                    ciudad=(input("Ingrese su ciudad: "))

                    tlf=input("Ingrese su telefono")

                    #EJECUTO CONSULTA CON LOS DATOS DADOS

                    consulta="insert into cliente(Nombre,ciudad,tlf) values(%s,%s,%s)"

                    cursor.execute(consulta,(Nombre,ciudad,tlf))

                    conexion.commit()

                    #GUARDO AL CLIENTE EN EL TXT COMO ARRAY UNA VEZ SE REGISTRE

                    consulta="select idcliente from cliente where Nombre=%s and ciudad=%s and tlf=%s"

                    cursor.execute(consulta,(Nombre,ciudad,tlf))

                    resultado=cursor.fetchall()

                    #INFORMO AL CLIENTE DE SU ID PARA PODER HACER COMPRA A POSTERIORI

                    for linea in resultado:

                        idcliente=linea

                        print(f"Se te ha asignado el id:{idcliente}")

                    #INTRODUCZCO LOS DATOS EN UN ARRAY Y POSTERIORMENTE UN ARCHIVO

                    array = [idcliente, Nombre, ciudad, tlf]

                    with open("Clientes.txt", "a") as archivo:

                        archivo.write("=== Lista de Clientes ===\n")

                        archivo.write("Datos insertados:\n\n")

                        for dato in array:

                            archivo.write(f"{dato}\n")

                    print(f"Se insertó el cliente: {Nombre}\n con el idcliente: {idcliente}\n la ciudad: {ciudad}\n y el telefono {tlf}")

                    break

             #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

            except ValueError as e:

                print(f"Tipo de dato incorrecto, error {e}")

            except Exception as unknown:

                print(f"No pudo ejecutarse la función, error {unknown}")

        while True:

            try:

                #PIDO DATOS PARA COMPRA

                print("Inicializando compra...")

                idcliente=int(input("Cual es su id de cliente?\n>"))

                Nombre=input("A nombre de quién es la compra?\n>")

                #MUESTRO LISTA DE PRODUCTOS

                lista= "select idproducto,Nombre,precio from producto"

                cursor.execute(lista)

                #PRINTEO PRODUCTOS POR MEDIO DE LISTADO

                resultado=cursor.fetchall()

                for idproducto, nombre,precio in resultado:

                    print(f"IDProducto: {idproducto}, Nombre: {nombre}, Precio: {precio}")

                #INSERT INTO EN PEDIDO ANTES DE COMPRA

                consulta1=("insert into pedido(idcliente) values(%s)")

                cursor.execute(consulta1,(idcliente,))

                conexion.commit()

                #INTRODUCCIÓN DE DATOS NECESARIOS PARA ASIGNARLOS A LOS REGISTROS DE LA TABLA DETALLE

                unidades = int(input("Cuántas unidades deseas comprar del producto al que echo un ojo?"))

                idproducto = int(input("De nuestros productos disponibles, introduce el id de aquel que quiera comprar:\n"))

                # OBTENCIÓN DEL PRECIO DEL PRODUCTO SELECCIONADO

                consulta2 = "select precio from producto where idproducto = %s"

                cursor.execute(consulta2, (idproducto,))

                resultado = cursor.fetchall()

                precio\_compra = resultado[0][0]

                #INSERT INTO EN DETALLE PARA COMPRA

                consulta3=("insert into detalle(idproducto,precio,unidades) values(%s,%s,%s)")

                cursor.execute(consulta3,(idproducto,precio\_compra,unidades))

                conexion.commit()

                print(f"Se realizó la compra del producto con el id: {idproducto} con el coste: {precio\_compra}, y la cantidad: {unidades}")

                array=[f"El cliente con el id:{idcliente}Y el nombre:{Nombre},Realizo la compra del producto con el id:{idproducto} ,A un precio de:{precio\_compra} En la cantidad de :{unidades}"]

                #ARCHIVO TXT PARA INTRODUCIR LOS DATOS DE COMPRA

                with open("Factura.txt", "a") as archivo:

                    archivo.write("=== Factura de Compra ===\n")

                    archivo.write("Detalles de la compra:\n\n")

                    for dato in array:

                        archivo.write(f"{dato}\n")

                #PREGUNTO SI DESEA CONTINUAR CON LA COMPRA

                continuar=input("Deseas seguir comprando? Si/No\n")

                if continuar.lower() =="si":

                    print("Siguiente compra")

                elif continuar.lower() =="no":

                    break

            #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

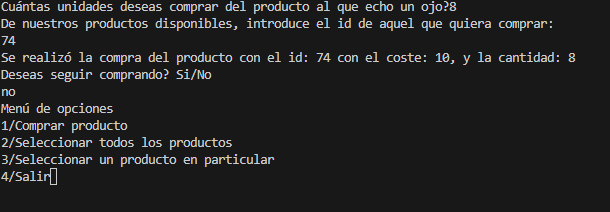
            except ValueError as e:

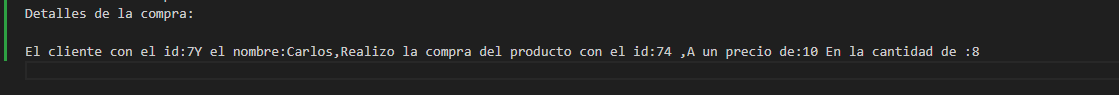
                print(f"Tipo de dato incorrecto, error {e}")

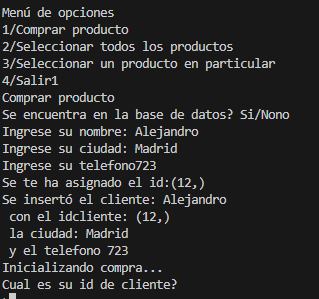
            except Exception as unknown:

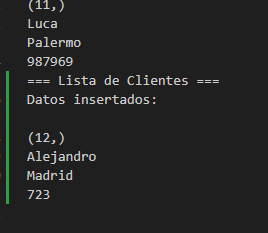
                print(f"No pudo ejecutarse la función, error {unknown}")











#DEF FUNCION PARA SELECCIONAR TODOS LOS PRODUCTOS

def select\_productos (cursor):

    try:

        #EJECUTO SELECT ALL DE PRODUCTOS

        consulta="select \* from producto order by idproducto"

        cursor.execute(consulta)

        #PRINTEA LOS RESULTADOS DE LA CONSULTA

        resultado=cursor.fetchall()

        for fila in resultado:

            print(f"IDProducto: {fila[0]}, Nombre: {fila[1]},  Precio: {fila[2]},")

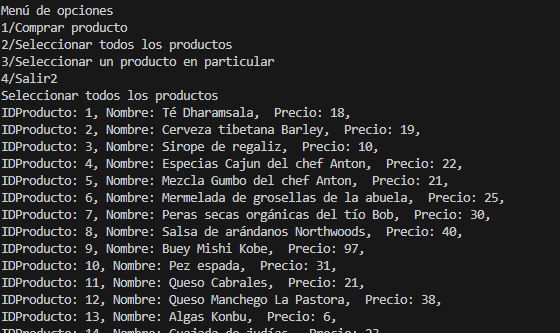
    #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

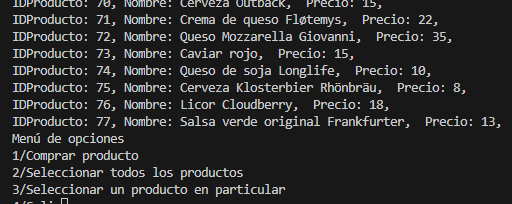
    except ValueError as e:

        print(f"Tipo de dato incorrecto, error {e}")

    except Exception as unkown:

        print(f"No pudo ejecutarse la funcion, error {unkown}")





#DEF FUNCION PARA SELECCIONAR UNO SOLO

def select\_producto (cursor):

    try:

        #EJECUTO SELECT PARA ENSEÑAR TODOS LOS PRODUCTOS Y POSTERIORMENTE ELEGIR Y EJECUTO

        lista= "select idproducto,nombre,precio from producto"

        cursor.execute(lista)

        #PRINTEO RESULTADOS

        resultado=cursor.fetchall()

        for idproducto, nombre,precio in resultado:

            print(f"IDProducto: {idproducto}, Nombre: {nombre}")

        #PREGUNTO EL ID DEL PRODUCTO DEL QUE QUIERE SABER EL PRECIO Y EJECUTO CON EL DATO

        idproducto=int(input("Que id tiene el producto del que deseas informacion?"))

        consulta="select \* from producto where idproducto=%s"

        cursor.execute(consulta,(idproducto,))

        #PRINTEO LOS RESULTADOS ASOCIADOS A LA CONSULTA SQL PREVIA

        resultado=cursor.fetchall()

        for idproducto, nombre,precio in resultado:

            print(f"IDProducto: {idproducto}, Nombre: {nombre}, Precio: {precio}")

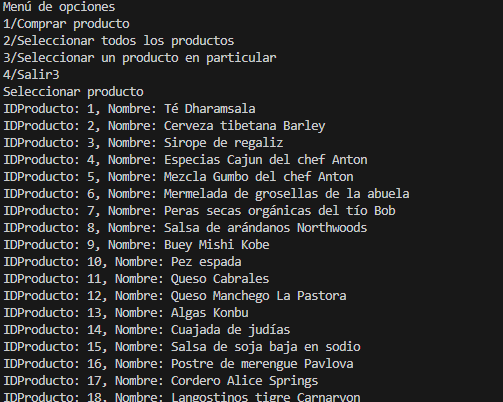
    #LOS MISMOS ERRORES (DATA TYPE ERRONEO Y OTROS ERRORES VARIOS POSIBLES)

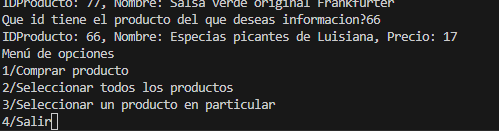
    except ValueError as e:

        print(f"Tipo de dato incorrecto, error {e}")

    except Exception as unkown:

        print(f"No pudo ejecutarse la funcion, error {unkown}")





# **Enlace a GitHub**

<https://github.com/Cortes-cmd/Programacion.git>

# **Bibliografía**

(s/f). El Libro De Python. Recuperado el 14 de noviembre de 2024, de <https://ellibrodepython.com/>

Bravo, B. I. (2023, octubre 2). Lectura y escritura de archivos en Python: Como crear, leer y escribir archivos. freecodecamp.org. <https://www.freecodecamp.org/espanol/news/lectura-y-escritura-de-archivos-en-python-como-crear-leer-y-escribir-archivos/>

Brugués, A. (2019, mayo 12). Leer y Escribir Ficheros de Texto. Programa en Python; Albert Brugués. <https://www.programaenpython.com/fundamentos/leer-y-escribir-ficheros-de-texto/>

ChatGPT. (s/f). Chatgpt.com. Recuperado el 14 de noviembre de 2024, de <https://chatgpt.com/>

Moisset, D. (s/f). Archivos de texto: creación, escritura y lectura. Tutorialesprogramacionya.com. Recuperado el 14 de noviembre de 2024, de <https://www.tutorialesprogramacionya.com/pythonya/detalleconcepto.php?punto=79&codigo=79&inicio=75>

(S/f). Programminghistorian.org. Recuperado el 14 de noviembre de 2024, de <https://programminghistorian.org/es/lecciones/trabajar-con-archivos-de-texto>