

PYTHON

TAREA 3

ALVARO LOZANO

Me ha costado entender cómo hacer que funcione el menú con python que es lo que llevo intentado durante las dos primeras tareas, pero creo que finalmente en esta Tarea 3 lo he conseguido.

//Este es mi código de la Tarea 3

```
class Tarea3:

    # Ejercicio 1
    # Escribir un programa que pregunte al usuario los números
    ganadores de la lotería primitiva, los
    # almacene en una lista y los muestre por pantalla ordenados de
    menor a mayor
    def ejercicio1(self):
        numeros = [] # Creamos una lista vacía
        cantidad = int(
            input("Dime los números ganadores de la Primitiva uno por
            uno y pulsa 'ENTER': "))

        for i in range(cantidad):
            num = int(input(f"Introduce el número {i+1}: "))
            numeros.append(num) # Añadimos el número a la lista

        numeros.sort()

        print("\nLos números ganadores de la Primitiva ordenados de menor
        a mayor son: ")
        print(numeros)

    # Ejercicio 2
    # Escribir un programa que almacene en una lista los números del 1 al
    10 y los muestre por pantalla en orden inverso separados por comas
    def ejercicio2(self):
        # Creamos una lista con los números del 1 al 10
        numeros = list(range(1, 11))
        # Mostramos en orden inverso, separados por comas
        print(", ".join(map(str, numeros[::-1])))

    # Ejercicio 3
    # Escribir un programa que pida al usuario una palabra y muestre por
    pantalla el número de veces que contiene cada vocal
    def ejercicio3(self):
```

```

    # Pedimos al usuario que introduzca una palabra
    # Con .lower lo transformamos todo a minusculas
    palabra = input("Introduce una palabra: ").lower()

    # Inicializamos un diccionario para contar las vocales
    vocales = {'a': 0, 'e': 0, 'i': 0, 'o': 0, 'u': 0}

    # Contar las vocales
    for letra in palabra:
        if letra in vocales:
            vocales[letra] += 1

    # Mostramos resultados
    print("\nNúmero de veces que aparece cada vocal:")
    for vocal, cantidad in vocales.items():
        print(f"{vocal}: {cantidad}")

# Ejercicio 4
# Escribir un programa que almacene en una lista los siguientes precios
# 50, 75, 46, 22, 80, 65, 8, y muestre por pantalla el menor y el mayor
# de los precios

def ejercicio4(self):
    precios = [50, 75, 46, 22, 80, 65, 8] # Lista de precios

    # Calcular el menor y el mayor precio
    menor = min(precios)
    mayor = max(precios)

    # Mostrar por pantalla los resultados
    print(f"El precio más bajo es: {menor}")
    print(f"El precio más alto es: {mayor}")

# Ejercicio 5
# Escribir un programa que guarde en una variable el diccionario:
# 'Euro': '€', 'Dollar': '$', 'Yen': '¥'
# pregunte al usuario por una divisa y muestre su simbolo o un mensaje
# de aviso si la divisa no está en el diccionario

```

```

def ejercicio5(self):
    # Creamos el diccionario
    divisa = {'Euro': '€', 'Dolar': '$', 'Yen': '¥'}

    # Pedimos al usuario que introduzca la divisa
    nombre = input("Introduce la divisa (Euro, Dolar, Yen): ")

    # Comprobamos si está en el diccionario
    if nombre in divisa:
        print(f"El símbolo de {nombre} es: {divisa[nombre]}")

# Ejercicio 6
# Escribir un programa que guarde en un diccionario los precios de las
# frutas de la tabla,
# pregunte al usuario por una fruta, un número de kilos y muestre por
# pantalla el precio de ese
# número de kilos de fruta. Si la fruta no está en el diccionario debe
# mostrar un mensaje
# informando de ello.
# Fruta Precio
# Plátano 1.35
# Manzana 0.80
# Pera 0.85
# Naranja 0.70

def ejercicio6(self):
    # Hacemos nuestro diccionario
    precios = {
        'Plátano': 1.35,
        'Manzana': 0.80,
        'Pera': 0.85,
        'Naranja': 0.70
    }

    # Pedir al usuario una fruta y la cantidad de kilos
    fruta = input(
        "Introduce una fruta (Plátano, Manzana, Pera, Naranja): ")
    kilos = float(input("Introduce la cantidad de kilos: "))

    # Calcular al usuario una fruta y la cantidad de kilos
    if fruta in precios:

```

```

        total = precios[fruta] * kilos
        print(f"El precio de {kilos} kilos de {fruta} es :
{total:.2f}€")
    else:
        print("Lo siento, la fruta no está en el diccionario.")

# Ejercicio 7
# Un programa que cree un diccionario simulando una cesta de la compra.
# El programa debe preguntar el artículo y su precio y añadir el par al
# diccionario, hasta que el usuario
# decida terminar.
# Después se debe mostrar por pantalla la lista de la compra y el coste
# total, con el siguiente formato:
# Lista de la Compra:
# Artículo 1      Precio
# Artículo 2      Precio
# Artículo 3      Precio
# Total           Coste

def ejercicio7(self):
    # Creamos nuestro diccionario
    cesta = {}

    # Pedir artículos y precios al usuario
    while True:
        articulo = input(
            "Introduce el nombre del artículo (o 'fin' para
terminar): ")
        if articulo.lower() == 'fin':
            break
        try:
            precio = float(input(f"Introduce el precio de
{articulo}: "))
            cesta[articulo] = precio
        except ValueError:
            print("Por favor, introduce un precio válido.")

    print("\nLista de la compra:")
    for articulo, precio in cesta.items():
        print(f"{articulo:<15} {precio:>6.2f}€")

```

```
# Ejercicio8
# Escribir un programa que gestione las facturas pendientes de cobro de
una empresa.
# Las facturas se almacenarán en un diccionario donde la clave de cada
factura será el número de factura y el valor el coste de la factura.
# El programa debe preguntar al usuario si quiere añadir una nueva
factura, pagar una existente o terminar.
# Si desea añadir una nueva factura se preguntará por el número de
factura y su coste y se añadirá al diccionario.
# Si se desea pagar una factura se preguntará por el número de factura
y se eliminará del diccionario.
# Después de cada operación el programa debe mostrar por pantalla la
cantidad cobrada hasta el momento y la cantidad pendiente de cobro.
```

```
def ejercicio8(self):
    # Creamos el Diccionario para facturas pendientes de cobro
    facturas = {}

    # Variables para acumular los totales
    cobrado = 0.0
    pendiente = 0.0

    while True:
        opcion = input("\nAñadir factura, cobrar o fin: ")

        if opcion == "fin":
            break

        elif opcion == "añadir":
            numero = input("Introduce el número de factura: ")

            try:
                coste = float(input("Introduce el coste de la
factura: "))

                facturas[numero] = coste
                pendiente += coste
            except ValueError:
                print("Error: introduce un número válido para el
coste.")

        elif opcion == "cobrar":
            numero = input("Introduce el número de factura a
cobrar: ")
```

```

        if numero in facturas:
            cobrado += facturas[numero]
            pendiente -= facturas[numero]
            del facturas[numero]
            print(f"Factura {numero} cobrada correctamente.")
        else:
            print("Opción no válida.")
    else:
        print("Error: Introduce ( añadir, cobrar o fin: )")

    # Mostrar resumen de cobros y pendientes
    print(f"\nCantidad cobrada: {cobrado:.2f}€")
    print(f"\nCantidad pendiente de cobro: {pendiente:.2f}€")

print("\nPrograma finalizado.")
print(f"Total cobrado: {cobrado:.2f}€")
print(f"Total pendiente: {pendiente:.2f}")

# Ejercicio 9
# Escribir un programa que almacene el diccionario con los créditos de
las asignaturas de un curso
# {'Matemáticas': 6, 'Física': 4, 'Química': 5}
# y después muestre por pantalla los créditos de cada asignatura en el
formato:
# <asignatura> tiene <créditos> créditos, donde <asignatura> es cada
una de las asignaturas del curso, y <créditos> son sus créditos.
# Al final debe mostrar también el número total de créditos del curso.

def ejercicio9(self):

    # Creamos el Diccionario
    creditos = {
        'Matematicas': 6,
        'Fisica': 4,
        'Quimica': 5
    }

    # Mostrar los créditos de cada asignatura
    for asignatura, credito in creditos.items():
        print(f"{asignatura} tiene {credito} créditos")

```

```
# Calcular el total de créditos
total_creditos = sum(creditos.values())

print(f"\nEl número total de créditos del curso es:
{total_creditos}")

def menu(self):
    while True:

        print("--- Menú ---")
        print("1° Ejercicio 1: Preguntar al usuario N° ganadores de
la lotería.")
        print(
            "2° Ejercicio 2: Mostrar por pantalla números del 1 al
10 separados por comas.")
        print("3° Ejercicio 3: Contar vocales")
        print("4° Ejercicio 4: Mostrar Precios")
        print("5° Ejercicio 5: Mostrar Símbolos de divisas")
        print("6° Ejercicio 6: Precio de Frutas por Kilo")
        print("7° Ejercicio 7: Cesta de la compra")
        print("8° Ejercicio 8: Facturas pendientes")
        print("9° Ejercicio 9: Créditos de asignaturas")
        print("10° Salir del programa")

        opcion = input("Elige una opción: ")

        if opcion == "1":
            self.ejercicio1()

        elif opcion == "2":
            self.ejercicio2()

        elif opcion == "3":
            self.ejercicio3()

        elif opcion == "4":
            self.ejercicio4()

        elif opcion == "5":
            self.ejercicio5()

        elif opcion == "6":
            self.ejercicio6()
```



```
        elif opcion == "7":
            self.ejercicio7()

        elif opcion == "8":
            self.ejercicio8()

        elif opcion == "9":
            self.ejercicio9()

        elif opcion == "10":
            print("Saliendo del programa...Adios")
            break
        else:
            print("Error: Opción no válida.")

if __name__ == "__main__": # Esta es la clase principal el método main
en Java
    programa = Tarea3() # Creamos un objeto de la clase Tarea3
    programa.menu() # Ejecutamos el método menu, para mostrar el Menú
de ejercicios
```

Lo que no me está gustando de PYTHON es que todo tiene que estar bien tabulado, le da mucha importancia los espacios de la indentación. Java es más estricto en la sintaxis, pero no le da tanta importancia a la tabulación. Es mi opinión.