

Universidad Americana

Facultad de Ingeniería y Arquitectura



Algoritmos y Estructuras de Datos

Grupo 4

Guía Práctica #2: Manejo de Listas y Funciones

Realizado por:

Joaquín Alberto Pérez Zúñiga

Docente:

MSc. César Marín López

Ejercicio #1- Código:

```
1  """
2  Crear un programa que simule la gestión de un inventario en una tienda.
3  Utilizar un menú para agregar, eliminar, modificar y consultar productos
4  en el inventario. Cada producto tendrá un código, nombre, cantidad y precio.
5  """
6
7  from os import system # para limpiar la pantalla
8  from typing import Optional
9
10
11  class Producto:
12      """
13      Clase de datos para representar un producto con:
14      - un código,
15      - un nombre,
16      - una cantidad,
17      - un precio
18      """
19
20      def __init__(
21          self,
22          codigo: int = 0,
23          nombre: str = "",
24          cantidad: int = 0,
25          precio: float = 0.0
26      ):
27          self.codigo = codigo
28          self.nombre = nombre
29          self.cantidad = cantidad
30          self.precio = precio
31
32
33  def pedir_codigo(accion: str) -> int:
34      """
35      Pide el código del producto a mostrar/modificar/eliminar.
36      """
37
38      try:
39          codigo = int(input(f"\nIngrese el código del producto que desea {accion}: "))
40          if codigo < 0:
41              raise ValueError
42
43      except (TypeError, ValueError):
44          input("¡El código debe ser un número entero positivo!")
45          return pedir_codigo(accion) # volver a pedir input
46
47      return codigo
48
49
50  def buscar_producto(codigo: int, inventario: list[Producto]) -> Optional[Producto]:
51      """
52      Busca el índice del producto con el código indicado
53      en el inventario de productos registrados.
54      Retorna -1 si el producto no existe.
55      """
56
57      for producto in inventario:
58          if producto.codigo == codigo:
59              return producto
60
61      return None
62
63
```

```

64 def agregar_producto(inventario: list[Producto]) -> None:
65     """
66     ... Pide los datos de un producto nuevo y lo agrega al inventario.
67     ... """
68
69     print("\nAgregar Producto Nuevo:")
70     print("-----")
71
72     print("\nIngrese los datos del producto:")
73     try:
74         codigo = int(input("Código: "))
75         nombre = input("Nombre: ")
76         cantidad = int(input("Cantidad: "))
77         precio = float(input("Precio (en C$): "))
78
79         if codigo < 0 or cantidad < 0 or precio <= 0:
80             raise ValueError
81
82     except (TypeError, ValueError):
83         input(
84             "\nEl código y la cantidad del producto deben ser números enteros"
85             + " positivos, y el precio debe ser un número real positivo!"
86         )
87
88         system("cls || clear")
89         return agregar_producto(inventario) # volver a pedir input
90
91     inventario.append(Producto(codigo, nombre, cantidad, precio))
92     input("\nProducto agregado exitosamente!")
93     return None
94
95
96 def consultar_producto(codigo: int, inventario: list[Producto]) -> None:
97     """
98     ... Busca un producto específico en el inventario y muestra sus datos.
99     ... """
100
101     producto = buscar_producto(codigo, inventario)
102     if producto is None:
103         input(f"\nNo existe un producto con el código '{codigo}'!")
104         return
105
106     print(f"\nProducto #{producto.codigo}:")
107     print("-----")
108     print(f"Nombre: {producto.nombre}")
109     print(f"Cantidad: {producto.cantidad}")
110     print(f"Precio: {producto.precio}")
111     print("-----")
112
113
114 def mostrar_productos(inventario: list[Producto]) -> None:
115     """
116     ... Llama consultar_producto() para todos los productos registrados.
117     ... """
118
119     print("\nProductos Registrados:")
120     print("-----")
121
122     for producto in inventario:
123         consultar_producto(producto.codigo, inventario)
124
125

```

```

126 def modificar_producto(codigo: int, inventario: list[Producto]) -> None:
127     """
128     ...Permite al usuario ingresar los datos de un producto de nuevo
129     ...para modificarlos (excepto el código, ya que se utiliza
130     ...para identificar los productos).
131     ..."""
132
133     ...producto_modificado = buscar_producto(codigo, inventario)
134     ...if producto_modificado is None:
135     ...|...input(f"\n¡No existe un producto con el código '{codigo}'!")
136     ...|...return None
137
138     ...# eliminar el producto original para sobrescribirlo
139     ...inventario.remove(producto_modificado)
140
141     ...print("\nModificar Producto:")
142     ...print("-----")
143
144     ...consultar_producto(codigo, inventario)
145     ...print()
146
147     ...# pedir nuevos datos
148     ...print("\nIngrese los nuevos datos del producto (dejar en blanco para no modificar):")
149     ...nuevo_nombre = input("Nombre: ")
150     ...nueva_cantidad = input("Cantidad: ")
151     ...nuevo_precio = input("Precio: ")
152
153     ...# los strings vacíos se evalúan como False,
154     ...# entonces si el usuario decide no modificar un atributo,
155     ...# el valor del objeto original permanecerá igual
156
157     ...if nuevo_nombre:
158     ...|... producto_modificado.nombre = nuevo_nombre
159
160     ...try:
161     ...|...if nueva_cantidad:
162     ...|...|... producto_modificado.cantidad = int(nueva_cantidad)
163     ...|...if nuevo_precio:
164     ...|...|... producto_modificado.precio = float(nuevo_precio)
165
166     ...|...if producto_modificado.cantidad < 0 or producto_modificado.precio <= 0:
167     ...|...|... raise ValueError
168
169     ...except (TypeError, ValueError):
170     ...|...input(
171     ...|...|..."\n¡El código y la cantidad del producto deben ser números enteros"
172     ...|...|...+ " positivos, y el precio debe ser un número real positivo!"
173     ...|...|...)
174
175     ...|...system("cls || clear")
176     ...|...return modificar_producto(codigo, inventario)
177
178     ...inventario.append(producto_modificado)
179     ...print(
180     ...|...f"\n¡Datos del producto #{producto_modificado.codigo}"
181     ...|...+ " fueron actualizados exitosamente!"
182     ...|...)
183
184     ...return None
185

```

```

187 def eliminar_producto(codigo: int, inventario: list[Producto]) -> None:
188     """
189     Encuentra el índice del producto a eliminar, y lo
190     quita del inventario de productos.
191     """
192
193     producto = buscar_producto(codigo, inventario)
194     if producto is None:
195         input(f"\n¡No existe un producto con el código '{codigo}'!")
196         return
197
198     print("\nEliminar Producto:")
199     print("-----")
200
201     consultar_producto(codigo, inventario)
202     print()
203
204     confirmacion = input(
205         "\n¿Está seguro que desea eliminar el producto seleccionado? (s/n) "
206     ).lower()
207
208     if confirmacion == "s":
209         inventario.remove(producto)
210         input(f"\nProducto #{producto.codigo} eliminado exitosamente!")
211     elif confirmacion == "n":
212         input("\nDe acuerdo! Regresando al menú principal...")
213     else:
214         input("\nRespuesta inválida! Regresando al menú principal...")
215
216
217 def menu_principal(inventario: list[Producto]) -> None:
218     """
219     Loop principal del programa que le muestra un menú de opciones al usuario.
220     """
221
222     opcion = ""
223     while opcion != "6":
224         system("cls || clear")
225         print("\nGestión de Inventario")
226         print("-----\n")
227         print("1. Agregar producto")
228         print("2. Consultar un producto específico")
229         print("3. Mostrar todos productos registrados")
230         print("4. Modificar producto")
231         print("5. Eliminar producto")
232         print("6. Salir")
233
234         opcion = input("\n-> ")
235
236         match opcion:
237             case "1":
238                 system("cls || clear")
239                 agregar_producto(inventario)
240             case "2":
241                 codigo = pedir_codigo("consultar")
242                 system("cls || clear")
243                 consultar_producto(codigo, inventario)
244                 input("\nPresione 'Enter' para regresar al menú principal...")
245             case "3":
246                 system("cls || clear")
247                 mostrar_productos(inventario)
248             case "4":

```

```

248     .... case "4":
249     ....     codigo = pedir_codigo("modificar")
250     ....     system("cls || clear")
251     ....     modificar_producto(codigo, inventario)
252     .... case "5":
253     ....     codigo = pedir_codigo("eliminar")
254     ....     system("cls || clear")
255     ....     eliminar_producto(codigo, inventario)
256     .... case "6":
257     ....     print("\nSaliendo del programa...\n")
258     .... case _:
259     ....     input("\n¡Opción inválida, intente nuevamente!")
260     .... inventario.sort(key=lambda p: p.codigo)
261
262
263 def main() -> None:
264     .... """
265     .... Ejecución del programa.
266     .... """
267
268     .... inventario: list[Producto] = []
269     .... menu_principal(inventario)
270
271
272 if __name__ == "__main__":
273     .... main()
274

```

Ejercicio #1 – Ouput:

```
Gestión de Inventario
```

```
-----
```

1. Agregar producto
2. Consultar un producto específico
3. Mostrar todos productos registrados
4. Modificar producto
5. Eliminar producto
6. Salir

```
-> █
```

```
Agregar Producto Nuevo:
```

```
-----
```

```
Ingrese los datos del producto:
```

```
Código: 1
```

```
Nombre: Calculadora
```

```
Cantidad: 3
```

```
Precio (en C$): 123.4
```

```
¡Producto agregado exitosamente!█
```

```
Productos Registrados:
```

```
-----
```

```
Producto #1:
```

```
-----
```

```
Nombre: Calculadora
```

```
Cantidad: 3
```

```
Precio: 123.4
```

```
-----
```

```
Producto #2:
```

```
-----
```

```
Nombre: Lapicero
```

```
Cantidad: 12
```

```
Precio: 40.5
```

```
-----
```

```
Presione 'Enter' para regresar al menú principal...█
```

Gestión de Inventario

1. Agregar producto
2. Consultar un producto específico
3. Mostrar todos productos registrados
4. Modificar producto
5. Eliminar producto
6. Salir

-> 2

Ingrese el código del producto que desea consultar: 2

Producto #2:

Nombre: Lapicero

Cantidad: 12

Precio: 40.5

Presione 'Enter' para regresar al menú principal...

Gestión de Inventario

1. Agregar producto
2. Consultar un producto específico
3. Mostrar todos productos registrados
4. Modificar producto
5. Eliminar producto
6. Salir

-> 4

Ingrese el código del producto que desea modificar: 2

Modificar Producto:

Producto #2:

Nombre: Lapicero

Cantidad: 12

Precio: 40.5

Ingrese los nuevos datos del producto (dejar en blanco para no modificar):

Nombre:

Cantidad: 5

Precio:

¡Datos del producto #2 fueron actualizados exitosamente!

Productos Registrados:

Producto #1:

Nombre: Calculadora

Cantidad: 3

Precio: 123.4

Producto #2:

Nombre: Lapicero

Cantidad: 5

Precio: 40.5

Presione 'Enter' para regresar al menú principal...

Gestión de Inventario

-
1. Agregar producto
 2. Consultar un producto específico
 3. Mostrar todos productos registrados
 4. Modificar producto
 5. Eliminar producto
 6. Salir

-> 5

Ingrese el código del producto que desea eliminar: 1

Eliminar Producto:

Producto #1:

Nombre: Calculadora

Cantidad: 3

Precio: 123.4

¿Está seguro que desea eliminar el producto seleccionado? (s/n) s

¡Producto #1 eliminado exitosamente!

Productos Registrados:

Producto #2:

Nombre: Lapicero

Cantidad: 12

Precio: 40.5

Presione 'Enter' para regresar al menú principal...

Gestión de Inventario

-
1. Agregar producto
 2. Consultar un producto específico
 3. Mostrar todos productos registrados
 4. Modificar producto
 5. Eliminar producto
 6. Salir

-> 6

○ Saliendo del programa...

C:\dev\algo-structs>

Ejercicio #2 – Código:

```
1  """
2  Desarrollar un programa que permita al usuario gestionar una cuenta bancaria.
3  El programa deberá utilizar un menú que permita realizar diferentes operaciones
4  sobre el saldo de la cuenta. El programa debe permitir al usuario ingresar la
5  cantidad para depositar o retirar, actualizar el saldo y mostrar los resultados.
6  Si se elige la opción de retiro, debe verificar que el saldo sea suficiente.
7  """
8
9  from os import system
10
11
12  def consultar_saldo(saldo: float) -> None:
13      """
14      Muestra el saldo de la cuenta del usuario.
15      """
16
17      print("\nEstado de Cuenta:")
18      print("-----")
19      print(f"Saldo actual: {saldo:.2f}")
20      input("\nPresione 'Enter' para regresar al menú principal...")
21
22
23
24  def pedir_monto(accion: str) -> float:
25      """
26      Le pide un monto al usuario para realizar
27      una acción con su cuenta bancaria.
28      """
29
30      try:
31          monto = float(input(f"\nIngrese el monto que desea {accion}: "))
32          if monto <= 0:
33              raise ValueError
34      except (TypeError, ValueError):
35          input("\n¡El monto debe ser un número real positivo!")
36          return pedir_monto(accion)
37
38      return monto
39
40
41  def depositar(saldo: float) -> float:
42      """
43      Deposita el monto deseado en la cuenta del usuario.
44      """
45
46      saldo += pedir_monto("depositar")
47      input("\n¡Monto depositado exitosamente!")
48      return saldo
49
50
```

```

49
50 def retirar(saldo: float) -> float:
51     """
52     Retira el monto deseado de la cuenta del usuario.
53     """
54
55     monto = pedir_monto("retirar")
56
57     if monto > saldo:
58         input(
59             "\n¡Error! Monto ingresado es mayor que el saldo"
60             + " actual, por favor intente nuevamente..."
61         )
62     else:
63         saldo -= monto
64         input("\n¡Monto retirado exitosamente!")
65
66     return saldo
67
68
69 def menu_principal(saldo: float) -> None:
70     """
71     Loop principal del programa que le muestra un menú de opciones al usuario.
72     """
73
74     opcion = ""
75     while opcion != "4":
76         system("cls || clear")
77         print("\nGestión de Cuenta Bancaria")
78         print("-----\n")
79         print("1. Consultar saldo")
80         print("2. Depositar")
81         print("3. Retirar")
82         print("4. Salir")
83

```

```

85
86     .... match opcion:
87     ....     case "1":
88     ....         system("cls || clear")
89     ....         consultar_saldo(saldo)
90     ....     case "2":
91     ....         saldo = depositar(saldo)
92     ....     case "3":
93     ....         saldo = retirar(saldo)
94     ....     case "4":
95     ....         print("\nSaliendo del programa...\n")
96     ....     case _:
97     ....         input("\n¡Opción inválida, intente nuevamente!")
98
99
100 def main() -> None:
101     .... """
102     .... Ejecución del programa.
103     .... """
104
105     .... saldo = 0.0
106     .... menu_principal(saldo)
107
108
109 if __name__ == "__main__":
110     .... main()
111

```

Ejercicio #2 – Output:

```
Gestión de Cuenta Bancaria
```

```
-----
```

1. Consultar saldo
2. Depositar
3. Retirar
4. Salir

```
-> 
```

```
Estado de Cuenta:
```

```
-----
```

```
Saldo actual: 0.00
```

```
Presione 'Enter' para regresar al menú principal... 
```

```
Gestión de Cuenta Bancaria
```

```
-----
```

1. Consultar saldo
2. Depositar
3. Retirar
4. Salir

```
-> 2
```

```
Ingrese el monto que desea depositar: 1234.5
```

```
¡Monto depositado exitosamente! 
```

```
Estado de Cuenta:
```

```
-----
```

```
Saldo actual: 1234.50
```

```
Presione 'Enter' para regresar al menú principal... 
```

```
Gestión de Cuenta Bancaria
```

```
-----
```

1. Consultar saldo
2. Depositar
3. Retirar
4. Salir

```
-> 3
```

```
Ingrese el monto que desea retirar: 2345.6
```

```
¡Error! Monto ingresado es mayor que el saldo actual, por favor intente nuevamente... 
```

Gestión de Cuenta Bancaria

1. Consultar saldo
2. Depositar
3. Retirar
4. Salir

-> 3

Ingrese el monto que desea retirar: 678.9

¡Monto retirado exitosamente!

Estado de Cuenta:

Saldo actual: 555.60

Presione 'Enter' para regresar al menú principal...

Gestión de Cuenta Bancaria

1. Consultar saldo
2. Depositar
3. Retirar
4. Salir

-> 4

Saliendo del programa...

○ C:\dev\algo-structs>

Ejercicio #3 – Código:

```
1  """
2  Desarrollar un programa que se comporte como un diccionario Inglés-Español;
3  esto es, solicitará una palabra en inglés y escribirá la correspondiente
4  palabra en español. Para hacer más sencillo el ejercicio, el número de
5  parejas de palabras estará limitado a 5. Una vez finalizada la introducción
6  de las listas de palabras, pasamos al modo traducción, de forma que si
7  introducimos "green", la respuesta ha de ser "verde". Si la palabra
8  no se encuentra, se emitirá un mensaje que lo indique.
9  """
10
11  from os import system
12
13
14  def crear_diccionario(diccionario: dict[str, str], limite: int = 5) -> dict[str, str]:
15      """
16      Crea un diccionario de palabras en inglés
17      asociadas con su traducción al español.
18      """
19
20      if len(diccionario.keys()) == limite:
21          return diccionario
22
23      try:
24          num_palabras = int(
25              input("\n¿Cuántas palabras desea ingresar al diccionario? (máx. 5) ")
26          )
27
28          if num_palabras > 5:
29              raise ValueError
30      except (TypeError, ValueError):
31          input(
32              "\n¡Error! El tamaño máximo del diccionario es de 5 palabras;"
33              + " por favor intente nuevamente..."
34          )
35
36      return diccionario
37
38  for i in range(num_palabras):
39      print(f"\n\nPalabra #{i + 1}:")
40      print("-----")
41      ingles = input("Inglés: ")
42      espanol = input("Traducción a español: ")
43      diccionario[ingles] = espanol
44
45  input("\nPresione 'Enter' para regresar al menú principal...")
46  return diccionario
47
48
```



```

48 def traducir(diccionario: dict[str, str]) -> None:
49     """
50     Pide una palabra e imprime su traducción a la pantalla.
51     """
52
53     palabra = input("\nIngrese la palabra (en inglés) que desea traducir: ")
54     if palabra not in diccionario.keys():
55         input(
56             "\n¡Error! La palabra ingresada no se encuentra en el diccionario;"
57             + " por favor intente nuevamente..."
58         )
59     else:
60         print(f"\nInglés: {palabra}\nEspañol: {diccionario[palabra]}")
61         input("\nPresione 'Enter' para regresar al menú principal...")
62
63 def menu_principal(diccionario: dict[str, str], limite: int = 5) -> None:
64     """
65     Loop principal del programa que le muestra un menú de opciones al usuario.
66     """
67
68     opcion = ""
69     while opcion != "6":
70         system("cls || clear")
71         print("\nDiccionario Inglés-Español:")
72         print("-----\n")
73         print("1. Introducir palabras")
74         print("2. Traducir")
75         print("3. Salir")
76
77         opcion = input("\n-> ")
78
79         match opcion:

```

```

78
79     ....match opcion:
80     ....case "1":
81     ....    ....system("cls || clear")
82     ....    ....crear_diccionario(diccionario, limite)
83     ....case "2":
84     ....    ....system("cls || clear")
85     ....    ....traducir(diccionario)
86     ....case "3":
87     ....    ....print("\nSaliendo del programa...\n")
88     ....case _:
89     ....    ....input("\n¡Opción inválida, intente nuevamente!")
90
91
92 def main() -> None:
93     .... """
94     .... Ejecución del programa.
95     .... """
96
97     .... limite_palabras = 5
98     .... diccionario: dict[str, str] = {}
99
100    .... menu_principal(diccionario, limite_palabras)
101
102
103 if __name__ == "__main__":
104     .... main()
105

```

Ejercicio #3 – Output:

Diccionario Inglés-Español:

- 1. Introducir palabras
- 2. Traducir
- 3. Salir

-> 1

¿Cuántas palabras desea ingresar al diccionario? (máx. 5) 5

Palabra #1:

Inglés: green

Traducción a español: verde

Palabra #2:

Inglés: bottle

Traducción a español: botella

Palabra #3:

Inglés: keyboard

Traducción a español: teclado

Palabra #4:

Inglés: screen

Traducción a español: pantalla

Palabra #5:

Inglés: window

Traducción a español: ventana

Presione 'Enter' para regresar al menú principal...

Ingrese la palabra (en inglés) que desea traducir: notebook

¡Error! La palabra ingresada no se encuentra en el diccionario; por favor intente nuevamente...

Ingrese la palabra (en inglés) que desea traducir: keyboard

Inglés: keyboard

Español: teclado

Presione 'Enter' para regresar al menú principal...

Ingrese la palabra (en inglés) que desea traducir: window

Inglés: window

Español: ventana

Presione 'Enter' para regresar al menú principal...