

Sistema de gestión de productos.

(26 de Noviembre, 2024)

Participantes: Johan Yesid Ramírez Torres, Néstor Iván Castellanos Merchán

Resumen: En el presente documento se realiza la respectiva documentación en Formato IEEE del punto D de laboratorio 2, el fin de este documento es presentar la metodología de dicho ejercicio.

I. INTRODUCCIÓN

En el presente formato encontraremos la solución del Punto D de laboratorio 2. El presente punto es referente a la simulación de un gestor de inventarios.

II. ANÁLISIS

Se tomarán en cuenta una serie de pasos y organización para la solución del ejercicio, es por ende de que como en la solución de este es poder gestionar la existencia de material para la venta de un local y/o empresa

- Contexto: Se realiza el punto D de laboratorio que consiste en un gestor de inventarios
- Población: Para usuarios que busquen dar un análisis para sus empresas de que material existente tienen presente en su bodega
- Limitación y Alcance: Usuarios interesados en gestionar la existencia de mercancía guardada

PROGRAMACION MODULAR

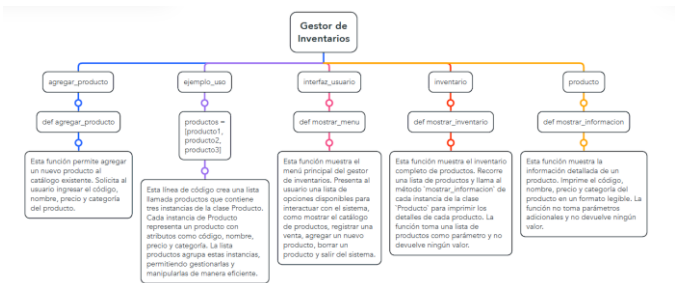


Figura 1: ProgramacionModular (Imagen fuente)

En la figura 1 se muestra la el mapa de programacion modular, la cual sera la estructura inicial para trabajar sobre el ejercicio D de laboratorio 2.

III. OBJETIVOS

- General:
En el presente formato es dar información compleja sobre la realización de dicho ejercicio, e implementarlo y documentar respectivas funciones del código.
- Específicos:
 - Presentar un código preciso en la codificación totalmente funcional.
 - Enfatizar el uso respectivo de matrices y solución de errores en la codificación.
 - Dar en la metodología de problemas las fases completas.

IV. CODIFICACIÓN

A. Tipos de aplicación

En el presente software a realizar se tendrá en cuenta en la fase de Análisis de la Programación Modular y los conceptos completos respectivos a las Matrices y bucles para la verificación de valores respectivos.

Python:

En el presente lenguaje de programación “Python” se dará la realización del Punto D de laboratorio 2, se utilizarán funciones y matrices para su respectiva solución.



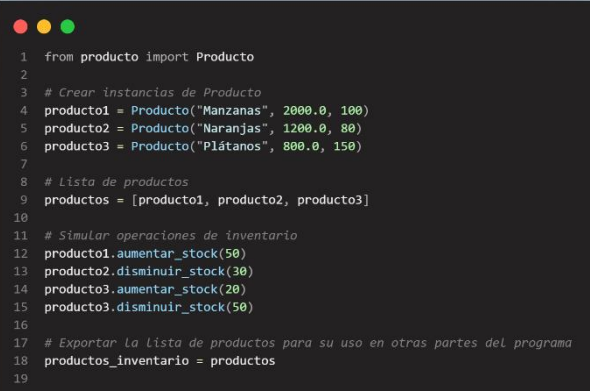
```

1 from producto import Producto
2
3 def agregar_producto(productos):
4     nombre = input("Ingrese el nombre del producto: ")
5     precio = float(input("Ingrese el precio del producto: "))
6     cantidad = int(input("Ingrese la cantidad del producto: "))
7     nuevo_producto = Producto(nombre, precio, cantidad)
8     productos.append(nuevo_producto)
9     print(f"Producto {nombre} añadido al inventario.")
10

```

Figura 2: Codificacion_Python1 (Imagen fuente)

- En la figura 2 encontraremos la función “def agregar_producto”, Esta función permite agregar un nuevo producto al catálogo existente. Solicita al usuario ingresar el código, nombre, precio y categoría del producto. Luego, crea una instancia de la clase Producto con los datos ingresados y la añade a la categoría correspondiente en el catálogo. Si la categoría no existe, se crea una nueva.



```

1 from producto import Producto
2
3 # Crear instancias de Producto
4 producto1 = Producto("Manzanas", 2000.0, 100)
5 producto2 = Producto("Naranjas", 1200.0, 80)
6 producto3 = Producto("Plátanos", 800.0, 150)
7
8 # Lista de productos
9 productos = [producto1, producto2, producto3]
10
11 # Simular operaciones de inventario
12 producto1.aumentar_stock(50)
13 producto2.disminuir_stock(30)
14 producto3.aumentar_stock(20)
15 producto3.disminuir_stock(50)
16
17 # Exportar la lista de productos para su uso en otras partes del programa
18 productos_inventario = productos
19

```

Figura 3: Codificacion_Python2 (Imagen fuente)

- En la figura 3 el código es útil para gestionar un inventario de productos, permitiendo crear instancias de productos, agruparlos en una lista, simular operaciones de inventario y exportar la lista

de productos para su uso posterior. Es una solución básica y eficiente para la gestión de inventarios en una tienda o negocio.



```

1 from inventario import mostrar_inventario
2 from ejemplo_uso import productos_inventario
3 from agregar_producto import agregar_producto
4
5 def mostrar_menu():
6     print("\nSistema de Gestión de Productos")
7     print("1. Mostrar inventario")
8     print("2. Aumentar stock")
9     print("3. Disminuir stock")
10    print("4. Agregar nuevo producto")
11    print("5. Salir")
12
13 def seleccionar_producto():
14    print("\nSeleccione un producto:")
15    for i, producto in enumerate(productos_inventario):
16        print(f"{i + 1}. {producto.nombre}")
17    seleccion = int(input("Ingrese el número del producto: ")) - 1
18    return productos_inventario[seleccion]
19
20 def realizar_operacion():
21    while True:
22        mostrar_menu()
23        opcion = int(input("Seleccione una opción: "))
24        if opcion == 1:
25            mostrar_inventario(productos_inventario)
26        elif opcion == 2:
27            producto = seleccionar_producto()
28            cantidad = int(input("Ingrese la cantidad a aumentar: "))
29            producto.aumentar_stock(cantidad)
30        elif opcion == 3:
31            producto = seleccionar_producto()
32            cantidad = int(input("Ingrese la cantidad a disminuir: "))
33            producto.disminuir_stock(cantidad)
34        elif opcion == 4:
35            agregar_producto(productos_inventario)
36        elif opcion == 5:
37            print("Gracias por usar el Sistema de Gestión de Productos. ¡Hasta luego!")
38            break
39        else:
40            print("Opción no válida. Intente de nuevo.")
41
42 # Iniciar la interfaz de usuario
43 realizar_operacion()
44

```

Figura 4: Codificacion_Python3 (Imagen fuente)

- En la figura 4 El código comienza importando las funciones necesarias y la lista de productos desde módulos externos. La función mostrar_menu presenta al usuario un menú con las opciones disponibles. La función seleccionar_producto permite al usuario elegir un producto del inventario mostrando una lista numerada de productos. La función realizar_operacion gestiona las operaciones del sistema, mostrando el menú, leyendo la opción seleccionada por el usuario y ejecutando la operación correspondiente, como mostrar el inventario, aumentar o disminuir el stock, agregar un nuevo producto o salir del sistema. Finalmente, el sistema inicia la interfaz de usuario llamando a la función realizar_operacion.

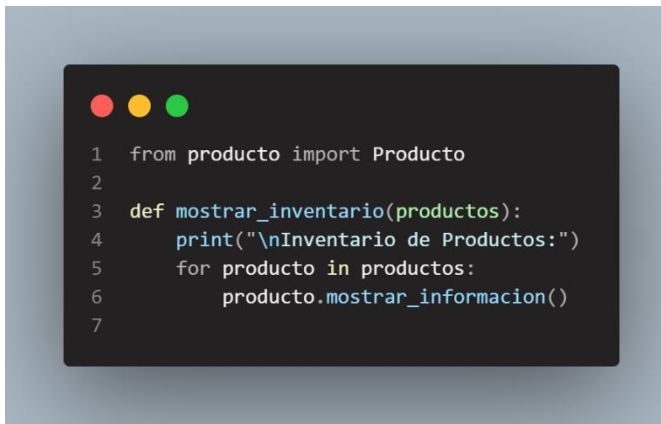


Figura5: Codificacion_Python4 (Imagen fuente)

- En la figura 5 implementa una función para mostrar el inventario de productos utilizando la clase Producto. La función `mostrar_inventario` recorre una lista de productos y llama al método `mostrar_informacion` de cada instancia de la clase Producto para imprimir los detalles de cada producto. La función toma una lista de productos como parámetro y no devuelve ningún valor. Este código es útil para visualizar el inventario actual de productos de manera organizada y detallada, permitiendo al usuario ver la información de cada producto en el inventario, incluyendo el nombre, precio y cantidad disponible.

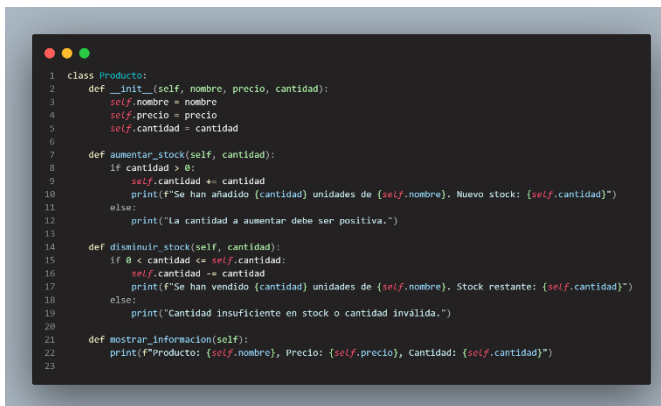


Figura6: Codificacion_Python4 (Imagen fuente)

- Este código define una clase `Producto` que se utiliza para gestionar productos en un inventario. La clase incluye métodos para aumentar y disminuir el stock de un producto, así como para mostrar la información del producto. Además, se implementa una función para mostrar el inventario completo de productos. El sistema permite al usuario realizar diversas operaciones sobre el inventario, como aumentar o disminuir el stock de productos y agregar nuevos productos. Es una solución básica y eficiente para la gestión de inventarios en una tienda o negocio, proporcionando una forma organizada y detallada de mantener el inventario actualizado.

V. REQUERIMIENTOS DEL SISTEMA

- Para teléfono:
Android: Versión 10
RAM: 2GB
Almacenamiento: 500Mb
- Para Ordenador:
Windows 7 o superior
RAM: 4GB
Almacenamiento: 500Mb

VI. DISEÑO DEL ALGORITMO

En el presente punto se da a conocer el diseño de algoritmo teniendo presente su proceso y explicación, en concreto con la programación modular donde esta misma es sutil en cuanto la estructura en la codificación.

VII. EJECUCIÓN DEL PROGRAMA

En la correspondiente ejecución del programa se establece sin novedad su funcionamiento como tal, siendo este logrando conseguir que se logre de antemano su finalidad bajo los parámetros y conclusión de errores.

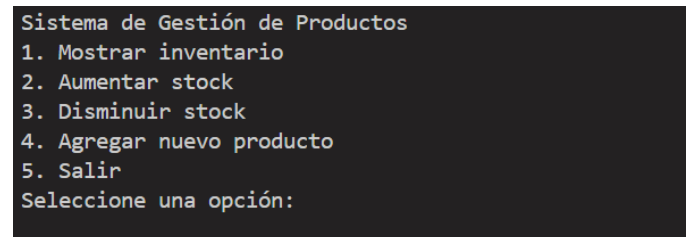


Figura 7: EjecucionPython1 (Imagen fuente)

- En la figura 7 encontramos la ejecución del programa. Se muestra el menú del usuario con 5 opciones
- Mostrar inventario
- Aumentar Stock
- Disminuir Stock
- Agregar nuevo producto
- Salir

Sistema de Gestión de Productos

1. Mostrar inventario
2. Aumentar stock
3. Disminuir stock
4. Agregar nuevo producto
5. Salir

Seleccione una opción: 3

Seleccione un producto:

1. Manzanas
2. Naranjas
3. Plátanos

Ingrese el número del producto: █

Figura 8: EjecucionPython2 (Imagen fuente)

- En la figura 8 encontramos la ejecución del menú número 3 la cual su función es disminuir el stock de lo que deseamos

Sistema de Gestión de Productos

1. Mostrar inventario
2. Aumentar stock
3. Disminuir stock
4. Agregar nuevo producto
5. Salir

Seleccione una opción: 2

Seleccione un producto:

1. Manzanas
2. Naranjas
3. Plátanos

Ingrese el número del producto: 1

Ingrese la cantidad a aumentar: 100

Se han añadido 100 unidades de Manzanas. Nuevo stock: 250

Figura 9: EjecucionPython3 (Imagen fuente)

- En la figura 9 encontramos en funcionalidad la opción del menú número 2 la cual muestra el funcionamiento del aumento del stock del inventario

Sistema de Gestión de Productos

1. Mostrar inventario
2. Aumentar stock
3. Disminuir stock
4. Agregar nuevo producto
5. Salir

Seleccione una opción: 4

Ingrese el nombre del producto: pera

Ingrese el precio del producto: 1200

Ingrese la cantidad del producto: 100

Producto pera añadido al inventario.

Figura 10: EjecucionPython4 (Imagen fuente)

- En la figura 10 encontramos la opción 4 para agregar un producto nuevo, el cual nos pedirá el nombre, precio y cantidad para registrarlo

Sistema de Gestión de Productos

1. Mostrar inventario
2. Aumentar stock
3. Disminuir stock
4. Agregar nuevo producto
5. Salir

Seleccione una opción: 5

Gracias por usar el Sistema de Gestión de Productos. ¡Hasta luego!

PS C:\Users\User\Documents\Gestor de Inventario> █

Figura 11: EjecucionPython4 (Imagen fuente)

- En la figura 11 encontramos la parte final de la opción 2 de generar temperaturas con las temperaturas frías e inclusive la opción número 4 de Salir la cual finaliza el programa realizado.

VIII. CONCLUSIÓN

- En la respectiva documentación se da por finalizado el formato IEEE referente al punto D de laboratorio 2, respectiva documentación se encuentra la programación modular y codificación completa dada en el lenguaje de programación de Python.