**UNIVERSIDAD GERARDO BARRIOS**

SAN MIGUEL

FACULTAD DE CIENCIA Y TECNOLOGIA

Técnico en ingeniería en sistemas y Redes Informáticas

Ciclo II

Programación III

Licenciado. Willian Alexis Montes Girón

PARCIAL I

Presentado por:

Ramírez Medrano Daniel Osvaldo SMTR075223

Emmanuel Heriberto Martinez Aviles SMTS164422

27/08/24

En el ejercicio 1 ayudamos a niña mary a manejar su tienda de la manera más eficiente y fácil posible. (Obtamos por la funcionalidad POO)

Explicación del problema

1. **Productos:**
   * **Problema:** Niña Mary necesita llevar un registro de los productos, sus cantidades y precios.
   * **Solución:** Crear una clase Producto que almacene estos datos y permita actualizarlos según sea necesario.
2. **Registro de Ventas:**
   * **Problema:** Niña Mary realiza ventas, calcula el total y da el vuelto al cliente.
   * **Solución:** Implementar una clase Inventario que gestione la venta de productos, actualice las cantidades en el inventario, calcule el total de la venta y determine el vuelto que debe entregarse al cliente.
3. **Recepción de Proveedores:**
   * **Problema:** Niña Mary recibe productos de proveedores, quienes sugieren un precio de venta.
   * **Solución:** La clase Inventario también maneja la adición de nuevos productos al inventario con el precio sugerido.

En el ejercicio 2 se necesitaba realizar un sistema que permita ingresar asistencia y evaluar si el alumno tenia o no permiso. Esta resuelto utilizando clase de POO junto con funciones que permitirán la entrada y salida de datos a fin que le resulte mas fácil de comprender y utilizar para su usuario

1. Primeramente se creó la clase llamada Asistencia(), a la cual se le agregó los parámetros Nombre, Edad, Grado y Asistencia.
2. Para agregarle los datos a dichos parámetros se creó la función DatosAsistencia(), la cual se encargaría de darle su respectivo valor según se ingresen los datos pedidos
3. Para que el usuario ingresara los datos se hizo la función IngresarAsistencia() la cual se encargará de enviar los datos que reciba a la función DatosAsistencia(), para que dicha función realice su trabaja anteriormente dicho
4. Por último de creó la función Mostrar(), la cual se encargará de imprimir los datos ingresados al momento de instanciar un objeto de la clase Asistencia(): ósea, ingresar los datos y la asistencia del alumno
5. Finalmente se instanció el objeto profe1 de la clase Asistencia() en el cual se ingresaran los datos llamando la función IngresarAsistencia(profe1) para imprimirlos en pantalla con la función Mostrar(profe1)

En el ejercicio 3 (siempre utilizando POO) el objetivo es automatizar el proceso que sigue un recepcionista en un hotel, desde la presentación de opciones de habitaciones hasta la generación de una factura detallada

**Desglose del Problema:**

1. **Opciones de Habitaciones:**
   * **Problema:** El recepcionista presenta a los clientes las opciones de habitaciones disponibles junto con sus precios.
   * **Solución:** Crear una clase Habitación que represente cada tipo de habitación, incluyendo atributos como el tipo de habitación y el precio por noche.
2. **Datos del Cliente:**
   * **Problema:** El recepcionista recopila los datos personales del cliente y el número de noches que permanecerá en el hotel.
   * **Solución:** Crear una clase Cliente que almacene los datos del cliente, la habitación elegida, el número de noches, y los servicios extras solicitados.
3. **Servicios Extra:**
   * **Problema:** Los clientes pueden solicitar servicios adicionales que incrementan el costo total de la estancia.
   * **Solución:** Implementar una clase Servicio Extra que represente cada servicio adicional, como el uso de la piscina o la cancha de golf, con su respectivo precio. El cliente puede agregar estos servicios extras a su factura.
4. **Generación de Factura:**
   * **Problema:** El recepcionista genera una factura detallada que incluye el costo de la habitación y cualquier servicio adicional.
   * **Solución:** Crear una clase Factura que tome la información del cliente, calcule el total de la estancia incluyendo servicios adicionales, y genere un recibo detallado.

En el 4° ejercicio se realizó con listas que almacenaran los datos del trabajador según el tipo de plaza que tenga. De tal manera que le permita al usuario ingresar los datos según correspondan a su respectiva plaza y para que al final el programa muestre la boleta de pago del trabajador que el usuario quiera visualizar según su elección.

1. Primeramente se crearon dos listas según el tipo de plaza que tenga el trabajador: plazaFija[] y plazaXHora[].
2. Mediante el uso de un bucle While True se condicionó si el usuario quiere ingresar datos o ver la boleta de pago de los trabajadores ya registrados.
3. Se creó otra condicional en la cual , si el usuario eligió ingresar datos, este deba elegir en que lista de trabajadores desea ingresar los datos mediante la escritura de 1 o 2.
4. Se crearon las variables en las cuales el usuario podrá ingresar los datos del trabajador y mediante método append() estos datos puedan ser agregados a la lista escogida
5. Al terminar de ingresar los datos de un trabajador, se le pregunta si desea ingresar los datos de otro mediante una condicional, de ingresar “S” se repetirá el ingreso de datos, de lo contrario se le preguntará al usuario si desea visualizar la boleta de pago del trabajador que acaba de ingresar
6. Finalmente, se le pregunta al usuario si este desea visualizar la boleta de pago de todos los trabajadores ingresados, si escoge dicha elección, se imprimirá la boleta de pago de cada uno de los trabajadores, de lo contrario se finalizará el programa