“ AÑO DE LA UNIDAD, LA PAZ Y EL DESARROLLO “

UNIVERSIDAD NACIONAL AGRARIA DE LA SELVA

FACULTAD DE INGENIERÍA INFORMÁTICA Y SISTEMAS

“EXAMEN FINAL”

CURSO:

Programación básica

DOCENTE:

Carlos A. Ríos Rivera

ESTUDIANTE:

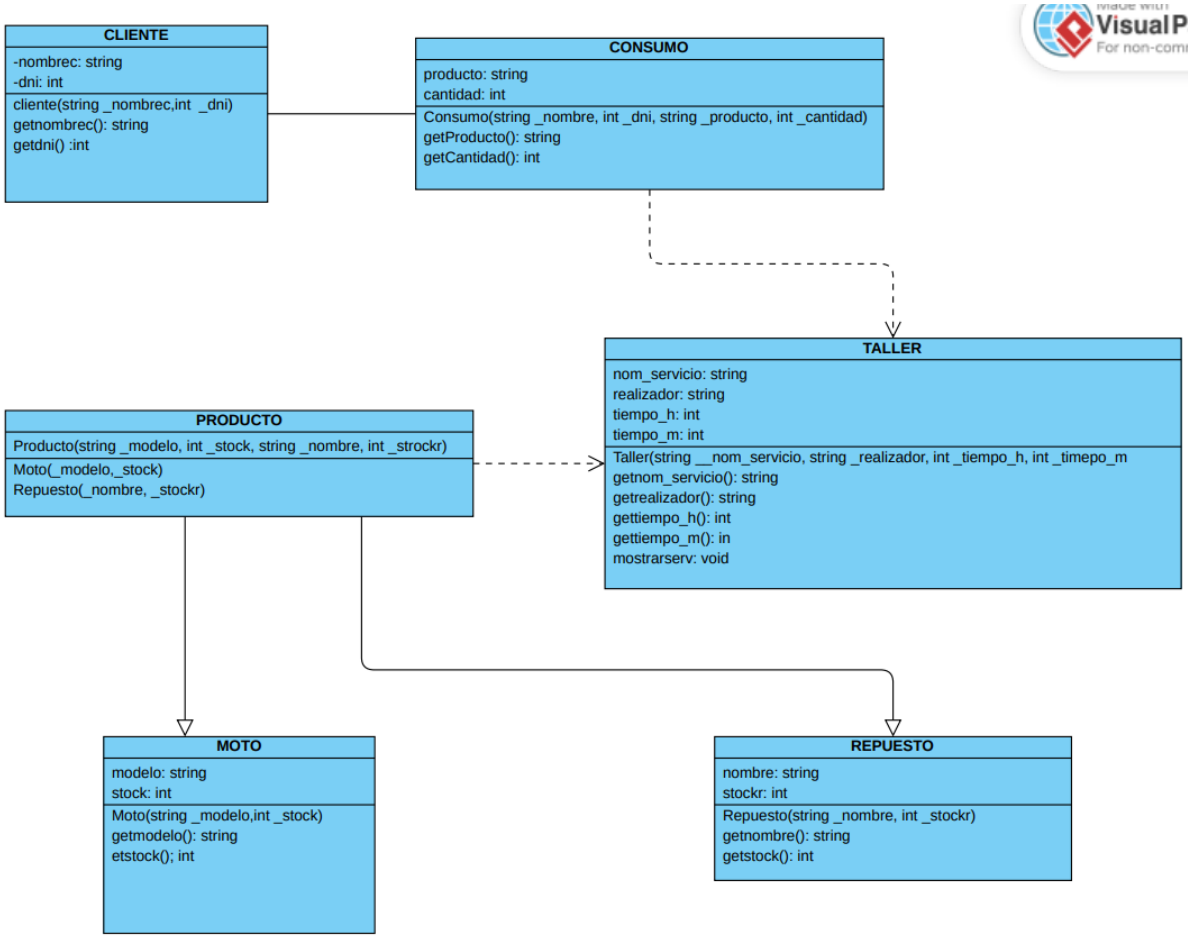
Rubina Savedra, Franco Daimbler

29 de mayo de 2023

INDICE

EXAMEN FINAL

1.1. Elabore el diagrama de clases



1.2. Implemente los métodos básicos para cada entidad

```cpp
#include<iostream>
using namespace std;
// Definición de la clase Moto
class Moto{
    private:
    string modelo;
    int stock;
    public:
    Moto(){} // Constructor por defecto de la clase Moto
    Moto(string, int); // Constructor de la clase Moto
    //void mostrar_moto();
    string getmodelo(); // Método para obtener el modelo de la moto
    int getstock(); // Método para obtener el stock de la moto};
// Definición de la clase Repuesto
class Repuesto{
    private:
    string nombre;
    int stockr;
    public:
    Repuesto(string,int); // Constructor de la clase Repuesto
    Repuesto(){} // Constructor por defecto de la clase Repuesto
    string getnombre(); // Método para obtener el nombre del repuesto
    int getstockr(); // Método para obtener el stock del repuesto
    //void mostrar_repuestos();};
// Definición de la clase Producto que hereda de las clases Moto y Repuesto
class Producto : public Moto, public Repuesto {
    public:
    Producto(string _modelo, int _stock, string _nombre, int _stockr): Moto(_modelo, _stock), Repuesto(_nombre, _stockr) {}};
// Definición de la clase Taller
class Taller{
    private:
    string nom_servicio;
    string realizador;
    int tiempo_h,tiempo_m;
    public:
    Taller(){} // Constructor por defecto de la clase Taller
    Taller(string,string,int,int); // Constructor de la clase Taller
    Taller(string,string); // Constructor de la clase Taller
    string getnom_servicio(); // Método para obtener el nombre del servicio del taller
    string getrealizador(); // Método para obtener el realizador del servicio del taller
    int gettiempo_h(); // Método para obtener las horas de tiempo del servicio del taller
    int gettiempo_m(); // Método para obtener los minutos de tiempo del servicio del taller
    void mostrarserv(); // Método para mostrar los detalles del servicio del taller};
// Definición de la clase Cliente
class Cliente {
private:
    string nombrec;
    int dni;
public:
    Cliente(){} // Constructor por defecto de la clase Cliente
    Cliente(string _nombrec,int _dni); // Constructor de la clase Cliente
    string getnombrec(); // Método para obtener el nombre del cliente
    int getdni(); // Método para obtener el DNI del cliente
};
// Definición de la clase Consumo que hereda de la clase Cliente
class Consumo : public Cliente {
private:
    string producto;
    int cantidad;
public:
    Consumo(string _nombre,int _dni, string _producto, int _cantidad); // Constructor de la clase Consumo
    Consumo(){} // Constructor por defecto de la clase Consumo
    string getProducto(); // Método para obtener el nombre del producto consumido
    int getCantidad(); // Método para obtener la cantidad del producto consumido};
// Implementación de los constructores de la clase Moto
Moto::Moto(string _modelo,int _stock){
    modelo=_modelo;
    stock=_stock;}
// Implementación de los constructores de la clase Repuesto
Repuesto::Repuesto(string _nombre,int _stockr){
    nombre=_nombre;
    stockr=stockr;}
// Implementación de los constructores de la clase Taller
Taller::Taller(string _nom_servicio,string _realizador,int _tiempo_h,int _tiempo_m){
    nom_servicio=_nom_servicio;
    realizador=_realizador;
    tiempo_h=_tiempo_h;
    tiempo_m=_tiempo_m;}
Taller::Taller(string _nom_servicio ,string _realizador){
    nom_servicio=_nom_servicio;
    realizador=_realizador;}
// Implementación del constructor de la clase Cliente
Cliente::Cliente(string _nombrec, int _dni) {
    nombrec = _nombrec;
    dni=_dni;}
// Implementación del constructor de la clase Consumo
Consumo::Consumo(string _nombre,int _dni, string _producto, int _cantidad) : Cliente(_nombre, _dni) {
    producto = _producto;
    cantidad = _cantidad;}
// Implementación de los métodos de la clase Taller
void Taller::mostrarserv(){
    cout<<"servicio: "<<nom_servicio<<endl;
    cout<<"realizador: "<<realizador<<endl;
    cout<<"..............................."<<endl;}
string Moto::getmodelo(){
    return modelo;}
int Moto::getstock(){
    return stock;}
string Repuesto::getnombre(){
    return nombre;}
int Repuesto::getstockr(){
    return stockr;}
string Taller::getnom_servicio(){
    return nom_servicio;}
string Taller::getrealizador(){
    return realizador;
}
int Taller::gettiempo_h(){
    return tiempo_h;
}
int Taller::gettiempo_m(){
    return tiempo_m;
}
string Cliente::getnombrec(){
    return nombrec;}
int Cliente::getdni(){
    return dni;
}
string Consumo::getProducto() {
    return producto;
}
int Consumo::getCantidad() {
    return cantidad;
}
```

1.3. Elabore los métodos necesarios para el problema

```
Clase Automovil:

_init_(self, modelo, color, precio): Método constructor que inicializa los
atributos de la clase Automovil con los valores proporcionados.

get_modelo(self): Método para obtener el modelo del automóvil.

set_precio(self, precio): Método para establecer el precio del automóvil.

mostrar(self): Método para mostrar los detalles del automóvil, como el
modelo, color y precio.

Clase Cliente:

_init_(self, nombre, telefono): Método constructor que inicializa los
atributos de la clase Cliente con los valores proporcionados.

get_nombre(self): Método para obtener el nombre del cliente.

set_telefono(self, telefono): Método para establecer el número de teléfono
del cliente.

mostrar(self): Método para mostrar los detalles del cliente, como el nombre y
teléfono.

Clase Taller:

_init_(self, nom_servicio, duracion, costo): Método constructor que
inicializa los atributos de la clase Taller con los valores proporcionados.

get_nom_servicio(self): Método para obtener el nombre del servicio del
taller.

set_costo(self, costo): Método para establecer el costo del servicio del
taller.

mostrar(self): Método para mostrar los detalles del taller, como el nombre
del servicio, duración y costo.
```

1.4. Implemente un reporte para ver el stock de productos por fecha de registro, cantidad y categoría

```
def generar_reporte_stock(productos):
    print("Reporte de Stock:")
    print("Fecha de Registro  |  Cantidad  |  Categoría")

    for producto in productos:
        fecha_registro = producto.fecha_registro.strftime("%Y-%m-%d")
        cantidad = producto.cantidad
        categoria = producto.categoria
        print(f"{fecha_registro}  |  {cantidad}  |  {categoria}")

# Ejemplo de uso
generar_reporte_stock(lista_productos)
```

1.5. . Implemente un reporte de los servicios realizados por el taller, quien lo realiza y que tiempo le toma, así mismo de los repuestos vendidos en el servicio

```
ofstream servici0;
                ofstream servi;
                string servicio, realizador;
                int tiempoh,tiempom,n;

                Taller s[100];//arreglo de objetos

                cout<<"ingrese la cantidad de servicios que aniadira: ";
                cin>>n;
                servici0.open("servicios.txt",ios::app);
                servi.open("serviciosm.txt",ios::app);
                for(int c=0;c<n;c++){
                    cout << "Ingrese el nombre del servicio: ";
                    cin>>servicio;
                    cout << "Ingrese el nombre del realizador: ";
                    cin >> realizador;
                    cout << "Ingrese el tiempo requerido en horas: ";
                    cin >> tiempoh;
                    cout << "Ingrese el tiempo requerido en minutos: ";
                    cin >> tiempom;
                    s[c]=Taller(servicio,realizador,tiempoh,tiempom);

                    servicio=s[c].getnom_servicio();
                    realizador=s[c].getrealizador();
                    tiempoh=s[c].gettiempo_h();
                    tiempom=s[c].gettiempo_m();
                    s[c]=Taller(servicio, realizador, tiempoh,tiempom);
                    //guardando los datos en el archivo
                    servici0<<"servicio: "<<servicio<<endl
                         <<"realizador: "<<realizador<<endl
                         <<"tiempo h.: "<<tiempoh<<endl
                         <<"tiempo m.: "<<tiempom<<endl
                         <<"............................."<<endl;

                    servi<<servicio<<endl
                         <<realizador<<endl;
                }
                servici0.close();
                servi.close();
                cout << "Servicio agregado correctamente.\n";
                cout << "\n\npresione cualquier tecla para volver al menu.";
                getch();
                system("cls");
```

1.6. Implemente un reporte de los clientes comunes y el consumo por fecha

```
string nombreCliente,clientec, producto;
int cantidad,dn,dia, mes, anio, horas, minutos,n;
ifstream reportec;
reportec.open("consumo.txt",ios::in);

Consumo csm[100];//arreglo de objetos
cout<<"digite la cantidad de clientes que evaluara: ";
cin>>n;
ofstream reporte;
reporte.open("consumo.txt", ios::app);
if(reportec.fail()){
        cout<<"no se pudo abrir el archivo";
    }
for(int l=0;l<n;l++){
    cout << "Ingrese el nombre del cliente: ";
    cin >> nombreCliente;
    cout<<"ingrese el DNI: ";
    cin>>dn;
    //bool cliente_comun=false;

    /*if(cliente_comun==1){
        cout<<"el cliente es habitual"<<endl;
    }
    else{
        cout<<"no es cliente habitual"<<endl;
    }*/
    cout << "Ingrese el producto: ";
    cin >> producto;
    cout << "Ingrese la cantidad: ";
    cin >> cantidad;
    csm[l]=Consumo(nombreCliente,dn, producto, cantidad);
        time_t t;
        t = time(NULL);
        struct tm* fecha; // Corrección: se debe declarar como puntero a struct
tm
        fecha = localtime(&t);
        dia = fecha->tm_mday;
        mes = fecha->tm_mon + 1;
        anio = fecha->tm_year + 1900;
        horas = fecha->tm_hour;
        minutos = fecha->tm_min;
    while(reportec>>dia>>horas>>minutos>>clientec>>dn>>producto>>cantidad){//lee
el arechivo y guarda las variables de este
```

```
        int r;
        if(dn==dn){
            r=11;
            cout<<"cacacacca";
        }
        else{
            cout<<"dodmoidioididw";
            r=10;
        }
        cout<<r<<"<--"<<endl;
    }
    reportec.close();
    clientec=csm[l].getnombrec();
    dn=csm[l].getdni();
    producto=csm[l].getProducto();
    cantidad=csm[l].getCantidad();
    reporte <<"fecha: "<<dia<<"/"<<mes<<"/"<<anio<<endl
            <<"hora: "<<horas<<":"<<minutos<<endl
            <<"Cliente: " << clientec<< endl
            <<"D.N.I: "<<dn<<endl
            << "Producto: " << producto << endl
            << "Cantidad: " << cantidad << endl
            << "..............................." << endl;
    reporte.close();//cierra el archivo
    cout << "\n\npresione cualquier tecla para volver al menu.";
    getch();
    system("cls");
```

1.7. Elabore un menú dinámico

```
// Función del menú que muestra las opciones y solicita una opción al usuario
int menu(){
    int opc;
    cout << "::::::::::::: MENU :::::::::::::\n";
    cout << "1. Agregar moto\n";
    cout << "2. Agregar repuesto\n";
    cout << "3. ver stock de productos\n";
    cout << "4. Agregar servicio del taller\n";
    cout << "5. usar servicio"<<endl;
    cout << "6. Agregar consumo de cliente\n";
    cout << "7. Salir\n";
    cout << "Ingrese una opciOn: ";
    cin>>opc;
    return opc;
```

INFORME:

2.1. Introducción

El presente informe describe el desarrollo de un sistema de gestión de inventario y servicios para un taller de motocicletas, implementado utilizando el lenguaje de programación C++. El objetivo principal del proyecto es proporcionar una herramienta eficiente y automatizada que permita al taller llevar un control preciso de su inventario, así como gestionar los servicios realizados a las motocicletas de sus clientes. El sistema se ha diseñado para agilizar los procesos internos del taller, mejorar la eficiencia operativa y ofrecer un servicio de calidad a los clientes.

2.2. Objetivos del proyecto

- Desarrollar un sistema de gestión de inventario en C++ que permita llevar un registro actualizado de las piezas, componentes y suministros disponibles en el taller.

- Implementar un módulo de gestión de servicios en C++ que permita registrar y dar seguimiento a los servicios realizados a las motocicletas de los clientes.

- Facilitar la generación de reportes y análisis en C++ sobre el inventario, los servicios realizados y otros indicadores relevantes para la gestión del taller.

2.3. . Metodología

El desarrollo del sistema se ha realizado siguiendo una metodología ágil, basada en iteraciones cortas y la colaboración cercana con los usuarios del taller. Se ha utilizado un enfoque incremental, lo que ha permitido ir incorporando funcionalidades adicionales a medida que se completaban las etapas iniciales del proyecto.

2.4. . Funcionalidades del sistema

El sistema de gestión de inventario y servicios para el taller de motocicletas implementado en C++ incluye las siguientes funcionalidades principales:

- Registro de productos: Permite añadir nuevos productos al inventario, indicando su nombre, descripción, número de pieza, proveedor, precio y cantidad disponible.

- Gestión de stock: Permite llevar un control actualizado de las existencias de cada producto, generando alertas cuando se alcanzan niveles mínimos.

- Registro de servicios: Permite registrar los servicios realizados a las motocicletas de los clientes, incluyendo detalles como el número de chasis, la fecha, el tipo de servicio y el costo asociado.

- Seguimiento de servicios: Permite dar seguimiento a los servicios registrados, actualizando su estado (pendiente, en proceso, completado) y generando recordatorios para el personal encargado.

- Generación de reportes: Permite generar informes en C++ sobre el inventario, los servicios realizados, los clientes y otros indicadores relevantes para la gestión del taller.

2.5. Tecnologías utilizadas

El sistema se ha desarrollado utilizando el lenguaje de programación C++ en conjunto con librerías y herramientas estándar. Se ha utilizado una arquitectura cliente-servidor, donde el servidor se ha implementado en C++ para gestionar las solicitudes y respuestas del sistema.

2.6. 6. Resultados y conclusiones

El sistema de gestión de inventario y servicios desarrollado en C++ para el taller de motocicletas ha demostrado ser una herramienta eficiente y efectiva para mejorar la gestión interna y la atención al cliente. A través de la automatización de procesos, se ha logrado reducir los errores y agilizar las tareas administrativas. Además, la generación de reportes