Examen 1 – Erick Roberto Nuñez Oreamuno

1. Análisis de Requerimientos

   - Identifica las clases necesarias para desarrollar el sistema propuesto. Justifica tu elección.

   R/ **Clases**

   - Clientes ➔ Lleva el registro completo de la información de los clientes del taller.
   - Vehículos ➔ Registra la informacion del vehículo y a cuál cliente está asociado.
   - Servicios ➔ Brinda un desglose de los servicios disponibles para los vehículos, especificando en qué consiste cada servicio y el costo.
   - Citas ➔ Asocia al cliente y su vehículo con un mecánico disponible para los servicios seleccionados, y lleva un control de la fecha y hora de las citas.
   - Mecánicos ➔ Mantiene una base de datos con la información de los mecánicos, incluyendo su especialidad y disponibilidad, para una asignación de citas efectiva.

   Las clases se basan en las necesidades específicas del dueño del taller para automatizar y mejorar la gestión de las operaciones del taller.

   - Define los atributos necesarios para cada clase identificada. Explica por qué cada atributo es necesario.

   R/ **Cliente**

   - '**nombre**' String ➔ Identificar al cliente.
   - '**direccion**' String ➔ Facturación y contacto.
   - '**telefono**' String ➔ Comunicación con el cliente.
   - '**correoElectronico**' String ➔ Confirmaciones de citas.

   Vehículo

   - '**marca**' String ➔ Marca del vehículo.
   - '**modelo**' String. ➔ Modelo del vehículo.
   - '**anio**' Int ➔ Año de fabricación del vehículo.
   - '**vin**' String ➔ Número de identificación del vehículo.
   - '**propietario**' Cliente ➔ Relaciona el vehículo con su propietario usando la clase Cliente

Servicio

- '**nombre**' String ➔ Nombre del servicio ofrecido.
- '**descripcion**' String ➔ Descripción del servicio.
- '**costo**' Double ➔ Costo del servicio.

Cita

- '**cliente**' Cliente ➔ Identifica al cliente que solicita la cita usando la clase Cliente.
- '**vehiculo**' Vehículo ➔ Indica el vehículo que se atenderá usando la clase Vehículo.
- '**servicio**' Servicio ➔ Especifica el servicio solicitado usando la clase Servicio.
- '**fechaHora**' DateTime ➔ Fecha y hora de la cita.
- '**mecanico**' Mecánico ➔ Asigna el mecánico que atenderá la cita usando la clase Mecánico

Mecánico

- '**nombre**' String ➔ Identifica al mecánico.
- '**especialidad**' String ➔ Describe la especialidad del mecánico.
- '**disponibilidad**' Boolean ➔ Indica si el mecánico está disponible para citas.

- Describe los métodos que debe tener cada clase para cumplir con los requisitos del sistema.

R/ **Cliente**

- RegistrarCliente() ➔ Registra un nuevo cliente.
- ActualizarInformacion() ➔ Actualiza la información del cliente.
- ObtenerDetalles() ➔ Devuelve los detalles del cliente.

Vehículo

- RegistrarVehiculo() ➔ Registra un nuevo vehículo.
- ActualizarInformacion() ➔ Actualiza la información del vehículo.
- ObtenerDetalles() ➔ Devuelve los detalles del vehículo.

Servicio

- RegistrarServicio() ➔ Registra un nuevo servicio.
- ActualizarServicio() ➔ Actualiza la información del servicio.

- ObtenerDetalles() ➔ Devuelve los detalles del servicio.

Cita

- RegistrarCita() ➔ Registra una nueva cita.
- CancelarCita() ➔ Cancela una cita existente.
- ObtenerDetalles() ➔ Devuelve los detalles de la cita.
- ModificarCita() ➔ Modifica la información de una cita existente.

Mecánico

- RegistrarMecanico() ➔ Registra un nuevo mecánico.
- ActualizarInformacion() ➔ Actualiza la información del mecánico.
- ObtenerDisponibilidad() ➔ Devuelve la disponibilidad del mecánico.
- AsignarTarea() ➔ Asigna una tarea específica al mecánico.

2. Diseño de la Pantalla

- Elige una de las necesidades especificadas (gestión de clientes, gestión de vehículos, gestión de servicios, gestión de citas, gestión de mecánicos). Y justificar porque

R/ Elegí la necesidad "**gestión de citas**" ya que es fundamental para coordinar entre los clientes, vehículos, servicios y mecánicos. Esto ayudara a que el programa utilice en su totalidad la información de clientes, vehículos, servicios y mecánicos, lo que centraliza la coordinación y seguimiento de dichas citas. Ademas le dará una experiencia más optima y profesional a los clientes al permitir una fácil programación y gestión de sus citas.

- Diseña una consola que resuelva dicha necesidad describe brevemente la funcionalidad de cada uno.

R/ A continuación, se presenta un diseño conceptual de una pantalla de consola para la gestión de citas en un taller mecánico:

Taller Mecánico AutoSoluciones

Gestión de citas

1 - Registrar nueva cita
2 - Modificar cita
3 - Cancelar cita
4 - Ver cita programada
5 - Salir

Seleccione una opción:

1. **Registrar Nueva Cita:**

    a) Se le pide al usuario que ingrese el nombre del cliente, la dirección, el teléfono y el correo electrónico.

    b) Luego se solicita la marca, el modelo, el año y el VIN del vehículo.

    c) Despues, el nombre de un servicio, la descripción y el costo.

    d) Se le pedirá al usuario que ingrese la fecha y hora deseada para la cita.

    e) Finalmente, el sistema asignará automáticamente un mecánico disponible para la cita.

2. **Modificar Cita:**

    a) Se le pide al usuario la fecha y hora de la cita que desea modificar.

    b) Una vez identificada la cita, el usuario podrá modificar la información del cliente, del vehículo o cambiar la fecha y hora de la cita.

3. **Cancelar Cita:**

    a) El usuario ingresa la fecha y hora de la cita que desea cancelar.

    b) El sistema buscará y eliminará la cita, confirmando la cancelación al usuario.

4. **Ver Cita Programada:**

    a) El sistema mostrará una lista de todas las citas programadas, permitiendo al usuario ver los detalles de cada cita, incluyendo cliente, vehículo, servicio, fecha y hora, y mecánico asignado.

5. **Salir:**

    a) El usuario selecciona esta opción para salir del sistema. El programa de consola se cierra.

3. Desarrollo del Programa

    - Implementa en código una de las funcionalidades del sistema basado en la pantalla que has diseñado.

- El código debe incluir la creación de clases con sus respectivos atributos y métodos.

Codigo:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Examen_1_Erick_Nunez
{
    // Clase Cita para gestionar las citas
    public class Cita
    {
        public Cliente cliente;
        public Vehiculo vehiculo;
        public Servicio servicio;
        public DateTime fechaHora;
        public Mecanico mecanico;

        public void RegistrarCita()
        {
            cliente = new Cliente();
            vehiculo = new Vehiculo();
            servicio = new Servicio();
            mecanico = new Mecanico();

            Console.WriteLine("\nRegistrar Nueva Cita\n");

            // Informacion del cliente
            Console.Write("Nombre del cliente: ");
            cliente.nombre = Console.ReadLine();

            Console.Write("Direccion del cliente: ");
            cliente.direccion = Console.ReadLine();

            Console.Write("Telefono del cliente: ");
            cliente.telefono = Console.ReadLine();

            Console.Write("Correo electronico del cliente: ");
            cliente.correoElectronico = Console.ReadLine();

            // Informacion del vehiculo
            Console.Write("Marca del vehiculo: ");
            vehiculo.marca = Console.ReadLine();
```

```
            Console.Write("Modelo del vehiculo: ");
            vehiculo.modelo = Console.ReadLine();

            Console.Write("Año del vehiculo: ");
            vehiculo.anio = Convert.ToInt32(Console.ReadLine());

            Console.Write("VIN del vehiculo: ");
            vehiculo.vin = Console.ReadLine();
            vehiculo.propietario = cliente;

            // Informacion del servicio
            Console.Write("Nombre del servicio: ");
            servicio.nombre = Console.ReadLine();

            Console.Write("Descripcion del servicio: ");
            servicio.descripcion = Console.ReadLine();

            Console.Write("Costo del servicio: ");
            servicio.costo = Convert.ToDouble(Console.ReadLine());

            // Informacion del mecanico
            Console.Write("Nombre del mecanico: ");
            mecanico.nombre = Console.ReadLine();
            mecanico.disponibilidad = true;

            Console.Write("Especialidad: ");
            mecanico.especialidad = Console.ReadLine();

            // Fecha y hora de la cita
            Console.Write("Fecha y hora de la cita (dd/mm/yyyy hh:mm): ");
            fechaHora = DateTime.Parse(Console.ReadLine());

            Console.WriteLine("\nLa cita ha sido registrada con exito.");
        }

        public void ModificarCita()
        {
            if (cliente == null || vehiculo == null || servicio == null || mecanico ==
null)
            {
                Console.WriteLine("\nNo hay citas para modificar.");
                return;
            }

            Console.WriteLine("\nModificar Cita\n");
```

```
            // Modificar el nombre del cliente
            Console.Write("Nuevo nombre del cliente (deje en blanco para no
cambiar): ");
            string nuevoNombreCliente = Console.ReadLine();
            if (!string.IsNullOrEmpty(nuevoNombreCliente))
            {
                cliente.nombre = nuevoNombreCliente;
            }

            // Modificar la fecha y hora de la cita
            Console.Write("Nueva fecha y hora de la cita (dd/mm/yyyy hh:mm, deje
en blanco para no cambiar): ");
            string nuevaFechaHora = Console.ReadLine();
            if (!string.IsNullOrEmpty(nuevaFechaHora))
            {
                fechaHora = DateTime.Parse(nuevaFechaHora);
            }

            Console.WriteLine("\nCita modificada exitosamente.");
        }

        public void CancelarCita()
        {
            if (cliente == null || vehiculo == null || servicio == null || mecanico ==
null)
            {
                Console.WriteLine("\nNo hay citas para cancelar.");
                return;
            }

            Console.WriteLine("\nCancelar Cita\n");

            Console.Write("Ingrese la fecha y hora de la cita a cancelar
(dd/mm/yyyy hh:mm): ");
            string fechaHoraStr = Console.ReadLine();

            // Verificar si el campo de entrada está vacío
            if (string.IsNullOrEmpty(fechaHoraStr))
            {
                Console.WriteLine("\nNo se puede dejar el campo de fecha y hora en
blanco. Inténtelo de nuevo.");
                return;
            }

            // Intentar parsear la fecha y hora proporcionada
            DateTime fechaHoraCancelar;
            if (!DateTime.TryParse(fechaHoraStr, out fechaHoraCancelar))
```

```
            {
                Console.WriteLine("\nFormato de fecha y hora no válido. Use el
formato dd/mm/yyyy hh:mm.");
                return;
            }

            // Comparar la fecha y hora ingresada con la fecha y hora de la cita
actual
            if (fechaHora == fechaHoraCancelar)
            {
                cliente = null;
                vehiculo = null;
                servicio = null;
                mecanico = null;
                fechaHora = default(DateTime);
                Console.WriteLine("\nCita cancelada exitosamente.");
            }
            else
            {
                Console.WriteLine("\nCita no encontrada.");
            }
        }

        public void ObtenerDetalles()
        {
            if (cliente == null || vehiculo == null || servicio == null || mecanico ==
null)
            {
                Console.WriteLine("\nNo hay citas programadas.");
                return;
            }

            Console.WriteLine("\nDetalles de la Cita:\n");
            Console.WriteLine($"Cliente: {cliente.nombre}");
            Console.WriteLine($"Vehiculo: {vehiculo.marca} {vehiculo.modelo},
Año: {vehiculo.anio}, VIN: {vehiculo.vin}");
            Console.WriteLine($"Servicio: {servicio.nombre}, Descripcion:
{servicio.descripcion}, Costo: {servicio.costo}");
            Console.WriteLine($"Fecha y Hora: {fechaHora}");
            Console.WriteLine($"Mecanico: {mecanico.nombre}, Especialidad:
{mecanico.especialidad}\n");
        }
    }

    // Clase Cliente para gestionar los clientes
    public class Cliente
    {
```

```
        public string nombre;
        public string direccion;
        public string telefono;
        public string correoElectronico;

        public void RegistrarCliente()
        {
            Console.WriteLine("\nRegistrar cliente\n");
            Console.Write("Nombre: ");
            nombre = Console.ReadLine();
            Console.Write("Direccion: ");
            direccion = Console.ReadLine();
            Console.Write("Telefono: ");
            telefono = Console.ReadLine();
            Console.Write("Correo electronico: ");
            correoElectronico = Console.ReadLine();
        }

        public void ActualizarInformacion()
        {
            Console.WriteLine("\nActualizar informacion del cliente\n");
            Console.Write("Nuevo nombre (deje en blanco para no cambiar): ");
            string nuevoNombre = Console.ReadLine();
            if (!string.IsNullOrEmpty(nuevoNombre))
            {
                nombre = nuevoNombre;
            }
            // Similarmente para los otros campos
        }

        public void ObtenerDetalles()
        {
            Console.WriteLine($"Cliente: {nombre}, Direccion: {direccion}, Telefono:
{telefono}, Correo electronico: {correoElectronico}");
        }
    }

    // Clase Vehiculo para gestionar los vehiculos
    public class Vehiculo
    {
        public string marca;
        public string modelo;
        public int anio;
        public string vin;
        public Cliente propietario;

        public void RegistrarVehiculo()
```

```
        {
            Console.WriteLine("\nRegistrar vehiculo\n");
            Console.Write("Marca: ");
            marca = Console.ReadLine();
            Console.Write("Modelo: ");
            modelo = Console.ReadLine();
            Console.Write("Año: ");
            anio = Convert.ToInt32(Console.ReadLine());
            Console.Write("VIN: ");
            vin = Console.ReadLine();
        }

        public void ActualizarInformacion()
        {
            Console.WriteLine("\nActualizar informacion del vehiculo\n");
            // Similar a Cliente
        }

        public void ObtenerDetalles()
        {
            Console.WriteLine($"Vehiculo: {marca} {modelo}, Año: {anio}, VIN:
{vin}");
        }
    }

    // Clase Servicio para gestionar los servicios
    public class Servicio
    {
        public string nombre;
        public string descripcion;
        public double costo;

        public void RegistrarServicio()
        {
            Console.WriteLine("\nRegistrar servicio\n");
            Console.Write("Nombre: ");
            nombre = Console.ReadLine();
            Console.Write("Descripcion: ");
            descripcion = Console.ReadLine();
            Console.Write("Costo: ");
            costo = Convert.ToDouble(Console.ReadLine());
        }

        public void ActualizarServicio()
        {
            Console.WriteLine("\nActualizar servicio\n");
            // Similar a Cliente
```

```
        }

        public void ObtenerDetalles()
        {
            Console.WriteLine($"Servicio: {nombre}, Descripcion: {descripcion},
Costo: {costo}");
        }
    }

    // Clase Mecanico para gestionar los mecanicos
    public class Mecanico
    {
        public string nombre;
        public string especialidad;
        public bool disponibilidad;

        public void RegistrarMecanico()
        {
            Console.WriteLine("\nRegistrar mecanico\n");
            Console.Write("Nombre: ");
            nombre = Console.ReadLine();
            Console.Write("Especialidad: ");
            especialidad = Console.ReadLine();
            disponibilidad = true; // Asumimos que el mecanico esta disponible
inicialmente
        }

        public void ActualizarInformacion()
        {
            Console.WriteLine("\nActualizar informacion del mecanico\n");
            // Similar a Cliente
        }

        public void ObtenerDisponibilidad()
        {
            Console.WriteLine($"Mecanico: {nombre}, Especialidad: {especialidad},
Disponibilidad: {(disponibilidad ? "Si" : "No")}");
        }

        public void AsignarTarea()
        {
            disponibilidad = false; // Al asignar una tarea, el mecanico ya no esta
disponible
        }
    }

    // Clase Program para la consola
```

```
    public class Program
    {
        public static Cita cita = new Cita();

        public static void Main(string[] args)
        {
            string opcion;

            do
            {
                Console.WriteLine("\nTaller Mecanico AutoSoluciones");
                Console.WriteLine("\nGestion de citas\n");
                Console.WriteLine("1 - Registrar nueva cita");
                Console.WriteLine("2 - Modificar cita");
                Console.WriteLine("3 - Cancelar cita");
                Console.WriteLine("4 - Ver cita programada");
                Console.WriteLine("5 - Salir");
                Console.Write("\nSeleccione una opcion: ");
                opcion = Console.ReadLine();

                switch (opcion)
                {
                    case "1":
                        cita.RegistrarCita();
                        break;
                    case "2":
                        cita.ModificarCita();
                        break;
                    case "3":
                        cita.CancelarCita();
                        break;
                    case "4":
                        cita.ObtenerDetalles();
                        break;
                    case "5":
                        Console.WriteLine("\nCerrando...\n");
                        break;
                    default:
                        Console.WriteLine("\nOpcion no valida.");
                        break;
                }

            } while (opcion != "5");
        }
    }
}
```

Entrega

Sube tu análisis, diseño y código desarrollado en un documento (puede ser un archivo PDF o un archivo de Word) a la plataforma de entrega GitHub y copiar enlace en Moodle antes de la fecha límite.