Proyecto 1. Valor 1%

Temas de Estudio

1. Tema 1 Introducción al lenguaje de programación
2. Tema 2 Particularidades del lenguaje C#
3. Tema 3 Manejo de excepciones en C#
4. Tema 4 Conceptos de Interfaz Gráfica

Objetivo

Aplicar los conceptos generales del lenguaje de programación C#, con el objetivo de profundizar en la semántica y sintaxis del lenguaje. Además, poner en práctica los conceptos de interfaz gráfica, así como el manejo de excepciones.

Software de Desarrollo

Visual Studio Community 2022 (C#, .Net Framework 4.8 o Net 6.0)

Desarrollo

La empresa **DESARROLLO-UNED** fue contratada para el desarrollo de una herramienta de escritorio que permita la administración de citas a los dentistas de San José, lo cual permita a los usuarios del sistema contar con el manejo de los clientes, doctores y el control de las citas asignadas a los clientes con los doctores asignados. Es por esa razón que le han contratado para que diseñe y programe un sistema en C# que cuente con las siguientes funcionalidades:

El sistema debe contar con un **menú principal** con las siguientes opciones:

- **Registrar Tipos de Consulta**
- **Administración de Clientes**
- **Administración de Doctores**
- **Registro de Citas**
- **Reporte Citas por Fecha**
- **Reporte Citas por Doctor**
- **Reporte Citas por Cliente**

Registrar Tipos de Consulta

Está opción del menú le permite al usuario realizar el registro, consulta y modificación de los tipos de consultas que realizarán en el consultorio, por ejemplo: extracción de piezas dentales, limpieza, cirugías, entre otros servicios brindados.

Esta pantalla deberá con la funcionalidad de registro de la siguiente información:

- Número (int), este campo corresponde a almacenar una identificación numérica única por tipo de consulta.
- Descripción(string), este campo corresponde almacenar la descripción del tipo de consulta y debe ser requerido.
- Estado (char), debe mostrar la información en un Combobox con el detalle del control de Activo o Inactivo, pero debe almacenar a lo interno una “A” para Activo y una “I” para Inactivo, este campo también es requerido.

Para ello se de crear una Clase con los atributos mencionados anteriormente. Así mismo, el sistema deberá mostrar un mensaje en las siguientes validaciones:

- Cuando se trate de ingresar un número de tipo de consulta que ya se encuentra registrada en el sistema.
- Se presente un error en el formato del campo numérico.
- Cuando los campos número y descripción se encuentren vacíos (nulo).
- Cuando el Combobox no tenga seleccionada una opción.
- Cualquier otro error que se presente en el sistema, debe estar manejado por una excepción.

Una vez capturada la información se debe agregar a un arreglo de objetos de tipo Consulta. El arreglo debe ser de 10 posiciones.

La pantalla debe tener la funcionalidad de consultar en un DataGridView la información de los tipos de Consulta registrados en el arreglo, cada vez que ingrese un dato nuevo debe actualizarse la consulta en el control del DataGridView.

Además, la pantalla debe tener la funcionalidad de seleccionar un tipo de consulta y modificar el estado del tipo de consulta (activo o inactivo). **No debe modificar el número de identificación ni la descripción.**

Se deberá crear como **mínimo** una nueva ventana (Windows Form) para resolver los requerimientos solicitados anteriormente.

Administración de Clientes

Está opción del menú le permite al usuario realizar el registro, consulta y modificación de clientes que serán atendidos en las citas programadas, es importante registrar sus datos personales.

Esta pantalla deberá con la funcionalidad de registro de la siguiente información:

- Identificación (int), este campo corresponde a almacenar una identificación numérica y única del cliente.
- Nombre(string), este campo corresponde almacenar el nombre del cliente y debe ser requerido.
- Apellido1 (string), este campo corresponde almacenar el primer apellido del cliente y debe ser requerido.
- Apellido2 (string), este campo corresponde almacenar el segundo apellido del cliente y debe ser requerido.
- Fecha de nacimiento (DateTime), este campo corresponde la fecha de nacimiento del cliente y debe ser requerido.
- Género (char), debe mostrar la información en un Combobox con el detalle del control de Femenino, Masculino o No Especificado, pero debe almacenar a lo interno una “F” para Femenino, una “M” para Masculino o una “N” para No especificado, este campo también es requerido.

Para ello se de crear una Clase con los atributos mencionados anteriormente. Así mismo, el sistema deberá mostrar un mensaje en las siguientes validaciones:

- Cuando se trate de ingresar una identificación que ya se encuentra registrada en el sistema.
- Se presente un error en el formato del campo, ya sea numérico o de fecha.
- Cuando los campos identificación, nombre, apellido1, apellido2 y fecha de nacimiento se encuentren vacíos (nulo).
- Cuando el Combobox no tenga seleccionada una opción.
- Cualquier otro error que se presente en el sistema, debe estar manejado por una excepción.

Una vez capturada la información se debe agregar a un arreglo de objetos de Cliente. El arreglo debe ser de 20 posiciones.

La pantalla debe tener la funcionalidad de consultar en un DataGridView la información de los clientes registrados en el arreglo, cada vez que ingrese un dato nuevo debe actualizarse la consulta en el control del DataGridView.

Además, la pantalla debe tener la funcionalidad de seleccionar un cliente y modificar la fecha de nacimiento y el género. **No debe modificar el número de identificación, nombre, apellido1 y apellido2.**

Se deberá crear como **mínimo** una nueva ventana (Windows Form) para resolver los requerimientos solicitados anteriormente.

Administración de Doctores

Está opción del menú le permite al usuario realizar el registro, consulta y modificación de doctores que serán los que brindaran las consultas programadas, es importante registrar los datos relevantes del doctor.

Esta pantalla deberá con la funcionalidad de registro de la siguiente información:

- Identificación (int), este campo corresponde a almacenar una identificación numérica y única del doctor.
- Nombre(string), este campo corresponde almacenar el nombre del doctor y debe ser requerido.
- Apellido1 (string), este campo corresponde almacenar el primer apellido del doctor y debe ser requerido.
- Apellido2 (string), este campo corresponde almacenar el segundo apellido del doctor y debe ser requerido.
- Estado (char), debe mostrar la información en un Combobox con el detalle del control de Activo o Inactivo, pero debe almacenar a lo interno una “A” para Activo y una “I” para Inactivo, este campo también es requerido.

Para ello se de crear una Clase con los atributos mencionados anteriormente. Así mismo, el sistema deberá mostrar un mensaje en las siguientes validaciones:

- Cuando se trate de ingresar una identificación que ya se encuentra registrada en el sistema.
- Se presente un error en el formato del campo, ya sea numérico o de fecha.
- Cuando los campos identificación, nombre, apellido1 y apellido2 se encuentren vacíos (nulo).
- Cuando el Combobox no tenga seleccionada una opción.
- Cualquier otro error que se presente en el sistema, debe estar manejado por una excepción.

Una vez capturada la información se debe agregar a un arreglo de objetos de Doctor. El arreglo debe ser de 20 posiciones.

La pantalla debe tener la funcionalidad de consultar en un DataGridView la información de los doctores registrados en el arreglo, cada vez que ingrese un dato nuevo debe actualizarse la consulta en el control del DataGridView.

Se deberá crear como **mínimo** una nueva ventana (Windows Form) para resolver los requerimientos solicitados anteriormente.

Registro de Citas

Funcionalidad que debe permitir al usuario registrar las citas que requiere el paciente (cliente) con el dentista disponible. El sistema deberá tener la funcionalidad de asignar un número de cita, poder seleccionar el cliente que ya se encuentre registrado, así como el doctor designado. El sistema debe validar que no encuentre una cita a la fecha y hora seleccionada para el doctor seleccionado. Para esto se debe realizar lo siguiente:

- Imprimir en pantalla la lista de doctores disponibles.
    - Se debe validar que al menos exista un doctor activo registrado en el sistema para realizar la asignación de la cita.
    - Solo debe mostrar los doctores activos.
- Imprimir en pantalla la lista de los clientes registrados.
    - Se debe validar que al menos exista un cliente registrado para ser asociado
    - Para cada registrado de la lista se debe mostrar la siguiente información: id placa, marca, modelo
- Número(int), este campo corresponde a almacenar el número de cita y único.
- Fecha y Hora de la cita (DateTime), este campo corresponde a almacenar la fecha y hora de la cita y debe ser requerido.
- Tipo de Consulta, debe tener la funcionalidad de seleccionar el tipo de consulta, de acuerdo con lo almacenado en el arreglo de Tipos de Consulta.
- Cliente (Objeto Cliente), debe tener la funcionalidad de seleccionar el cliente, de acuerdo con lo almacenado en el arreglo de Clientes.
- Doctor (Objeto Doctor), debe tener la funcionalidad de seleccionar el doctor, de acuerdo con lo almacenado en el arreglo de Doctores.

Para ello se de crear una Clase con los atributos mencionados anteriormente. Así mismo, el sistema deberá mostrar un mensaje en las siguientes validaciones:

- Cuando se trate de ingresar un número que ya se encuentra registrada en el sistema.
- Se presente un error en el formato del campo, ya sea numérico o de fecha.

- Cuando los campos identificación, nombre, apellido1 y apellido2 se encuentren vacíos (nulo).
- Cualquier otro error que se presente en el sistema, debe estar manejado por una excepción.

Una vez capturada la información se debe agregar a un arreglo de objetos de Citas. El arreglo debe ser de 20 posiciones.

Reporte Citas por Fecha

Esta opción debe permita al usuario visualizar las citas por una fecha en específico, debe contar con la funcionalidad de que el usuario seleccione una fecha y el sistema muestre solamente las citas que existen para el día seleccionado. La información que debe mostrar en el control DataGridView es el siguiente:

- Número de cita.
- Fecha y Hora de la cita.
- Descripción del tipo de consulta.
- Nombre completo del cliente.
- Nombre completo del doctor.

Se deberá crear como **mínimo** una nueva ventana (Windows Form) para resolver los requerimientos solicitados anteriormente.

Reporte Citas por Doctor

Esta opción debe permita al usuario visualizar las citas por un doctor en específico, debe contar con la funcionalidad de que el usuario seleccione un doctor que ya se encuentre registrado en el arreglo de doctores y el sistema muestre solamente las citas que existen para el doctor seleccionado. La información que debe mostrar en el control DataGridView es el siguiente:

- Número de cita.
- Fecha y Hora de la cita.
- Descripción del tipo de consulta.
- Nombre completo del cliente.

Se deberá crear como **mínimo** una nueva ventana (Windows Form) para resolver los requerimientos solicitados anteriormente.

Reporte Citas por Cliente

Esta opción debe permita al usuario visualizar las citas por un cliente en específico, debe contar con la funcionalidad de que el usuario seleccione un cliente que ya se encuentre registrado en el arreglo de cliente y el sistema muestre solamente las citas que existen para el cliente seleccionado. La información que debe mostrar en el control DataGridView es el siguiente:

- Número de cita.
- Fecha y Hora de la cita.
- Descripción del tipo de consulta.
- Nombre completo del doctor.

Se deberá crear como **mínimo** una nueva ventana (Windows Form) para resolver los requerimientos solicitados anteriormente.

Consideraciones técnicas

Uso de comentarios:

Se deberá implementar el uso de comentarios cuando se realiza la codificación del programa, en la parte superior indicar la universidad, el cuatrimestre, el nombre y la descripción del proyecto, el nombre del estudiante y la fecha en que está realizando la programación. Lo anterior debe estar en cada archivo cs (clase, program, etc) que se realiza el código de programación; como se puede observar en el siguiente ejemplo:

```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6
7   /* UNED III Cuatrimestre
8    * Proyecto XX: Descripción del Proyecto
9    * Estudiante:
10   * Fecha:
11
12  */
13
14  namespace CapaEntidades
15  {
16
```

Uso de Programación Orientada a Objetos

Debe utilizar **POO** (Programación orientada a objetos) para resolver el problema. Implementar en la Solución proyectos independientes con el fin de tener las distintas capas: Capa de Entidades, Capa de Lógica Negocio y Capa de Presentación (Interfaz del Usuario).

- Las clases de objetos (Tipos de Consulta. Clientes, Doctores y Citas) no deben contener lógica para solicitar información al usuario, solo debe tener la estructura de la clase y sus propiedades.
- Las clases no deben tener métodos vacíos.
- Cada clase debe ser creada en archivo por separado.

Uso de arreglos

- No deben ser arreglos de tipo “string” ni “int”, deben ser arreglos de objetos de las clases definidas.
- Cada vez que se agrega un objeto al arreglo se debe preguntar al usuario si desea agregar otro y no obligar al usuario a ingresar los 20 registros de una vez.
- No debe utilizar colecciones, solo se permite el uso de arreglos para este proyecto.

Debe implementar el manejo de excepciones

- Si ocurre una excepción, el sistema no debe cerrarse, se debe mostrar un mensaje al usuario y manejar la excepción de forma apropiada.

Interfaz de usuario

- Debe usar interfaz de usuario con formularios GUI.

Honestidad Académica

https://audiovisuales.uned.ac.cr/play/player/23048

Nota Importante

Cada estudiante es responsable del contenido que entrega, si no es el archivo correcto, no podrá entregarlo posterior a la fecha establecida.

Si el contenido del archivo coincide con algún otro estudiante, o se comprueba que no es de su autoría, se expone a las sanciones indicadas en la plataforma en el documento Lineamientos ante casos de plagio

Indicaciones Importantes

- **Es obligatorio que incluya todo el directorio donde se encuentra < Proyecto1>.**
- **El proyecto 1** debe estar desarrollado en **Visual Studio Community 2019 o superior** que es la herramienta oficial del curso.
- El programa debe ser modular, utilizando de la mejor manera funciones definidas por usted.
- **Los trabajos deben realizarse en forma individual. Dentro del código del programa debe de indicar la documentación que explique cómo fue realizado el programa.**
- **Si utiliza código de algún ejemplo del libro, del algún profesor o de otra fuente que no sea de su autoría, debe de indicarlo.**
- Comprima todos los archivos en un solo archivo .zip o .rar.
- **Nombre del archivo que envía**: debe ser nombre y primer apellido del estudiante, y nombre del proyecto**. Ejemplo: JuanRojas-proyecto1**.
- La entrega de la **<Proyecto1>**en las fechas establecidas en la plataforma de aprendizaje en línea Moodle en el apartado que se indique.
- Si no concluyó a tiempo la asignación, debe entregar lo que pudo hacer e incluir una carta explicando las razones por las cuales no finalizó.

Rúbrica de Evaluación

Criterio	Cumple a satisfacción lo indicado en la evaluación	Cumple medianamente en lo indicado en la evaluación	Cumple en contenido y formato pero los aportes no son significantes	No cumple o no presenta lo solicitado
Presentación del código: Correcto uso de variables, nombres de métodos significativos, no presenta métodos vacíos.	5	3	2	0.5
POO y creación de las clases (Tipos de Consulta, Clientes, Doctores, Citas)	10	5	2	0.5
Menú Principal	5	3	2	0.5
Registrar Tipos Consulta	5	3	2	0.5
Registrar Clientes	13	8	4	1
Registrar Doctores	13	8	4	1
Registrar Citas	15	10	5	1
Reporte de Citas por Fecha	5	3	2	0.5
Reporte de Citas por Doctor	5	3	2	0.5
Reporte de Citas por Cliente	5	3	2	0.5
Correcto uso del manejo de excepciones	5	3	2	0.5
Uso de comentarios con los datos solicitados en el proyecto	4	3	2	0.5
Interfaz de usuario de consola es fácil de usarse y presenta buen diseño.	10	5	2	0.5
TOTAL	**100**			