I Proyecto
Hospital Veterinario

Fecha máxima de entrega: domingo 29 de setiembre.
Equipos de trabajo: uno o dos estudiantes por equipo.

Temas:

- Principios del PPOO
- Adecuada abstracción y encapsulamiento
- Memoria dinámica
- Vectores dinámicos implementados mediante clases tipo colección

Requerimientos del proyecto:

La Universidad Nacional ha visto crecer su clínica veterinaria, la cual, cada vez cuenta con más especialidades, doctores y pacientes.

El hospital veterinario requiere sus servicios como programador para realizar un sistema computacional que permita automatizar su servicio de citas médicas.

El sistema deberá implementar las siguientes funcionalidades específicas:

- Ingresar y listar especialidades.
- Ingresar y listar doctores. Al ingresar un doctor este debe ser asociado a una especialidad existente.
- Registrar y listar dueños de mascotas.
- Registrar y listar mascotas. Cada nueva mascota debe asociarse con un dueño existente (recordar que un mismo dueño puede tener **n** mascotas).
- Asignación de citas. Antes de brindar una cita, el paciente, el dueño y el doctor deben haber sido ingresados al sistema. Antes de asignar una cita se debe visualizar adecuadamente la agenda del doctor especifico. No se pueden brindar citas en horarios ocupados. A cada cita se le asigna una hora de tiempo. Solo se maneja la agenda de la semana actual, de lunes a sábado de las 8:00 a 19:00 hrs.
- Se debe permitir realizar cambios y/o cancelaciones de citas asignadas
- En cuanto búsquedas, el sistema deberá permitir:
    - Búsqueda de un paciente específico, mostrando su historial de citas.
    - Búsqueda de un doctor específico, mostrado el listado de todos sus pacientes.

Se espera que el sistema ofrezca un menú de opciones adecuado que permita ingresar y administrar cómodamente la funcionalidad del sistema.

Genere una interfaz agradable e intuitiva para el usuario. No olvide limpiar la pantalla cada que ves que se requiera. Utilice interfaces limpias y claras.

Para cada vector que se requiera, usted, como programador, determina su tamaño o capacidad máxima adecuada.

Todos los objetos y colecciones deberán crearse con asignación de memoria dinámica.

Detalle de la Interfaz de Usuario:

Nota: a continuación se presenta una Guía de puntos mínimos que debe tener la interfaz de usuario, puede realizar cambios si lo considera.

Menú Principal

1- Submenú Administración
2- Submenú Control de Citas
3- Submenú Búsquedas y Listados

Ingrese la opción: _

El menú principal permitirá acceder a los diferentes submenús del programa

Submenú Administración

(1) Ingresar Especialidades
(2) Ingresar Doctor (por especialidad)
(3) Ingresar Dueño
(4) Ingresar Mascota (por dueño)
(0) Regresar al Menú Principal

Ingrese la opción: _

Recuerde que no se puede ingresar un doctor en una especialidad no existente.

Recuerde que no se puede ingresar una mascota para un dueño no existente.

Submenú Control Citas

(1) Sacar Cita
(2) Cancelar Cita
(3) Mostrar Calendario de Citas por Doctor
(4) Mostrar Citas por Dueño
(0) Regresar al menú Principal

Ingrese la opción: _

Búsquedas y Listados

(1) Mostrar Listado de Especialidades
(2) Mostrar Listado de Doctores por Especialidad
(3) Mostrar Dueños con sus Mascotas
(4) Mostrar Pacientes por Doctor
(0) Regresar al Menú Principal

Ingrese la opción: _

- Para **sacar cita**, se debe pedir el número de identificación del dueño y mostrar la listas de sus mascotas. Se debe elegir la mascota deseada de la lista mostrada.

Para sacar cita también se debe pedir la especialidad deseada y mostrar la lista de doctores disponibles en esa especialidad, de modo tal que el usuario pueda elegir el doctor deseado.

Se debe mostrar el horario semanal del doctor seleccionado para que el usuario pueda elegir la fecha y hora para la cita.

- Para **cancelar una cita**, se debe pedir el número de identificación del dueño y mostrar los doctores con los cuales tiene citas asignadas. Una vez seleccionado el doctor, se deben mostrar las citas asignadas al dueño o mascota con ese doctor para que el usuario seleccione la cita por cancelar.

- El calendario de citas por doctor se debe mostrar en formato de tabla con los 6 días y todas las horas de consulta. Se debe destacar o marcar las horas ocupadas por citas asignadas.

- Al **Mostrar Citas por Dueño**, se debe mostrar todos las mascotas del dueño, con el objetivo que el usuario pueda elegir la mascota de la cual desea visualizar las citas.

Condiciones Generales:

- Se evalúa diseño, funcionalidad y eficiencia del sistema implementado.
- La estructura de datos que se utilice debe ser vectores dinámicos.
- El programa debe correr y no tener errores de compilación. Si no corre o tiene errores de compilación se calificará con cero.
- En los casos que se requiera utilizar conjuntos de datos, se deberán implementar clases tipo colección creadas por el propio programador. No se podrán utilizar las colecciones provistas por la STL (Standard Template Library) o cualquier por otra biblioteca de C++. Si se incumple esta directriz, se asignará una calificación de cero.
- En proyectos programación, el main() deberá quedar con la menor cantidad de código posible: básicamente sin funcionalidad. Funcionará únicamente como punto de inicio del programa.
- En la definición de las clases se deben separar correctamente su declaración y su implementación, utilizando archivos de cabecera (archivos .h) para la declaración y archivos de código fuente (archivos .cpp) para la implementación. Los archivos de cabecera deberán siempre usar guardas (#ifndef, etc).
- Las clases tipo entidad no deberán contener código de entrada/salida (como, por ejemplo, salida a consola usando **cout**, o lectura por medio de **cin**).
- Para ser evaluado, el programa debe tener implementadas, por lo menos, el 20% de las **funcionalidades específicas** descritas en el apartado **Requerimientos del Proyecto**.
- En caso de detectarse plagio en alguno de los métodos o en cualquier parte del programa, se asignará cero a la calificación del proyecto.
- Queda a criterio de cada profesor solicitar la defensa oral del proyecto, dentro o fuera del horario de clase del curso.
- El proyecto se realizará individualmente o en grupos de dos estudiantes máximo y se entregará por medio del aula virtual, en la actividad correspondiente. No se recibirán documentos ni materiales por otro medio, salvo indicación manifiesta del profesor.
- No se aceptará la entrega tardía del proyecto. Evite subir el proyecto a último momento pues corre el riesgo de presentarse cualquier imprevisto(falta de luz, caída del internet, falla de la maquina etc).
- Recuerde incluir en los documentos de entrega un archivo bloc de notas llamado equipo de trabajo, que incluya el nombre completo de cada integrante del equipo.
- Solo uno de los integrantes debe entregar el proyecto.
- Elabore y entregue el diseño UML de clases en formato pdf.
- Entregue el proyecto completo en el IDE formalmente solicitado por cada profesor.