**Proyecto Final de Programación**

**Nombre:** Cicely Suárez Hernández.

**Carrera:** Ciencia de la Computación.

Universidad Central de Las Villas.

El siguiente proyecto tiene como objetivo plantear la solución de la siguiente **problemática:**

La Facultad de Matemática Física y Computación (MFC) desea manejar un conjunto de información relacionada con la matrícula de los estudiantes, y se necesita un programa que debe contar con las siguientes funcionalidades:

1. Obtener la lista de todos los estudiantes de la facultad.
2. Obtener los datos de un estudiante en específico dado su carnet de identidad.
3. Obtener los datos de los 5 estudiantes con mejor promedio de la facultad.
4. Obtener los estudiantes con promedio superior o igual a una nota dada por el usuario.
5. Obtener el listado de los estudiantes ayudantes.

Para esto se brindan datos de los estudiantes como ci, nombre, apellidos, sexo, carrera, año escolar, una lista con sus notas y si es o no estudiante ayudante.

**Solución propuesta:**

Se ha implementado un sistema que utiliza clases para representar a los estudiantes y a la facultad.



**Clase Estudiante:** representa a un estudiante y tiene atributos como CI, nombre, apellidos, carrera, sexo, año escolar, notas y si es o no ayudante.

*class Estudiante {*

*protected:*

*string ci, nombre, apellidos, carrera;*

*char sexo;*

*int annoEscolar;*

*vector<double> notas;*

*bool isAyudante;*

**Clase EstudianteAyudante:** hereda de Estudiante y agrega un atributo adicional para la asignatura que imparte.

*class EstudianteAyudante : public Estudiante {*

*private:*

*string asignatura;*

**Clase Facultad:** gestiona a los estudiantes y proporciona métodos para realizar las operaciones requeridas, como obtener la lista de todos los estudiantes, obtener los datos de un estudiante específico, obtener los 5 mejores promedios de la facultad, obtener estudiantes cuyo promedio sea superior a una nota dada y obtener un listado de estudiantes ayudantes.

*class Facultad{*

*vector<Estudiante> est;*

Inicializa un vector de estudiantes con datos predefinidos. Crea un objeto de tipo Facultad con la lista de estudiantes. Muestra un menú al usuario para seleccionar las operaciones a realizar. Ejecuta la operación seleccionada por el usuario utilizando los métodos de la clase Facultad.

*double calcularPromedio(): Método que calcula el promedio de las notas del estudiante.*

*int operator<(Estudiante\* e): Sobrecarga del operador menor que para comparar estudiantes según su promedio.*

*void imprimirDatos(): Método que imprime los datos del estudiante, incluyendo su CI, nombre, apellidos, carrera, sexo, año escolar, notas y si es o no ayudante.*

*bool esEstudianteAyudante(): Método que indica si el estudiante es ayudante o no.*

*void escribir\_lista\_estudiantes(): Método que imprime la lista de todos los estudiantes de la facultad, incluyendo sus datos y promedio.*

*void obtener\_ayudantes(): Método que imprime un listado de estudiantes ayudantes, incluyendo sus datos y promedio.*

*void promedios\_superiores(): Método que imprime los estudiantes cuyo promedio es superior a una nota dada por el usuario.*

*void mejores\_5\_estudiantes(): Método que imprime los 5 mejores promedios de la facultad.*

*void buscar\_estudiante(): Método que permite buscar un estudiante por su CI e imprime sus datos, incluyendo si es ayudante o no.*

**Podemos llegar a la conclusión de que:**

Este código proporciona una solución básica para la gestión de estudiantes en una facultad y puede ser ampliado y mejorado según las necesidades específicas del usuario.

Se han implementado diversas funcionalidades clave, como el cálculo del promedio de notas, la impresión de datos de estudiantes, la identificación de estudiantes ayudantes, la búsqueda de estudiantes por CI, y la obtención de listas filtradas por criterios específicos, como los 5 mejores promedios o aquellos con promedio superior a una nota dada.

El proyecto hace un buen uso de conceptos de POO como la herencia, la sobrecarga de operadores y el uso de estructuras de datos como los vectores. Esto facilita la organización del código y su mantenimiento a largo plazo.

El sistema se ejecuta en un bucle donde se muestra un menú de opciones al usuario. El usuario puede seleccionar una de las opciones disponibles y el sistema realiza la operación correspondiente. El bucle se repite hasta que el usuario elige salir del sistema.