# Proyecto 1: Entrega 2.

## Integrantes:

Jaime Andres Torres Bermejo - 202014866

Esteban Gonzalez Amaya - 202010737

Contenido

[Proyecto 1: Entrega 2. 1](#_Toc69052193)

[Integrantes: 1](#_Toc69052194)

[Link Repositorio: 1](#_Toc69052195)

[Objetivos funcionales: 1](#_Toc69052196)

[Mundo del problema: 2](#_Toc69052197)

[Carga de archivos: 2](#_Toc69052198)

[El método “main”: 3](#_Toc69052199)

[Condiciones candidatura a grado: 3](#_Toc69052200)

[Diagrama de función. 4](#_Toc69052201)

## Link Repositorio:

Existe un repositorio en GitHub a partir del cuál se manejó el trabajo en equipo, de esa misma forma, en este están todas las modificaciones hechas al código.

Link:

<https://github.com/DPOO-JaiYEst/Proy-1>

## Objetivos funcionales:

1. La aplicación debe generar un reporte de notas de un estudiante, que incluya los cursos tomados en cada semestre, promedio por semestre, promedio acumulado, semestre según créditos, estado académico y los cursos inscritos en el semestre actual.

2. La aplicación debe diferenciar el uso que hace un estudiante de ella a los cambios que los coordinadores puedan implementar.

3.La aplicación tiene que reconocer que estudiante es candidato a grado y para ilustrarlo debe generar una lista con los cursos y requisitos aprobados y una lista con los requisitos de grado que le faltan.

4. La aplicación debe ser capaz de permitirle a los coordinadores revisar el avance de los estudiantes y añadir información relevante para sus carreras y su progresión en el programa.

5.El programa deberá revisar y validar los prerrequisitos, correquisitos y número máximo de créditos por semestre de los cursos que los estudiantes deseen inscribir.

6. La aplicación debe guardar los datos en archivos de texto.

## Mundo del problema:

Se presentaron varios cambios con respecto al modelo original, simplificando ciertas partes del diseño presentado originalmente y expandiendo otras, véase entonces que en el modelo original el mundo del problema se definía de esta forma:

Diagrama

Descripción generada automáticamente

#### Diagrama anterior del mundo del problema

En cambio, el diagrama final de la solución quedó así:

#### Diagrama actual del mundo del problema

Entre los cambios notables de ambas versiones puede resaltarse la simplificación de los cursos, pues CLEs, Tipo E, etc. Ya no son clases de por sí, sino que se manejan por medio de un array llamado “atributos”, este nos permite que un curso sea varios tipos de cursos a la vez (por ejemplo, un CBU que también es tipo E) y simplifica el código de manera bastante notable.

## Carga de archivos:

En un inicio se pensaba que el programa trabaja con archivos de tipo .csv para el almacenamiento, lectura y escritura de datos. Esta decisión se tomó debido a que ambos desarrolladores del proyecto están familiarizados con el manejo de este formato gracias a los cursos de IP y EDA. Sin embargo, este formato resultó en algunas limitaciones dentro del almacenamiento de los datos, especialmente concerniendo la sintaxis de los Strings, pues no tenemos permitido usar comas para el almacenamiento de esta información fuera de aquellas utilizadas para separar los espacios. Por lo tanto y pensando en la posibilidad de futuros problemas con esa limitación de sintaxis preferimos el uso de archivos .txt para tener mayor flexibilidad en la carga y almacenamiento de los datos.

La información no está toda en un solo archivo, sino que guarda la información de clases diferentes en diferentes archivos. Por ejemplo, el archivo “estudiante.csv“ guarda la información de los estudiantes que hayan sido registrados en el sistema. De misma forma, “curso.csv” guarda la información de los cursos, secciones y notas (que implícitamente forman el pensum). Etcétera.

## El método “main”:

El método main del proyecto está definido como “interfaz\_informe.java”, una aplicación de consola (sin interfaz gráfica) definida en su propio package ajeno al modelo. En este se presenta y da el informe. Esta clase cumple como forma de interacción para ambos tipos de usuarios contemplados en el enunciado (coordinadores y estudiantes), sin embargo, a ambos les sirve de formas diferentes. Pues al correr la aplicación presenta un input en el cual el usuario se identifica o como un estudiante o un coordinador, dependiendo de la cual se presentan las herramientas correspondientes para interactuar de la forma adecuada dependiendo del usuario. En un futuro esta identificación podría ser más específica, pero por el momento, solo se requiere la identificación como estudiante o coordinador.

Dependiendo de la opción escogida se le ofrecen diferentes opciones al usuario. Tal vez la diferencia mas notable de los dos tipos de usuario es el hecho de que en el caso del coordinador, no solo se le permite conseguir un reporte o inscribir una materia a un estudiante, sino que puede editar la información contenida en el programa en los archivos mencionados anteriormente mencionados. En cambio, la única modificación posible para un estudiante es la inscripción de materias, fuera de esto, puede consultar su candidatura a grado.

## Condiciones candidatura a grado:

El programa determina que un estudiante es candidato a grado con respecto a estas condiciones:

1. El estudiante debe de haber completado todas las materias del pensum principal (134 creditos en el caso de ISIS)

2. Haber cumplido el requisito de idioma extranjero.

3.Haber cumplido el requisito de español.

4.Haber visto Constitución y Democracia (DERE-1300)

5.Haber visto por lo menos dos (2) cursos Tipo E

6.Haber visto seis (6) créditos de tipo CLE

7.Haber visto por lo menos dos (2) créditos de tipo Epsilon

8.Haber visto por lo menos siete (7) cursos de tipo CBU, con las condiciones de:

-Ver uno en cada área mínimo

-Haber visto el curso común de Colombia (CBCC-1177)

-ver tres cursos adicionales fuera de los ya nombrados.

El programa hace un “check” de estos requerimientos y presenta un informe que avisa si el estudiante es o no candidato a grado.

### Diagrama de función.