Universidad de San Carlos de Guatemala
División de Ciencias de la Ingeniería
Ingeniería en Ciencias y Sistemas
Estructura de Datos 2021
Aux: Juan Pablo Valiente González

Proyecto #2

“Sistema de Asignación de Cursos”

Objetivos

1. Objetivo General

- Aplicar los conocimientos del curso de Estructuras de Datos en la creación de soluciones de software.

2. Objetivos Específicos

- Aplicar los conocimientos adquiridos sobre estructuras de datos lineales, matrices ortogonales, árboles binarios, árboles B y tablas de dispersión.
- Aplicar los conocimientos adquiridos sobre memoria dinámica y apuntadores en los lenguajes de programación Java.
- Utilizar la herramienta graphviz para la generación de reportes gráficos.

Descripción General

El proyecto consiste en la implementación de un sistema de asignación de cursos de la Facultad de Ingeniería. Esta aplicación será realizada utilizando un servidor web con Java y graphviz para la generación de reportes. (El servidor web es figurativo, puede ser una aplicación de escritorio)

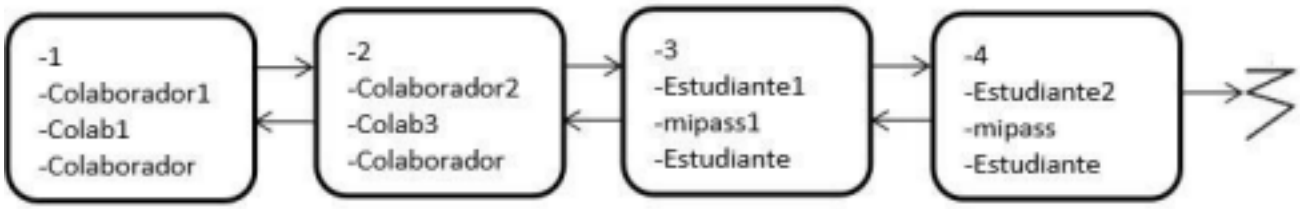
En la aplicación se tendrá el control de las siguientes entidades:

- Usuarios
- Edificios
- Salones
- Cursos
- Estudiantes
- Catedráticos
- Asignaciones

Usuarios

Se manejarán dos tipos de usuarios: Colaborador y Estudiante. Adicionalmente, se tendrá un súper usuario, que será el que podrá crear, eliminar y modificar los otros tipos de usuarios, además de las funcionalidades de colaborador y estudiante. La información de los usuarios se manejará en una lista circular doblemente enlazada. La información que debe llevar es su id (numérico), nombre, contraseña y tipo de usuario.

1. Estructura



Edificios y Salones

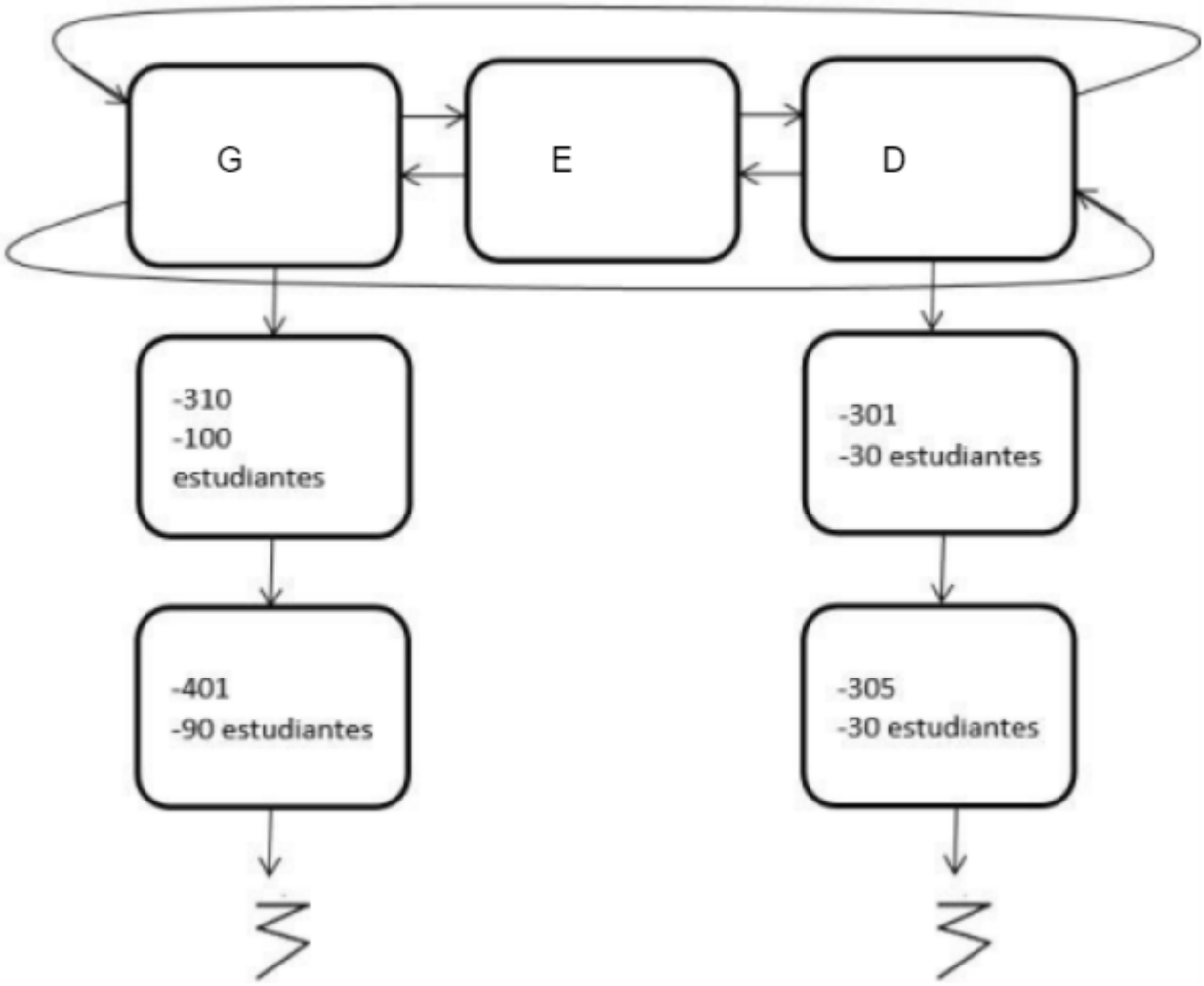
Se deben implementar las opciones de inserción, modificación y eliminación de los diferentes edificios de los que dispone la facultad para impartir los cursos. De los edificios solamente es necesario tener el nombre de los mismos. De los salones, es

necesario llevar el número de salón así como la cantidad de estudiantes que tiene de capacidad.

La estructura para guardar esta información es una lista circular doblemente enlazada. Cada nodo de la lista es un edificio y además tendrá una referencia a una lista ordenada simplemente enlazada que contendrá los salones.

Se debe implementar ABC para los edificios, así como para los salones.

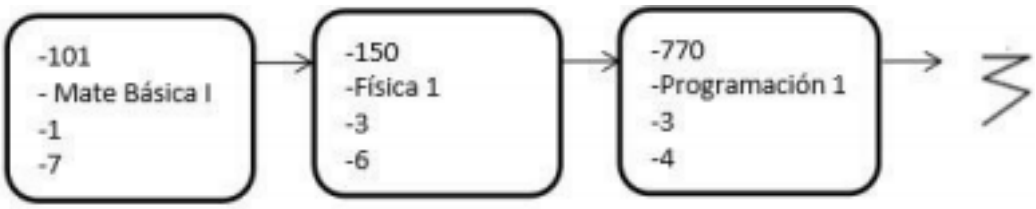
1. Estructura



Cursos

Cada curso tendrá información de su código, nombre, semestre y número de créditos. Esta información se manejará en una lista circular doblemente enlazada. Los cursos solo pueden ser creados por un colaborador o el súper usuario. Se debe implementar el CRUD respectivo.

1. Estructura

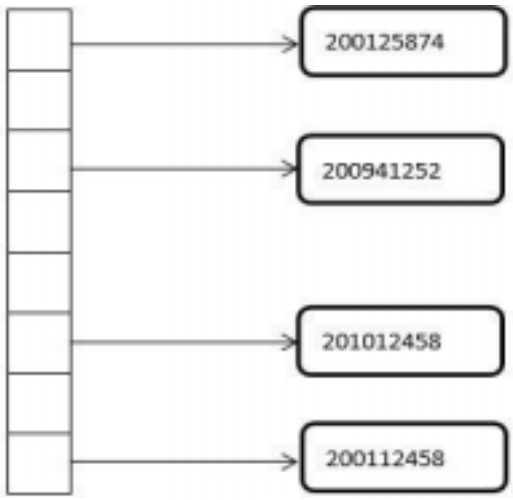


Estudiantes

Se tendrá, por cada estudiante, la siguiente información: Número de carnet (numérico y único), nombre, dirección. Esta información tendrá su propio CRUD, pero solo podrá realizarlo un colaborador.
La estructura a utilizar será una tabla hash, utilizando como llave el número de carnet:

- El tamaño inicial de la tabla hash será 37 (M = 37)
- La función de dispersión a utilizar es la siguiente: h(llv) = llv mod M
- El porcentaje máximo de ocupación será del 55%
- La política para la resolución de colisiones será la doble dispersión utilizando la siguiente función: s(llv, i) = (llv mod 7 + 1) * i.

1. Estructura

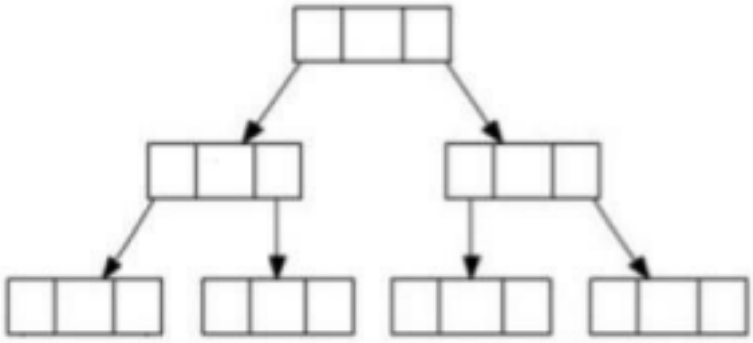


Catedráticos

Se tendrá, por cada catedrático, la siguiente información: Número de identificación (numérico y único), nombre, dirección. Esta información tendrá su propio CRUD, pero solo podrá realizarlo un colaborador.

La estructura de datos es un árbol AVL, ordenado por número de identificación.

1. Estructura



Asignación

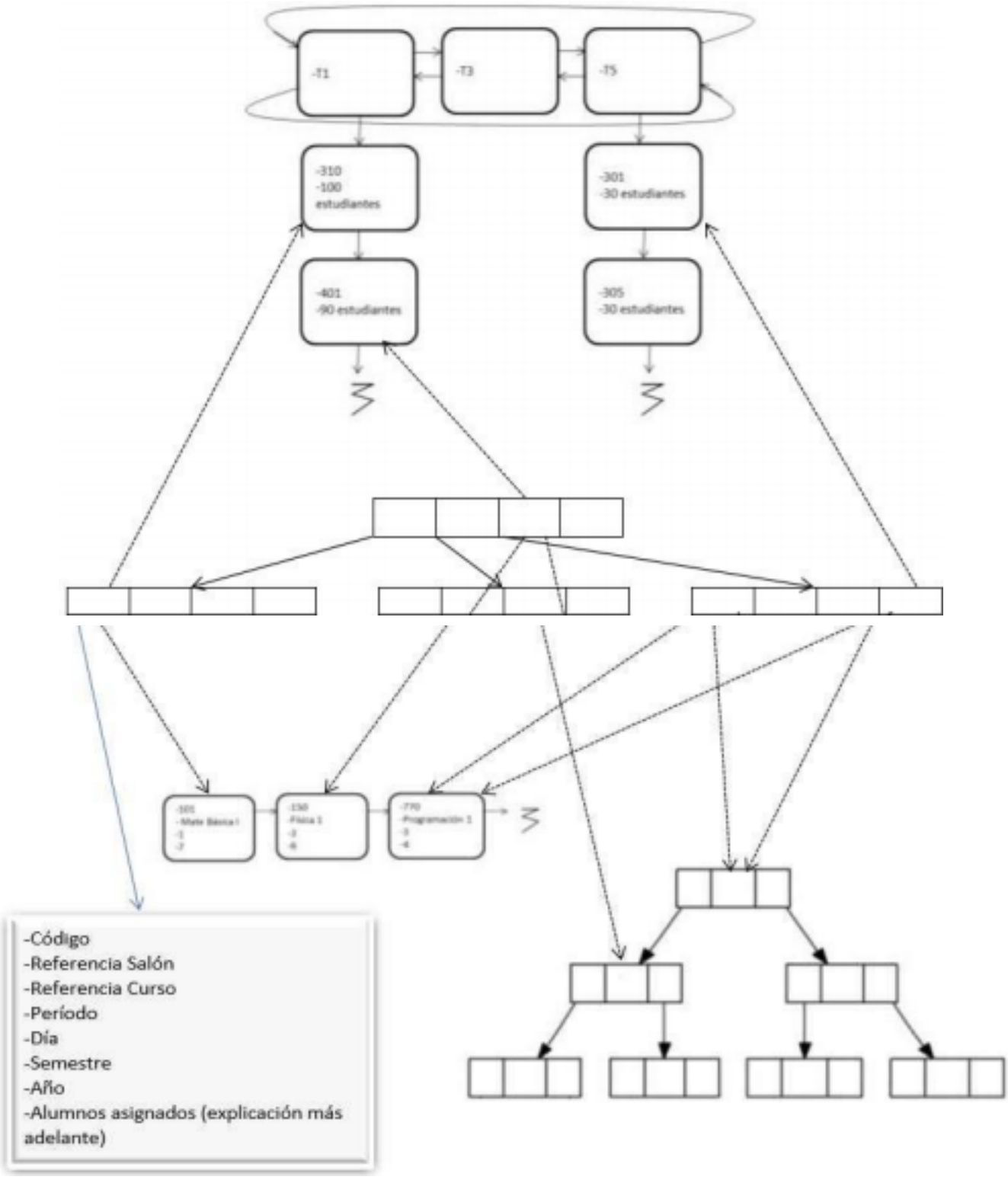
La asignación se explicará en dos pasos:(Horario y Asignación de un Estudiante a un Curso).

1. Horario

Cada curso es impartido en un horario determinado. Los horarios serán un árbol B, cada horario tiene un código, rango de hora, día, referencia al salón, referencia al curso y, además, referencia al catedrático que imparte el curso.

Esta asignación sólo puede ser realizada por un usuario de tipo colaborador.

2. Estructura

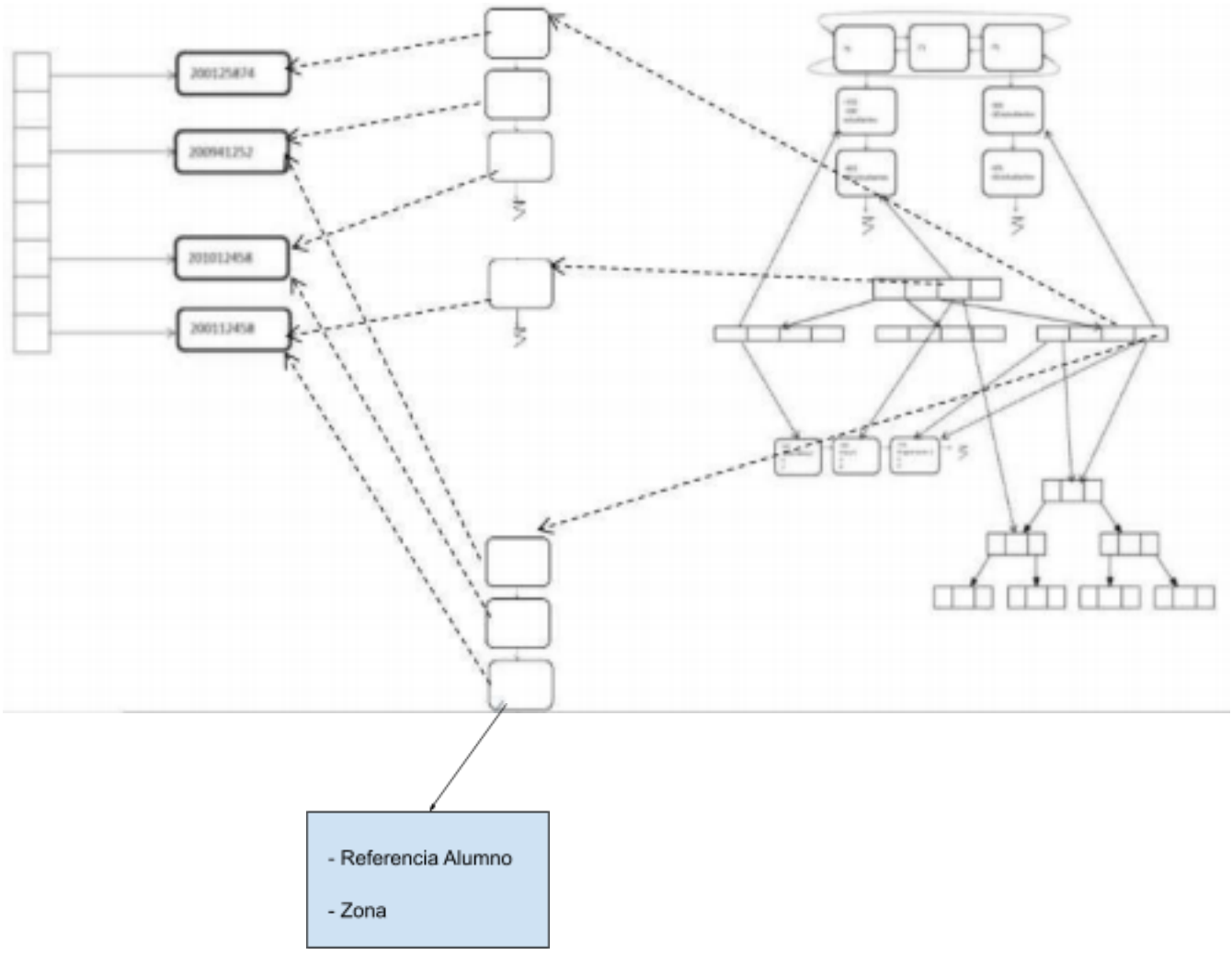


Asignación de un Estudiante a un Curso

En el nodo anterior, se debe agregar una referencia a una lista circular doblemente enlazada, que contendrá una referencia a un estudiante, zona, final. Así, todos los datos estarán relacionados mediante el árbol B.

Se debe tomar en cuenta el tamaño del salón y de esta forma limitar la cantidad de inserciones a la lista enlazada de asignación.

1. Estructura



Carga Masiva de Datos

Se podrá hacer una carga masiva de edificios, salones, cursos, estudiantes, asignación de salones a edificios, creación de horarios y asignación de estudiantes.

Esta carga sólo podrá ser realizada por el súper usuario. Se hará a través de un archivo de texto con la siguiente sintaxis:

Usuario(nombre, contraseña, tipo);
nombre = alfanumérico
contraseña = alfanumérico
tipo = alfanumérico (super, colaborador, estudiante)

Edificio(nombre);
nombre = alfanumérico

Salón(nombre, número, capacidad);
nombre = alfanumérico, donde nombre es el nombre del edificio al que se asigna el salón
número = numérico
capacidad = numérico

Curso(código, nombre, semestre, créditos);
código = numérico
nombre = alfanumérico
semestre = número
créditos = número

Estudiante(carnet, nombre, dirección);
carnet = numérico
nombre = alfanumérico
dirección = alfanumérico

Catedrático(identificador, nombre, dirección);
identificador = numérico
nombre = alfanumérico
dirección = alfanumérico

Horario(código, período, día, codCurso, codSalón, codEdificio, numIdentificacion);
código = numérico
codSalon = numérico
codCurso = numérico
codEdificio = numérico
numIdentificacion = numérico
período = alfanumérico, es un rango de horas día = alfanumérico

Asignar(carnet, codCurso, zona, final);
carnet = numérico
codCurso = numérico

zona = numérico
final = numérico

Ejemplo

```
Usuario(Secretaria, “murphychica1”, colaborador);
Usuario(Control, “control123”, colaborador);
Usuario(FMarroquín, “fffmmm”, estudiante);
Usuario(Ulises, “uliuli”, estudiante);

Estudiante(200145785, “Fernando Marroquín”, “Quetzaltenango”);
Estudiante(200745781, “Alfonso Antonio”, “Cantel”);
Estudiante(200478542, “Mario Morales”, “La Esperanza”);

Catedratico(12345,“Prof1”,“Ciudad”;
Catedratico(47854, “Prof2”, “Ciudad”);
Catedratico(12445, “Prof3”, “Ciudad”);
Catedratico(47354, “Prof4”, “Ciudad”);
Catedratico(12365, “Prof5”, “Ciudad”);

Edificio(“G”);
Edificio(“E”);
Edificio(“D”);

Salon(“G”, 12, 75);
Salon (“E”, 10, 25);
Salon (“D”, 5, 125);

Curso (777,"compiladores1", 5,4);
Curso (771,"programacion2", 4, 4);
Curso (779,"Arquitectura de computadoras y ensambladores 2", 7, 4);
Curso (732,"Estadistica 1”, 4, 4);
Curso (795,"Leguajes formales", 4, 4);

Horario(1,"7:10am-8:00am","lunes",777,105,G,12345);
Horario(2,"8:00am-8:50am","lunes",777,105,G,12345);
Horario(3,"7:10am-8:00am","viernes",777,105,G,47854);
Horario(4,"8:00am-8:50am","viernes",777,105,G,47854);
Horario(5,"7:10am-8:00am","martes",771,109,G,12345);
Horario(6,"8:00am-8:50am","martes",771,109,G,12345);
Horario(7,"7:10am-8:00am","jueves",771,109,G,47854);
Horario(8,"8:00am-8:50am","jueves",771,109,G,47854);
Horario(9,"9:10am-10:00am","martes",779,110,G,12445);
Horario(10,"10:00am-10:50am","martes",779,110,G,12445);
Horario(11,"9:10am-10:00am","jueves",779,110,G,12365);
Horario(12,"10:00am-10:50am","jueves",779,110,G,12365);

Asignar (200145785, 777, 30, 25);
```

Asignar (200745781, 777, 60, 10);
Asignar (200478542, 771, 70, 20);

Reportes

1. Graphviz

Se debe realizar el dibujo de las siguientes estructuras:

1. Usuarios
2. Edificios y salones
3. Estudiantes
4. Horarios
5. Asignación
6. Todo

Se recomienda realizar los reportes de la misma forma que se presentaron en este enunciado. Para hacerlos de esta forma utilizar subgraph.

2. Reportes en Aplicación

1. Mostrar los cursos asignados por un estudiante. Se puede hacer desde el lado de colaborador/superusuario o desde el estudiante. Si está del lado del estudiante, se pueden ver solamente los cursos que se ha asignado él mismo.
2. Mostrar a los estudiantes asignados a un curso. Se puede hacer solo por un colaborador/super usuario.
3. Mostrar qué cursos se dan en cierto salón. El colaborador ingresará el número de salón.
4. Mostrar cuántos estudiantes aprobados y reprobados hay en los cursos de un mismo semestre. El colaborador ingresa el número de semestre y se buscará en todos los cursos que sean de ese semestre.

Observaciones

- Lenguaje de programación a utilizar: Java
- Sistema Operativo: Libre
- IDE: Libre.
- Las listas deben crearse mediante clases genéricas.
- La aplicación que no genere gráficas/imágenes no podrá ser calificada, se debe utilizar Graphviz.
- Todas las estructuras deben de ser realizadas por el estudiante, sin el uso de librerías específicas de ningún IDE o Framework.
- Durante la calificación se harán preguntas para validar que el estudiante realizó la práctica, de no responder correctamente anulara la nota obtenida en la o las secciones en la que aplique

tal concepto
- La aplicación será compilada y ejecutada al momento de la calificación.
- REQUISITO MÍNIMO PARA CALIFICACIÓN: Carga masiva de datos.

Entregables

- Se debe entregar un manual de usuario con capturas de la aplicación explicando como funciona
- Se debe entregar un manual técnico explicando la organización del código de la aplicación (diagramas, etc)
- Se debe entregar el código fuente y ejecutable de la aplicación
- Todos los puntos anteriores se entregan por Github

Fecha y modo de entrega:

- [Tentativa] Jueves 20 de mayo antes de las 12:00 horas vía Classroom
- Si no se cumple con la hora de entrega, se tendrá una penalización de 10% cada 15 minutos incumplidos empezando por los 2 minutos de retraso.
- Si no se entrega en Classroom, no se calificará.
- Las copias tendrán nota de 0 puntos, serán reportadas al catedrático y a coordinación de Sistemas.