

Prueba de Caja Blanca

“Gestión de información académica InClass”

Integrantes:

Stephen Drouet
Bryan Morales
Alejandro Sarmiento
Jairo Quilumbaquin

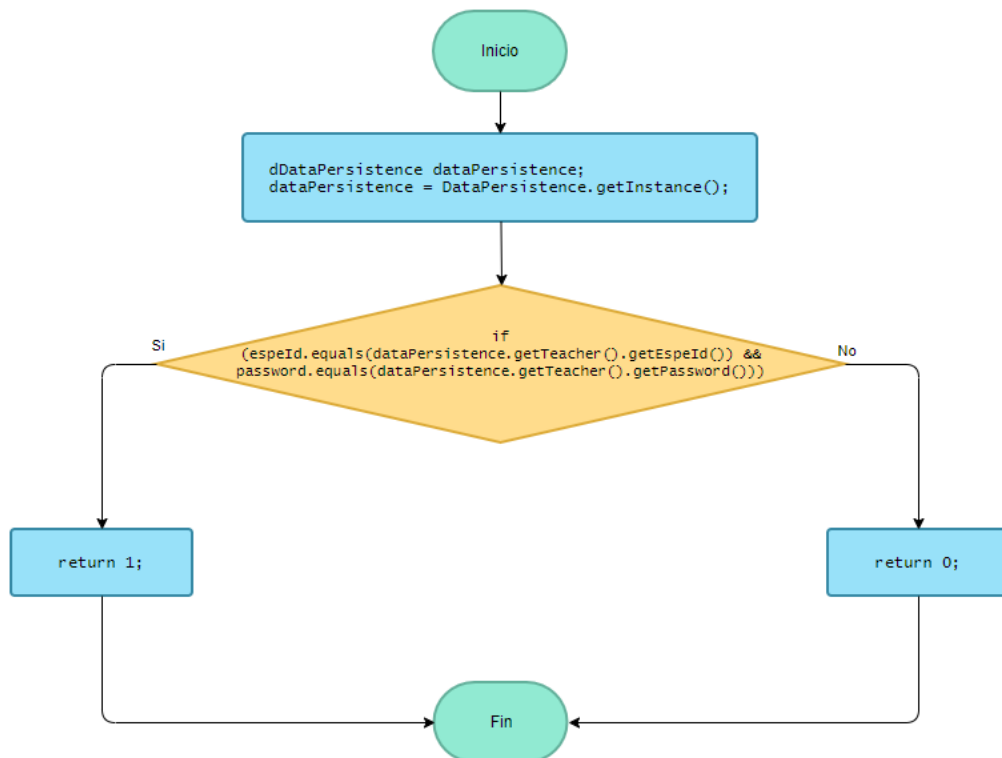
Fecha 21/02/2024

Prueba caja blanca login

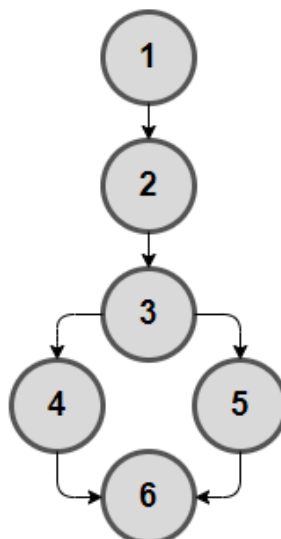
CÓDIGO FUENTE

```
public static int loginTeacher(String espeId, String password) {  
    DataPersistence dataPersistence;  
    dataPersistence = DataPersistence.getInstance();  
    if (espeId.equals(dataPersistence.getTeacher().getEspeId()) &&  
        password.equals(dataPersistence.getTeacher().getPassword())) {  
        return 1;  
    } else {  
        return 0;  
    }  
}
```

DIAGRAMA DE FLUJO



Grafo



RUTAS

R1: 1, 2, 3, 4, 6

R2: 1, 2, 3, 5, 6

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 1 + 1 = 2$
- $V(G) = A - N + 2$
 $V(G) = 6 - 6 + 2 = 2$

DONDE:

P: Número de nodos predichado

A: Número de aristas

N: Número de nodos

Prueba caja blanca calcular porcentaje de asistencias

CÓDIGO FUENTE

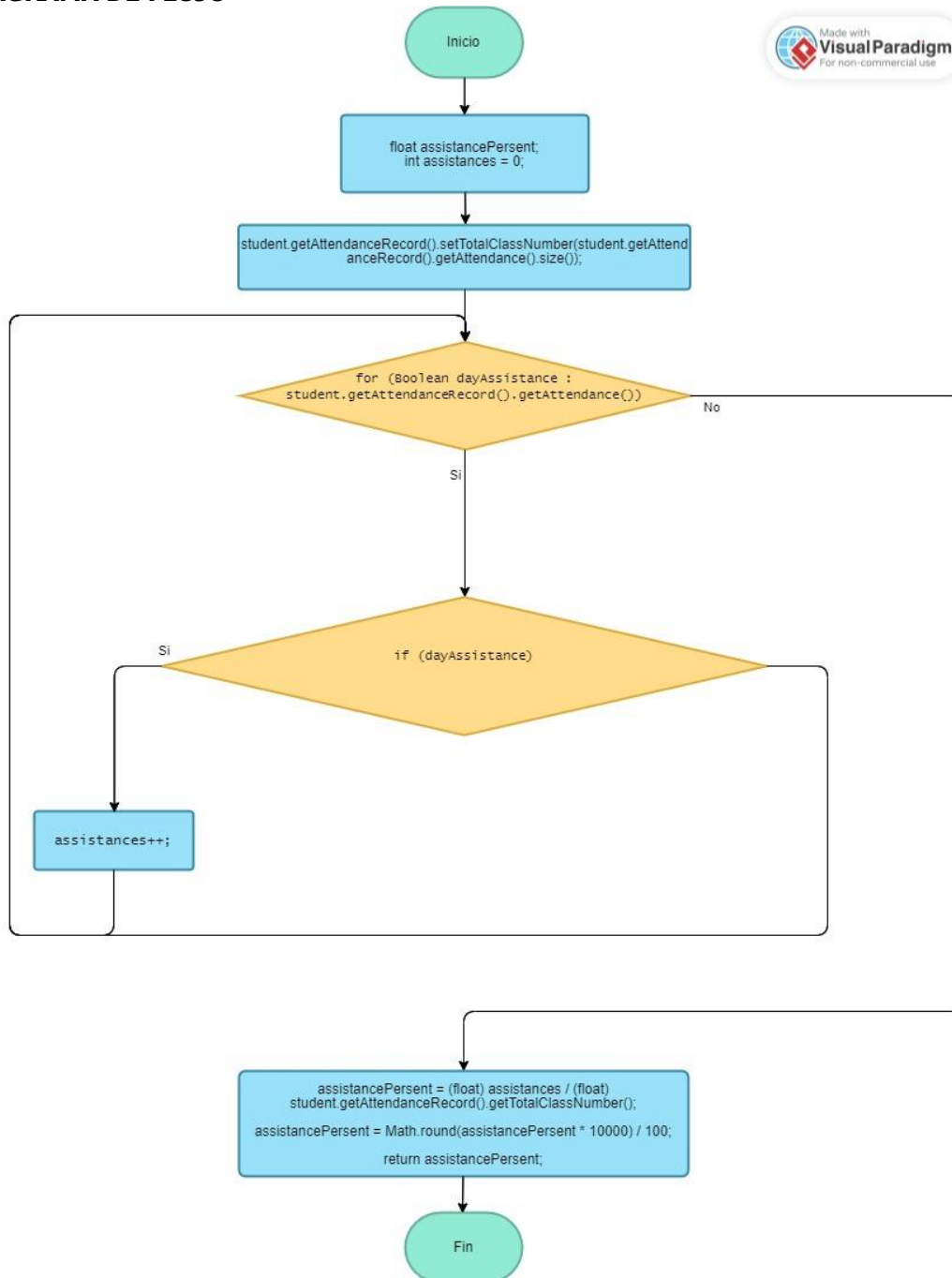
```
public static float calculateAssistancePerSetn(Student student) {
    float assistancePersent;
    int assistances = 0;
    student.getAttendanceRecord().setTotalClassNumber(student.
getAttendanceRecord().getAttendance().size());
    for (Boolean dayAssistance : student.getAttendanceRecord().
getAttendance()) {
        if (dayAssistance) {
            assistances++;
        }
    }

    assistancePersent = (float) assistances / (float)
student.getAttendanceRecord().getTotalClassNumber();

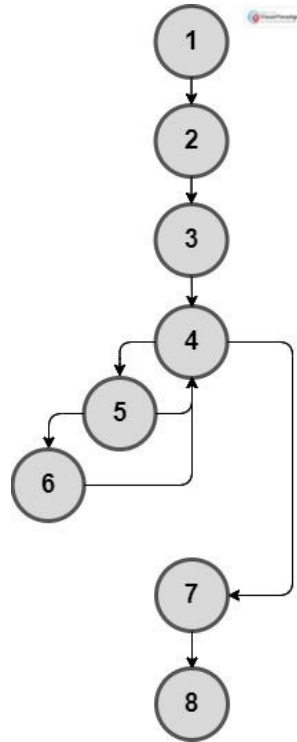
    assistancePersent = Math.round(assistancePersent * 10000) / 100;

    return assistancePersent;
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 7, 8

R2: 1, 2, 3, 4, 5, 4, 7, 8

R3: 1, 2, 3, 4, 5, 6, 4, 7, 8

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predcados(decisiones)} + 1$
 $V(G) = 2 + 1 = 3$
- $V(G) = A - N + 2$
 $V(G) = 9 - 8 + 2 = 3$

DONDE:

P: Número de nodos predcado

A: Número de aristas

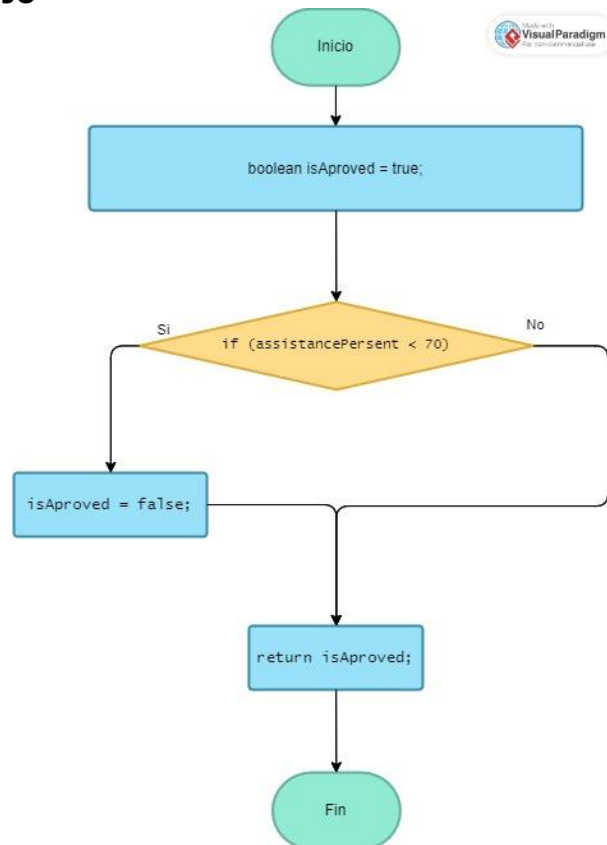
N: Número de nodos

Prueba caja blanca verificar si desaprobo por asistencias

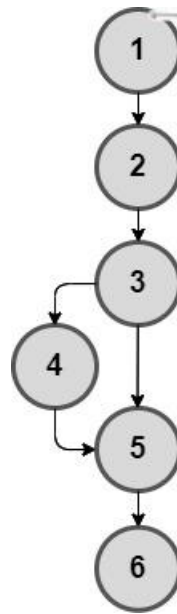
CÓDIGO FUENTE

```
public static boolean verifyAttendancePesent(float assistancePersent) {  
    boolean isAproved = true;  
  
    if (assistancePersent < 70) {  
        isAproved = false;  
    }  
  
    return isAproved;  
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 5, 6

R2: 1, 2, 3, 4, 5, 6

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 1 + 1 = 2$
- $V(G) = A - N + 2$
 $V(G) = 6 - 6 + 2 = 2$

DONDE:

P: Número de nodos predichado

A: Número de aristas

N: Número de nodos

Prueba caja blanca conectar a mongoDB

CÓDIGO FUENTE

```
public boolean connectMongoDB() {
    ArrayList<Tutorship> tutorships;
    ArrayList<Course> courses;
    ArrayList<Student> students;
    String name = "";
    String espeId = "";

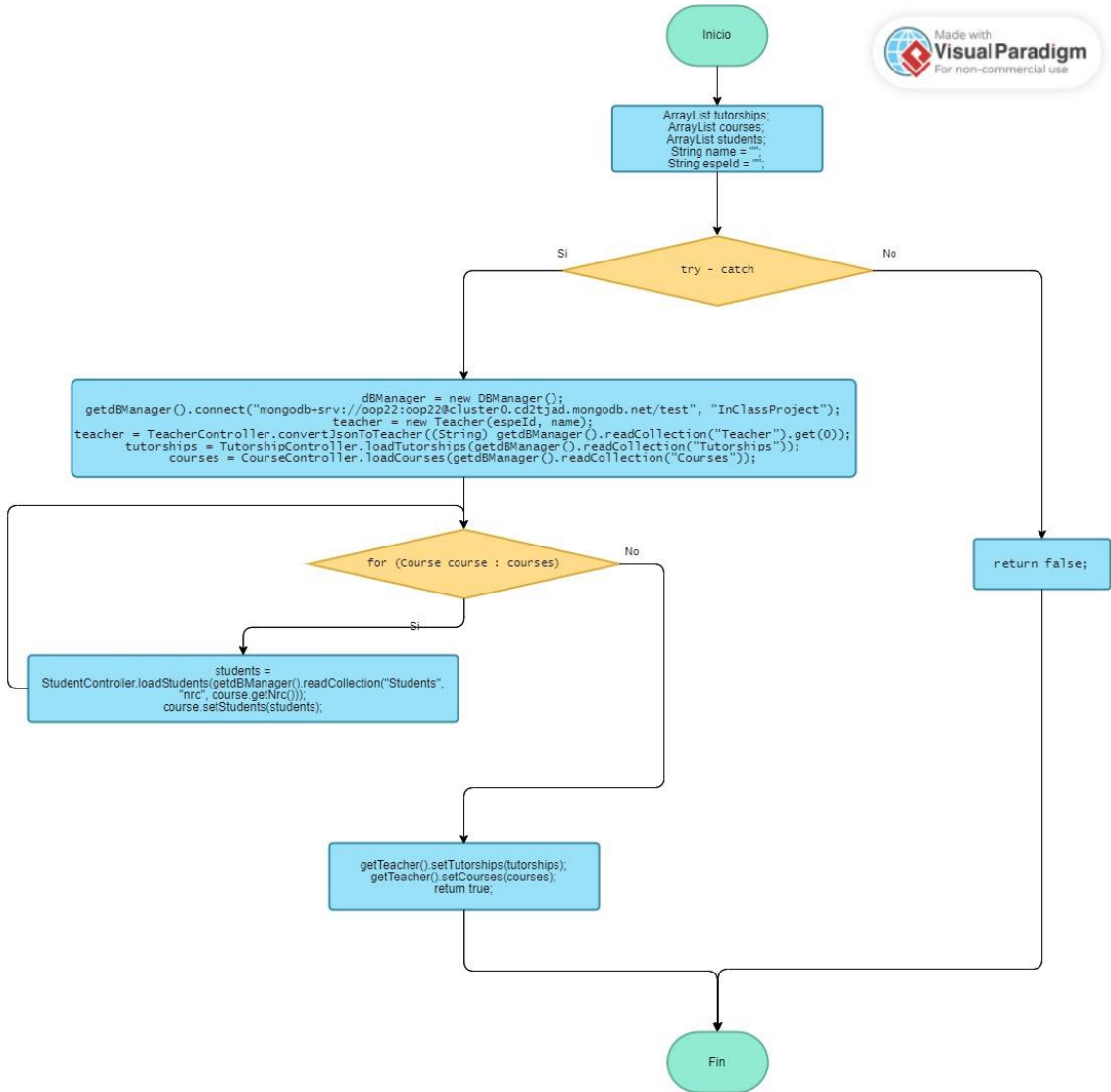
    try {
        dbManager = new DBManager();

        getdbManager().connect("mongodb+srv://oop22:oop22@cluster0.cd2tjad.mongodb.net
/test", "InClassProject");
        teacher = new Teacher(espeId, name);
        teacher = TeacherController.convertJsonToTeacher((String)
getdbManager().readCollection("Teacher").get(0));
        tutorships =
TutorshipController.loadTutorships(getdbManager().readCollection("Tutorships")
);
        courses =
CourseController.loadCourses(getdbManager().readCollection("Courses"));

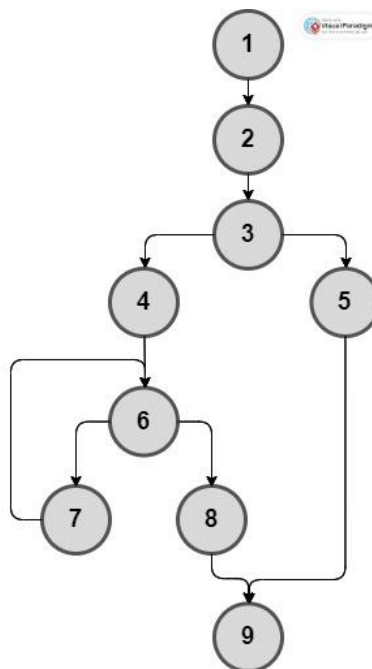
        for (Course course : courses) {
            students =
StudentController.loadStudents(getdbManager().readCollection("Students",
"nrc", course.getNrc()));
            course.setStudents(students);
        }

        getTeacher().setTutorships(tutorships);
        getTeacher().setCourses(courses);
        return true;
    } catch (Exception e) {
        return false;
    }
}
```


DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 5, 9

R2: 1, 2, 3, 4, 6, 8, 9

R3: 1, 2, 3, 4, 6, 7, 6, 8, 9

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 2 + 1 = 3$
- $V(G) = A - N + 2$
 $V(G) = 10 - 9 + 2 = 3$

DONDE:

P: Número de nodos predichado

A: Número de aristas

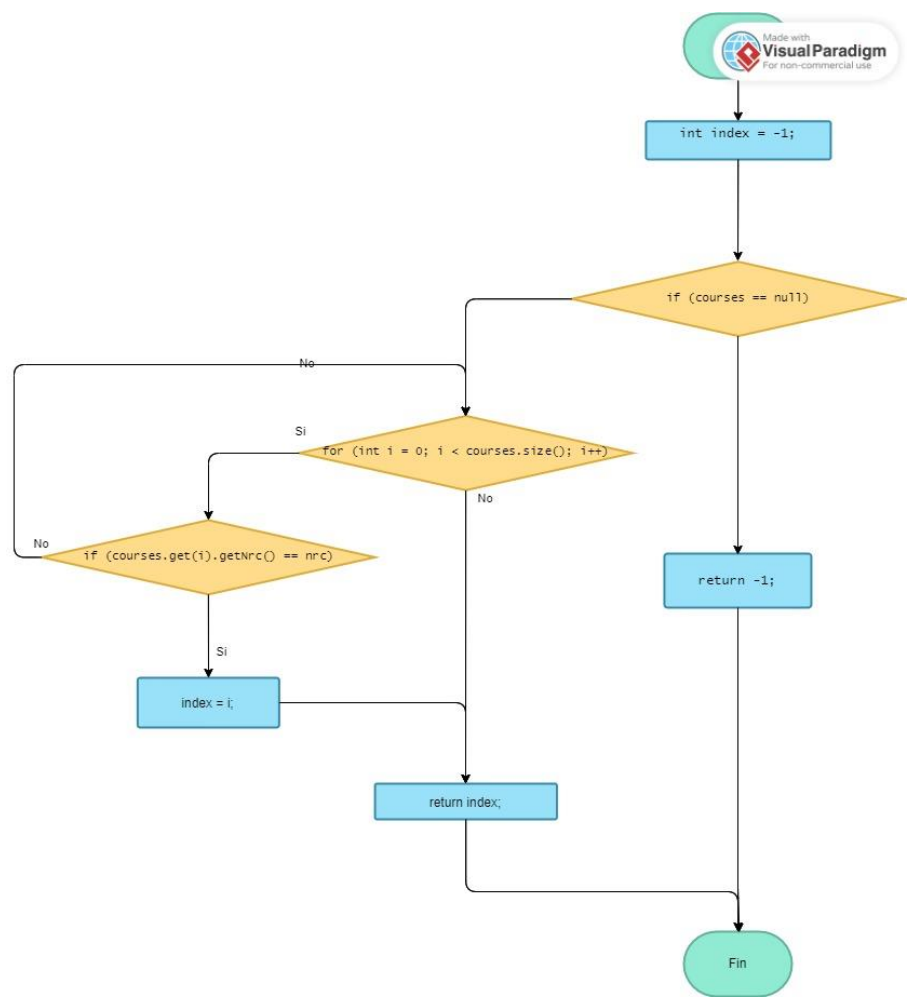
N: Número de nodos

Prueba caja blanca conectar buscar cursos

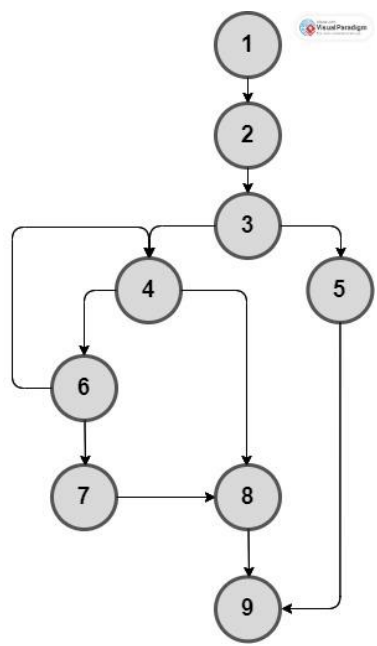
CÓDIGO FUENTE

```
public static int findCourse(ArrayList<Course> courses, int nrc) {  
    int index = -1;  
  
    if (courses == null) {  
        return -1;  
    }  
  
    for (int i = 0; i < courses.size(); i++) {  
        if (courses.get(i).getNrc() == nrc) {  
            index = i;  
            return index;  
        }  
    }  
  
    return index;  
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 5, 9

R2: 1, 2, 3, 4, 8, 9

R3: 1, 2, 3, 4, 6, 7, 8, 9

R4: 1, 2, 3, 4, 6, 4, 8, 9

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 3 + 1 = 4$
- $V(G) = A - N + 2$
 $V(G) = 11 - 9 + 2 = 4$

DONDE:

P: Número de nodos predichado

A: Número de aristas

N: Número de nodos