Guatemala 13 de Mayo de 2024
Adrián López 231361

Algoritmos y Estructuras de Datos
HDT 10: Grafos Algoritmo de Floyd

Objetivos:
a. Implementación de un grafo en Java.
b. Implementación del algoritmo de Floyd (camino más corto entre cualquier par de vértices del grafo) en Java.
c. Usar los resultados del algoritmo de Floyd para calcular el centro del grafo, en Java.
d. Opcional: usar el módulo de Python NetworkX para el algoritmo de Floyd.

"El programa debe de leer un archivo.txt que representa el grafo llamado guategrafo.txt donde cada línea tiene el nombre de la ciudad1, nombre de la ciudad2 y la distancia en KM que hay desde la ciudad1 a la ciudad2 (recuerde que es un grafo con direcciones). Luego de leer el archivo, se construye el grafo y se aplica el algoritmo de Floyd para calcular la distancia más corta entre cualquier par de ciudades y cuál es la ciudad que queda en el centro del grafo. Se debe mostrar la matriz de adyacencia que representa el grafo (o la representación que usted haya implementado)."

Link de Repositorio:
https://github.com/adirnnn/HDT10-Grafos-Algoritmo-Floyd.git

a. Implementación del Grafo:
Todo lo relacionado a la implementación de Java del grafo se puede encontrar en la clase Grafo.java

b. Implementación del Algoritmo de Floyd:

```java
// Implementación del algoritmo de Floyd-Warshall para calcular las rutas más cortas entre todas las parejas de ciudades
public void aplicarFloydWarshall() {
    for (String intermedia : obtenerCiudades()) {
        for (String origen : obtenerCiudades()) {
            for (String destino : obtenerCiudades()) {
                int distanciaOrigenIntermedia = obtenerDistancia(origen, intermedia);
                int distanciaIntermediaDestino = obtenerDistancia(intermedia, destino);
                int distanciaOrigenDestino = obtenerDistancia(origen, destino);
                int nuevaDistancia = distanciaOrigenIntermedia != Integer.MAX_VALUE &&
                        distanciaIntermediaDestino != Integer.MAX_VALUE ?
                        distanciaOrigenIntermedia + distanciaIntermediaDestino : Integer.MAX_VALUE;
                if (distanciaOrigenDestino > nuevaDistancia) {
                    agregarConexion(origen, destino, nuevaDistancia);
                }
            }
        }
    }
}
```

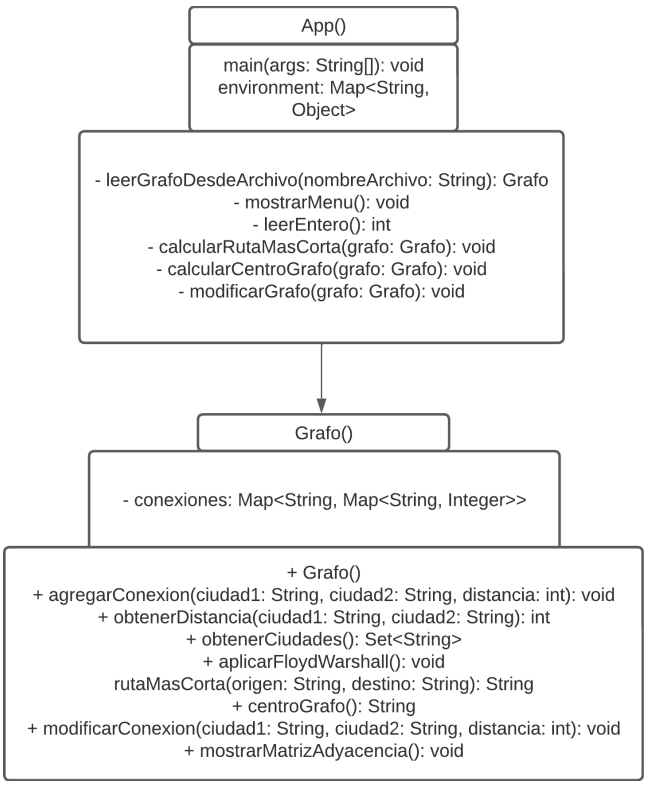
c. Implementación del cálculo del centro del grafo:

```java
// Método para calcular la ciudad que queda en el centro del grafo
public String centroGrafo() {
    String centro = null;
    int menorDistancia = Integer.MAX_VALUE;
    for (String ciudad : obtenerCiudades()) {
        int mayorDistancia = 0;
        for (String otraCiudad : obtenerCiudades()) {
            if (!ciudad.equals(otraCiudad)) {
                mayorDistancia = Math.max(mayorDistancia, obtenerDistancia(ciudad, otraCiudad));
            }
        }
        if (menorDistancia > mayorDistancia) {
            menorDistancia = mayorDistancia;
            centro = ciudad;
        }
    }
    return "El centro del grafo es: " + centro;
}
```

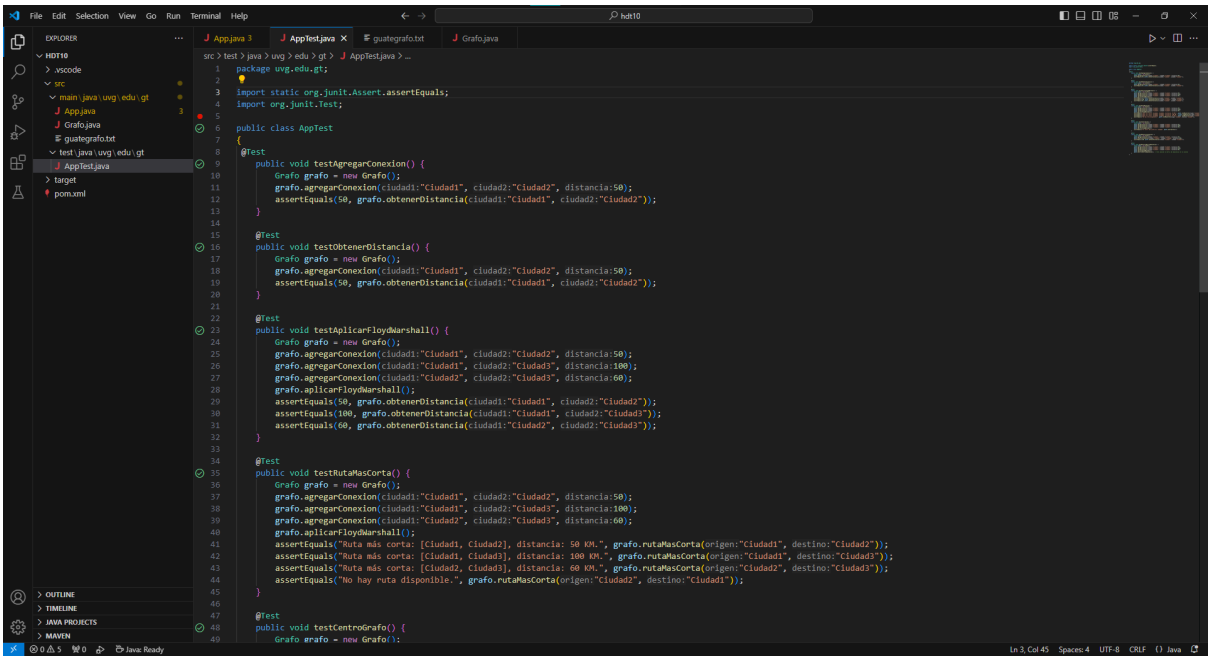
d. Programa Principal:

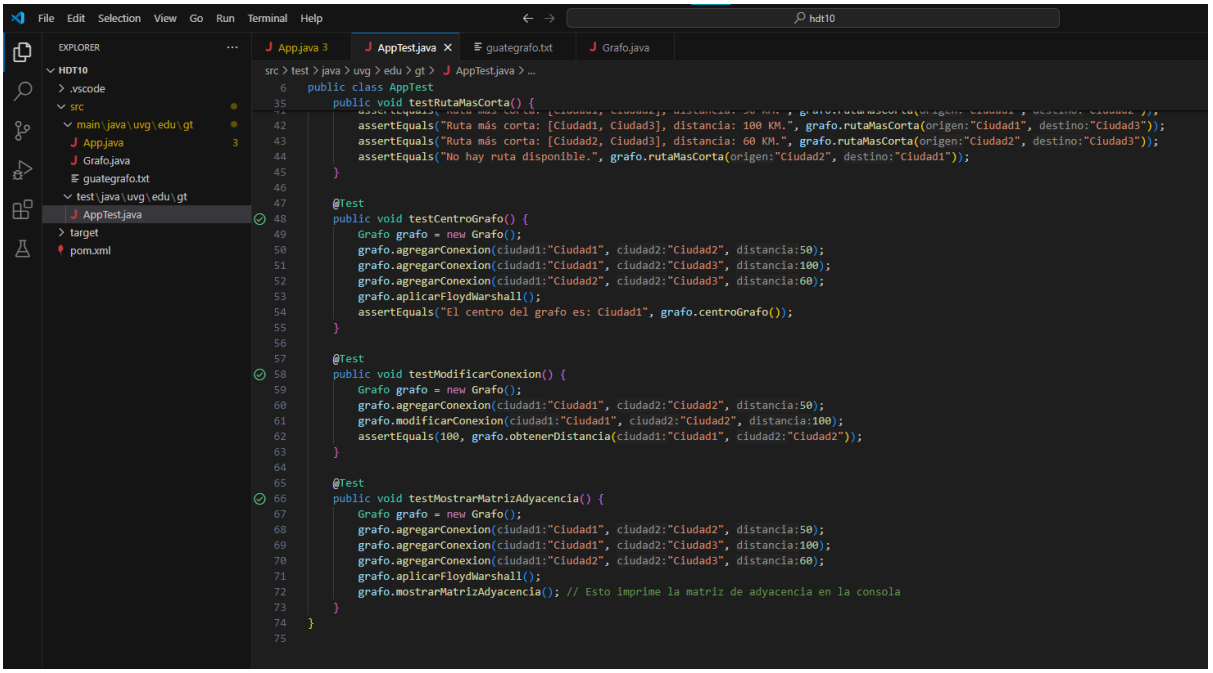
Todo lo relacionado al programa principal se encuentra en la clase de App.java

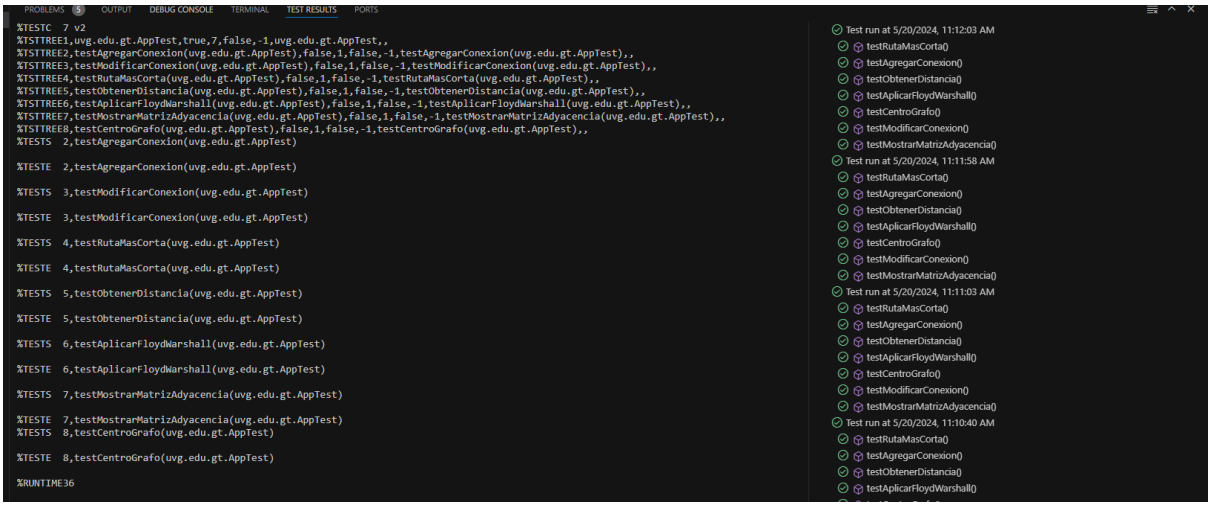
e. Diagrama UML:


App()
main(args: String[]): void
environment: Map<String, Object>
- leerGrafoDesdeArchivo(nombreArchivo: String): Grafo
- mostrarMenu(): void
- leerEntero(): int
- calcularRutaMasCorta(grafo: Grafo): void
- calcularCentroGrafo(grafo: Grafo): void
- modificarGrafo(grafo: Grafo): void
Grafo()
- conexiones: Map<String, Map<String, Integer>>
+ Grafo()
+ agregarConexion(ciudad1: String, ciudad2: String, distancia: int): void
+ obtenerDistancia(ciudad1: String, ciudad2: String): int
+ obtenerCiudades(): Set<String>
+ aplicarFloydWarshall(): void
rutaMasCorta(origen: String, destino: String): String
+ centroGrafo(): String
+ modificarConexion(ciudad1: String, ciudad2: String, distancia: int): void
+ mostrarMatrizAdyacencia(): void

f. JUnit:


File Edit Selection View Go Run Terminal Help
hdt10
EXPLORER
HDT10
.vscode
src
main\java\uvg\edu\gt
App.java 3
Grafo.java
guategrafo.txt
test\java\uvg\edu\gt
AppTest.java
target
pom.xml
App.java 3
AppTest.java
guategrafo.txt
Grafo.java
src > test > java > uvg > edu > gt > AppTest.java > ...
package uvg.edu.gt;
import static org.junit.Assert.assertEquals;
import org.junit.Test;
public class AppTest
{
@Test
public void testAgregarConexion() {
Grafo grafo = new Grafo();
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:50);
assertEquals(50, grafo.obtenerDistancia(ciudad1:"Ciudad1", ciudad2:"Ciudad2"));
}
@Test
public void testObtenerDistancia() {
Grafo grafo = new Grafo();
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:50);
assertEquals(50, grafo.obtenerDistancia(ciudad1:"Ciudad1", ciudad2:"Ciudad2"));
}
@Test
public void testAplicarFloydWarshall() {
Grafo grafo = new Grafo();
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:50);
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad3", distancia:100);
grafo.agregarConexion(ciudad1:"Ciudad2", ciudad2:"Ciudad3", distancia:60);
grafo.aplicarFloydWarshall();
assertEquals(50, grafo.obtenerDistancia(ciudad1:"Ciudad1", ciudad2:"Ciudad2"));
assertEquals(100, grafo.obtenerDistancia(ciudad1:"Ciudad1", ciudad2:"Ciudad3"));
assertEquals(60, grafo.obtenerDistancia(ciudad1:"Ciudad2", ciudad2:"Ciudad3"));
}
@Test
public void testRutaMasCorta() {
Grafo grafo = new Grafo();
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:50);
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad3", distancia:100);
grafo.agregarConexion(ciudad1:"Ciudad2", ciudad2:"Ciudad3", distancia:60);
grafo.aplicarFloydWarshall();
assertEquals("Ruta más corta: [Ciudad1, Ciudad2], distancia: 50 KM.", grafo.rutaMasCorta(origen:"Ciudad1", destino:"Ciudad2"));
assertEquals("Ruta más corta: [Ciudad1, Ciudad3], distancia: 100 KM.", grafo.rutaMasCorta(origen:"Ciudad1", destino:"Ciudad3"));
assertEquals("Ruta más corta: [Ciudad2, Ciudad3], distancia: 60 KM.", grafo.rutaMasCorta(origen:"Ciudad2", destino:"Ciudad3"));
assertEquals("No hay ruta disponible.", grafo.rutaMasCorta(origen:"Ciudad2", destino:"Ciudad1"));
}
@Test
public void testCentroGrafo() {
Grafo grafo = new Grafo();
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
Java: Ready
Ln 3, Col 45
Spaces: 4
UTF-8
CRLF
Java


File Edit Selection View Go Run Terminal Help
hdt10
EXPLORER
HDT10
.vscode
src
main\java\uvg\edu\gt
App.java 3
Grafo.java
guategrafo.txt
test\java\uvg\edu\gt
AppTest.java
target
pom.xml
App.java 3
AppTest.java
guategrafo.txt
Grafo.java
src > test > java > uvg > edu > gt > AppTest.java > ...
public class AppTest
public void testRutaMasCorta() {
assertEquals("Ruta más corta: [Ciudad1, Ciudad3], distancia: 100 KM.", grafo.rutaMasCorta(origen:"Ciudad1", destino:"Ciudad3"));
assertEquals("Ruta más corta: [Ciudad2, Ciudad3], distancia: 60 KM.", grafo.rutaMasCorta(origen:"Ciudad2", destino:"Ciudad3"));
assertEquals("No hay ruta disponible.", grafo.rutaMasCorta(origen:"Ciudad2", destino:"Ciudad1"));
@Test
public void testCentroGrafo() {
Grafo grafo = new Grafo();
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:50);
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad3", distancia:100);
grafo.agregarConexion(ciudad1:"Ciudad2", ciudad2:"Ciudad3", distancia:60);
grafo.aplicarFloydWarshall();
assertEquals("El centro del grafo es: Ciudad1", grafo.centroGrafo());
@Test
public void testModificarConexion() {
Grafo grafo = new Grafo();
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:50);
grafo.modificarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:100);
assertEquals(100, grafo.obtenerDistancia(ciudad1:"Ciudad1", ciudad2:"Ciudad2"));
@Test
public void testMostrarMatrizAdyacencia() {
Grafo grafo = new Grafo();
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad2", distancia:50);
grafo.agregarConexion(ciudad1:"Ciudad1", ciudad2:"Ciudad3", distancia:100);
grafo.agregarConexion(ciudad1:"Ciudad2", ciudad2:"Ciudad3", distancia:60);
grafo.aplicarFloydWarshall();
grafo.mostrarMatrizAdyacencia(); // Esto imprime la matriz de adyacencia en la consola
}


PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS
%TESTC 7 v2
%TSTTREE1,uvg.edu.gt.AppTest,true,7,false,-1,uvg.edu.gt.AppTest,,
%TSTTREE2,testAgregarConexion(uvg.edu.gt.AppTest),false,1,false,-1,testAgregarConexion(uvg.edu.gt.AppTest),,
%TSTTREE3,testModificarConexion(uvg.edu.gt.AppTest),false,1,false,-1,testModificarConexion(uvg.edu.gt.AppTest),,
%TSTTREE4,testRutaMasCorta(uvg.edu.gt.AppTest),false,1,false,-1,testRutaMasCorta(uvg.edu.gt.AppTest),,
%TSTTREE5,testObtenerDistancia(uvg.edu.gt.AppTest),false,1,false,-1,testObtenerDistancia(uvg.edu.gt.AppTest),,
%TSTTREE6,testAplicarFloydWarshall(uvg.edu.gt.AppTest),false,1,false,-1,testAplicarFloydWarshall(uvg.edu.gt.AppTest),,
%TSTTREE7,testMostrarMatrizAdyacencia(uvg.edu.gt.AppTest),false,1,false,-1,testMostrarMatrizAdyacencia(uvg.edu.gt.AppTest),,
%TSTTREE8,testCentroGrafo(uvg.edu.gt.AppTest),false,1,false,-1,testCentroGrafo(uvg.edu.gt.AppTest),,
%TESTS 2,testAgregarConexion(uvg.edu.gt.AppTest)
%TESTE 2,testAgregarConexion(uvg.edu.gt.AppTest)
%TESTS 3,testModificarConexion(uvg.edu.gt.AppTest)
%TESTE 3,testModificarConexion(uvg.edu.gt.AppTest)
%TESTS 4,testRutaMasCorta(uvg.edu.gt.AppTest)
%TESTE 4,testRutaMasCorta(uvg.edu.gt.AppTest)
%TESTS 5,testObtenerDistancia(uvg.edu.gt.AppTest)
%TESTE 5,testObtenerDistancia(uvg.edu.gt.AppTest)
%TESTS 6,testAplicarFloydWarshall(uvg.edu.gt.AppTest)
%TESTE 6,testAplicarFloydWarshall(uvg.edu.gt.AppTest)
%TESTS 7,testMostrarMatrizAdyacencia(uvg.edu.gt.AppTest)
%TESTE 7,testMostrarMatrizAdyacencia(uvg.edu.gt.AppTest)
%TESTS 8,testCentroGrafo(uvg.edu.gt.AppTest)
%TESTE 8,testCentroGrafo(uvg.edu.gt.AppTest)
%RUNTIME36
Test run at 5/20/2024, 11:12:03 AM
testRutaMasCorta()
testAgregarConexion()
testObtenerDistancia()
testAplicarFloydWarshall()
testCentroGrafo()
testModificarConexion()
testMostrarMatrizAdyacencia()
Test run at 5/20/2024, 11:11:58 AM
testRutaMasCorta()
testAgregarConexion()
testObtenerDistancia()
testAplicarFloydWarshall()
testCentroGrafo()
testModificarConexion()
testMostrarMatrizAdyacencia()
Test run at 5/20/2024, 11:11:03 AM
testRutaMasCorta()
testAgregarConexion()
testObtenerDistancia()
testAplicarFloydWarshall()
testCentroGrafo()
testModificarConexion()
testMostrarMatrizAdyacencia()
Test run at 5/20/2024, 11:10:40 AM
testRutaMasCorta()
testAgregarConexion()
testObtenerDistancia()
testAplicarFloydWarshall()