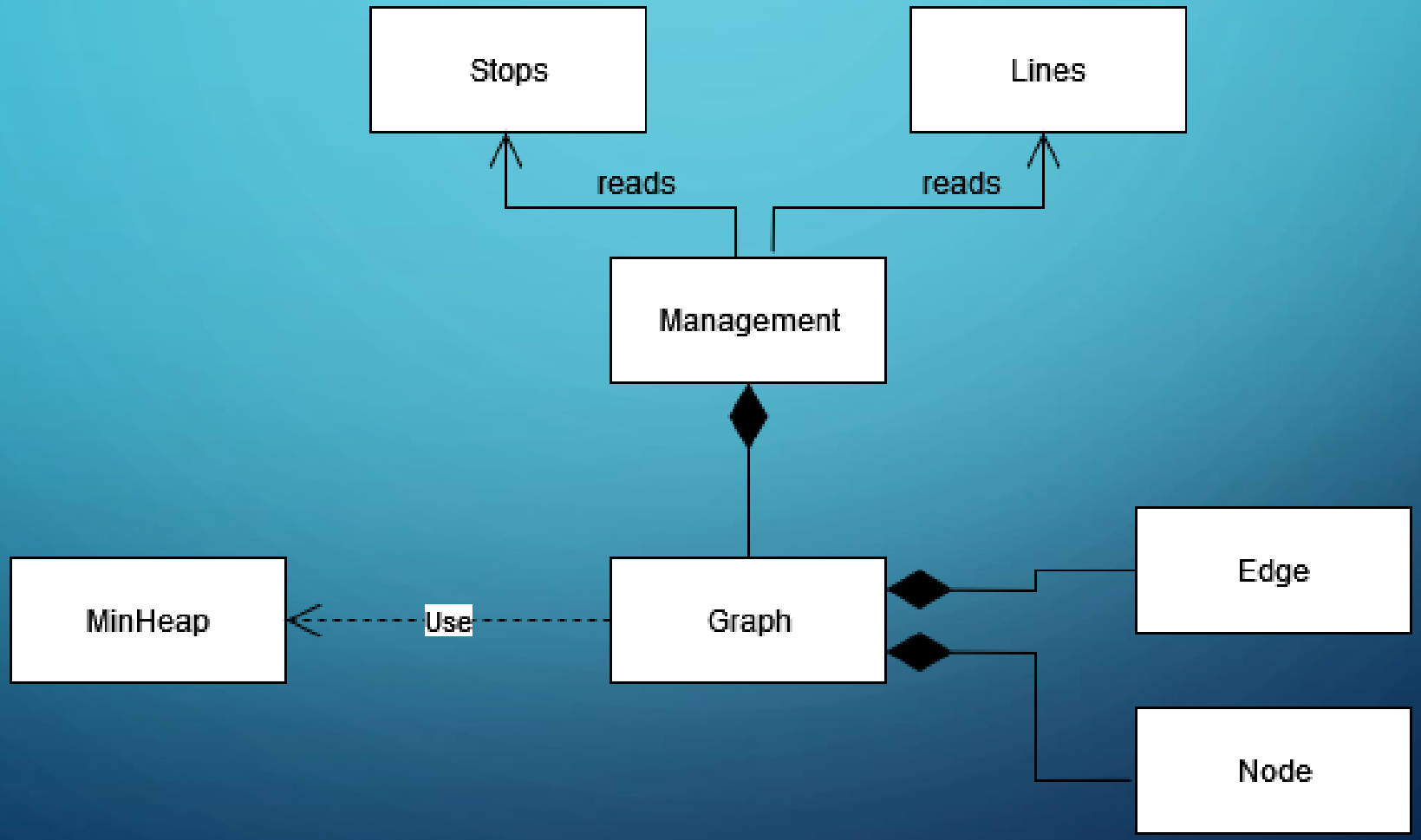
ALGORITMOS E ESTRUTURAS DE DADOS

SISTEMA PARA NAVEGAÇÃO NOS TRANSPORTES PÚBLICOS DO PORTO

GRUPO 12

Bernardo Campos – 202006056
José Sousa - 202006141

DIAGRAMA DE CLASSES



LEITURA DO DATASET

Leitura das Paragens

```
void Management :: readStops() {
    string code, name, zone, temp, lat, lon;
    int nodeNum = 1;

    fstream f;

    f.open( s: "stops.csv", ios::in);

    getline( &: f, &: temp);

    while (getline( &: f, &: temp)) {
        stringstream aux(temp);
        getline( &: aux, &: code, delim: ',');
        getline( &: aux, &: name, delim: ',');
        getline( &: aux, &: zone, delim: ',');
        getline( &: aux, &: lat, delim: ',');
        getline( &: aux, &: lon, delim: ',');

        Stops stop( c: code, n: name, z: zone, lt: stof(lat), ln: stof(lon));
        pair<string, int> p(code, nodeNum);
        mapStop.insert(p);
        stops.push_back(stop);
        nodeNum++;
    }
}
```

Leitura das Linhas

```
void Management::readLines() {
    string code, name, temp;
    fstream f;

    f.open( s: "lines.csv", ios::in);
    getline( &: f, &: temp);

    while (getline( &: f, &: temp)) {
        stringstream aux(temp);
        getline( &: aux, &: code, delim: ',');
        getline( &: aux, &: name, delim: ',');


        Lines line( c: code, n: name);
        lines.push_back(line);
    }
}
```

Leitura das Paragens por Linha

```
void Management::addLineToGraph(Lines line) {
    int dir, StopNumber, currentStopCode, nextStopCode;
    float weight;
    string currentStop, nextStop, LineFile;
    fstream f;

    //linhas circulares
    if(line.getCode() =="300" || line.getCode() =="301" || line.getCode() =="302" || line.getCode() =="303"){

        dir=0;
        LineFile="line_"+line.getCode()+"_"+ to_string(dir)+".csv";

        f.open(LineFile);

        f>>StopNumber;
        f>>currentStop;

        while(f>>nextStop){
            currentStopCode=mapStop.at(currentStop);
            nextStopCode=mapStop.at(nextStop);

            weight=haversine( lat1: stopFinder( n: currentStop).getLatitude(), lon1: stopFinder( n: currentStop).getLongitude(),
             lat2: stopFinder( n: nextStop).getLatitude(), lon2: stopFinder( n: nextStop).getLongitude());

            dayGraphz.addEdge(currentStopCode, nextStopCode, weight);

            currentStop=nextStop;
        }
    }
```

GRAFOS USADOS

- Os grafos implementados neste trabalho usam a classe Graph usada nas aulas.
- Foram criados dois grafos a partir da Dataset: um grafo para as linhas da rede regular e outro para as linhas da rede noturna, com o sufixo “M”.


```
class Graph {
    struct Edge {
        int dest;   // Destination node
        int weight; // An integer weight
    };

    struct Node {
        list<Edge> adj; // The list of outgoing edges (to adjacent nodes)
        bool visited;// As the node been visited on a search?
        float distance;
        int predecessor;
    };

    int n;              // Graph size (vertices are numbered from 1 to n)
    bool hasDir;        // false: undirect; true: directed
    vector<Node> nodes; // The list of nodes being represented
```

FUNCIONALIDADES IMPLEMENTADAS

- O utilizador deve indicar o código das paragens de origem e de destino, e se deseja deslocar-se durante o dia ou a noite (já que existem o grafo para as linhas regulares e outro para as linhas noturnas “M”).

FUNCIONALIDADES IMPLEMENTADAS (CONTINUAÇÃO)

- É dada ao utilizador a escolha para o tipo de percurso que deseja: menor número de paragens ou o que percorre menor distância.
- Para obter o percurso com menor número de paragens foi usado um algoritmo de pesquisa em largura (BFS).
- Para obter o percurso que percorre menor distância foi usado o algoritmo de Dijkstra e a fórmula de Haversine para calcular a distância entre cada paragem.

FUNCIONALIDADES IMPLEMENTADAS (CONTINUAÇÃO)

- O programa indica a sequência de paragens que o utilizador deve seguir da origem até ao destino.
- Assim o transbordo de autocarros é dado como implícito, não sendo considerada a possibilidade de um utilizador se deslocar a pé de uma paragem para outra.

INTERFACE COM O UTILIZADOR

- Começa por perguntar o período da viagem para operar a partir do grafo apropriado.
- Pergunta o tipo de itinerário desejado.
- Pede os códigos das paragens de origem e destino, e indica os códigos da sequência de paragens a seguir.

```
Welcome to our bus navigation system. Please select the period of travel:
A) Day
B) Night
A
Insert the desired type of recommended itinerary:
A) Fewest stops
B) Shortest itinerary
A
Insert the code for the bus stop of origin:
IPO2
Insert the code for the bus stop of destiny:
BVLH1
The sequence of stops to follow are: IPO2 CSJ4 ISEP4 DSS IPRN2 LGCL2 CPL VFM2 SVP1 BVLH1.
```

EXEMPLO DE EXECUÇÃO DO PROGRAMA

DESTAQUE DE FUNCIONALIDADE

- Por, como é mencionado a seguir, ter sido algo que foi muito dispendioso em termos de tempo, a função responsável por adicionar os dados nos grafos merece algum destaque.
-

```
void Management::addLineToGraph(Lines line) {
    int dir, StopNumber, currentStopCode, nextStopCode;
    float weight;
    string currentStop, nextStop, LineFile;
    fstream f;

    //linhas circulares
    if(line.getCode() =="300" || line.getCode() =="301" || line.getCode() =="302" || line.getCode() =="303"){

        dir=0;
        LineFile="line_"+line.getCode()+"_"+ to_string(dir)+".csv";

        f.open(LineFile);

        f>>StopNumber;
        f>>currentStop;

        while(f>>nextStop){
            currentStopCode=mapStop.at(currentStop);
            nextStopCode=mapStop.at(nextStop);

            weight=haversine( lat1: stopFinder( n: currentStop).getLatitude(), lon1: stopFinder( n: currentStop).getLongitude(),
             lat2: stopFinder( n: nextStop).getLatitude(), lon2: stopFinder( n: nextStop).getLongitude());

            dayGraphz.addEdge(currentStopCode, nextStopCode, weight);

            currentStop=nextStop;
        }
    }
```

PRINCIPAIS DIFICULDADES

- Gestão de tempo para a realização do projeto: a atual carga de trabalhos e exames de outras Unidades Curriculares dificultou o progresso do trabalho.
- Inserção dos dados fornecidos em grafos foi uma das partes mais trabalhosas e dispendiosas em termos de tempo.

- As tarefas foram igualmente distribuídas por nós, e cada um cumpriu com aquilo que lhe foi designado