CSE222

HW8 REPORT

OĞULCAN

KALAFATOĞLU

1801042613

2 – Problem solution approach

For part1, I added new edge constructors to edge class that takes, float, integer and a new constructor that adds all of them as weight, and holds them in their respective fields.

```
public Edge(int source, int dest, double w, int w2, float w3){
    this.source = source;
    this.dest = dest;
    this.weight = w;
    this.weightI = w2;
    this.weightF = w3;
}
```

```
public static void dijkstrasAlgorithm(Graph graph, int start, int[] pred, double[] dist, type t, int op) {
```

```
/**
 * enum for holding weight types.
 */
public enum type {
    INTEGER,
    DOUBLE,
    FLOAT
}
```

part2: for bfs and dfs, while traversing if first start vertex is iterable then i incremented my variable.

bfs

Holding another visited array for whole traversal.

```
if(visited[current] != 1){
    ++connectedNum;
    visited[current] = 1;
```

dfs

```
}
for (int i = 0; i < n; i++) {
    if (!visited[i]){
        Iterator<Edge> it = graph.edgeIterator(i);
        if(it.hasNext()) {
            ++connected;
        }
        depthFirstSearch(i);
    }
```

3- test cases

part1

```
lg.insert(new Edge(0, 1,1.0,12,2.132f));
lg.insert(new Edge(1,3,1.0,6,3.4520f));
lg.insert(new Edge(2,5,1.0,4,8.1230f));
lg.insert(new Edge(3,4,1.0,3,2.023f));
lg.insert(new Edge(4,5,1.0,1,3.343f));
```

```
public static void testAlgorithm2(Graph lg, type t){
    System.out.println("Calling method with op +");
    testAlgorithm(lg, t, 0);
    System.out.println("Calling method with op multiplication");
    testAlgorithm(lg, t, 1);
    System.out.println("Calling method with  op = *");
    testAlgorithm(lg, t, 2);
}
```

part2

```
public static void test(){
    ListGraph graph = new ListGraph(50, false);
    int size = 50;
    graph.insert(new Edge(0, 6));
    graph.insert(new Edge(9, 10));
    graph.insert(new Edge(6, 2));

    System.out.println("Connected component outputs for graph of 0->6->2 9->10");
    System.out.println("dfs is called : " + graph.dfs());
    System.out.println("bfs is called : " + graph.bfs_connected(50));

    graph = null;
    graph = new ListGraph(100, false);

    graph.insert(new Edge(0, 34));
    graph.insert(new Edge(54, 65));
    graph.insert(new Edge(65, 75));
    graph.insert(new Edge(75, 33));
    graph.insert(new Edge(88, 22));

    System.out.println("Connected compnonent outputs for graph of 0->34, 54->65->75->33, 88->22");
    System.out.println("dfs is called : " + graph.dfs());
    System.out.println("bfs is called : " + graph.bfs_connected(50));

}
```

part1 test case results:

```
Testing double typed weight
Calling method with op +
pred -> 0 dist -> 0.0
pred -> 0 dist -> 1.0
pred -> 5 dist -> 5.0
pred -> 1 dist -> 2.0
pred -> 3 dist -> 3.0
Calling method with op multiplication
pred -> 0 dist -> 0.0
pred -> 0 dist -> 1.0
pred -> 5 dist -> 1.0
pred -> 1 dist -> 1.0
pred -> 3 dist -> 1.0
Calling method with  op = *
pred -> 0 dist -> 0.0
pred -> 0 dist -> 1.0
pred -> 5 dist -> 1.0
pred -> 1 dist -> 1.0
pred -> 3 dist -> 1.0
```

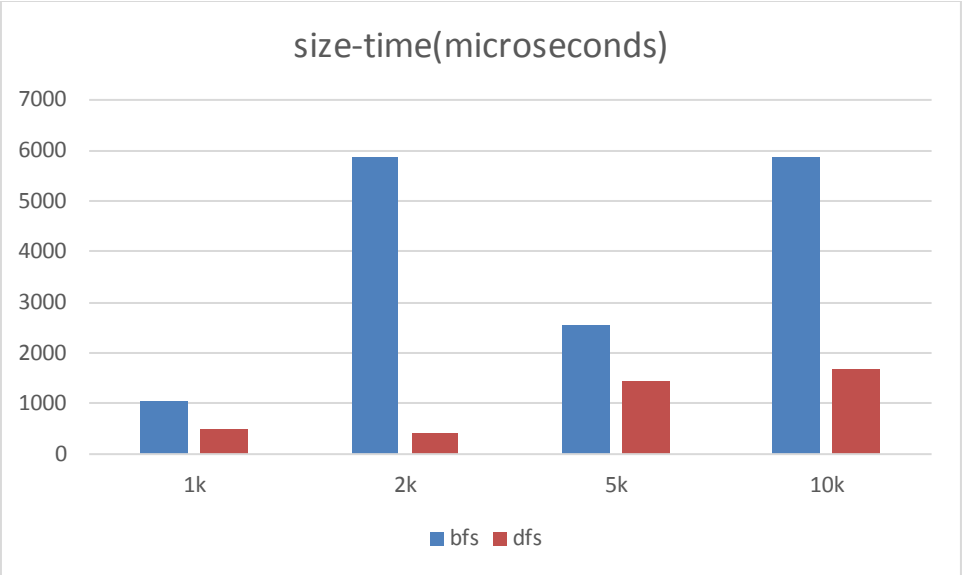
```
Testing int typed weight
Calling method with op +
pred -> 0 dist -> 0.0
pred -> 0 dist -> 12.0
pred -> 5 dist -> 23.0
pred -> 1 dist -> 18.0
pred -> 3 dist -> 21.0
Calling method with op multiplication
pred -> 0 dist -> 0.0
pred -> 0 dist -> 12.0
pred -> 5 dist -> 216.0
pred -> 1 dist -> 72.0
pred -> 3 dist -> 216.0
Calling method with  op = *
pred -> 0 dist -> 0.0
pred -> 0 dist -> 12.0
pred -> 5 dist -> 1.0
pred -> 1 dist -> -54.0
pred -> 3 dist -> 111.0
```

```
Testing float typed weight
Calling method with op +
pred -> 0 dist -> 0.0
pred -> 0 dist -> 2.131999969482422
pred -> 5 dist -> 11.949999809265137
pred -> 1 dist -> 5.583999872207642
pred -> 3 dist -> 7.60699987411499
Calling method with op multiplication
pred -> 0 dist -> 0.0
pred -> 0 dist -> 2.131999969482422
pred -> 5 dist -> 49.77258767569738
pred -> 1 dist -> 7.359663687263492
pred -> 3 dist -> 14.888599653371488
Calling method with  op = *
pred -> 0 dist -> 0.0
pred -> 0 dist -> 2.131999969482422
pred -> 5 dist -> 1.0
pred -> 1 dist -> -1.7756638150558501
pred -> 3 dist -> 3.839504088096293
```

part2 test case results:

```
dfs is called : 2
6
0
2
6
10
9
bfs is called : 2
Connected compnonent outputs for graph of 0->34, 54->65->75->33, 88->22
dfs is called : 3
34
0
88
22
75
65
33
54
75
65
bfs is called : 3
```

3 – Running results:




```
Testing Graph methods 5 times with size of 1000
connected component by bfs : 3, connected component by dfs : 3
connected component by bfs : 4, connected component by dfs : 4
connected component by bfs : 4, connected component by dfs : 4
connected component by bfs : 8, connected component by dfs : 8
connected component by bfs : 6, connected component by dfs : 6

Avg BFS time for max num of 266 edge is 4302 microseconds
Avg DFS time for max num of 266 edge is 1053 microseconds

connected component by bfs : 12, connected component by dfs : 12
connected component by bfs : 11, connected component by dfs : 11
connected component by bfs : 11, connected component by dfs : 11
connected component by bfs : 15, connected component by dfs : 15
connected component by bfs : 11, connected component by dfs : 11

Avg BFS time for max num of 433 edge is 2751 microseconds
Avg DFS time for max num of 433 edge is 891 microseconds

connected component by bfs : 20, connected component by dfs : 20
connected component by bfs : 15, connected component by dfs : 15
connected component by bfs : 12, connected component by dfs : 12
connected component by bfs : 13, connected component by dfs : 13
connected component by bfs : 19, connected component by dfs : 19

Avg BFS time for max num of 600 edge is 1050 microseconds
Avg DFS time for max num of 600 edge is 486 microseconds

--------End of this size--------
```

```
Testing Graph methods 5 times with size of 2000
connected component by bfs : 17, connected component by dfs : 17
connected component by bfs : 13, connected component by dfs : 13
connected component by bfs : 19, connected component by dfs : 19
connected component by bfs : 14, connected component by dfs : 14
connected component by bfs : 15, connected component by dfs : 15

Avg BFS time for max num of 533 edge is 1408 microseconds
Avg DFS time for max num of 533 edge is 773 microseconds

connected component by bfs : 21, connected component by dfs : 21
connected component by bfs : 22, connected component by dfs : 22
connected component by bfs : 20, connected component by dfs : 20
connected component by bfs : 25, connected component by dfs : 25
connected component by bfs : 27, connected component by dfs : 27

Avg BFS time for max num of 866 edge is 5824 microseconds
Avg DFS time for max num of 866 edge is 1666 microseconds

connected component by bfs : 33, connected component by dfs : 33
connected component by bfs : 26, connected component by dfs : 26
connected component by bfs : 34, connected component by dfs : 34
connected component by bfs : 32, connected component by dfs : 32
connected component by bfs : 26, connected component by dfs : 26

Avg BFS time for max num of 1200 edge is 5887 microseconds
Avg DFS time for max num of 1200 edge is 416 microseconds

--------End of this size--------
```

```
Testing Graph methods 5 times with size of 5000
connected component by bfs : 32, connected component by dfs : 32
connected component by bfs : 29, connected component by dfs : 29
connected component by bfs : 43, connected component by dfs : 43
connected component by bfs : 43, connected component by dfs : 43
connected component by bfs : 37, connected component by dfs : 37

Avg BFS time for max num of 1333 edge is 9488 microseconds
Avg DFS time for max num of 1333 edge is 860 microseconds

connected component by bfs : 64, connected component by dfs : 64
connected component by bfs : 64, connected component by dfs : 64
connected component by bfs : 64, connected component by dfs : 64
connected component by bfs : 54, connected component by dfs : 54
connected component by bfs : 42, connected component by dfs : 42

Avg BFS time for max num of 2166 edge is 4191 microseconds
Avg DFS time for max num of 2166 edge is 1232 microseconds

connected component by bfs : 62, connected component by dfs : 62
connected component by bfs : 99, connected component by dfs : 99
connected component by bfs : 81, connected component by dfs : 81
connected component by bfs : 67, connected component by dfs : 67
connected component by bfs : 69, connected component by dfs : 69

Avg BFS time for max num of 3000 edge is 2536 microseconds
Avg DFS time for max num of 3000 edge is 1446 microseconds

--------End of this size--------
```

```
Testing Graph methods 5 times with size of 10000
connected component by bfs : 102, connected component by dfs : 102
connected component by bfs : 58, connected component by dfs : 58
connected component by bfs : 89, connected component by dfs : 89
connected component by bfs : 62, connected component by dfs : 62
connected component by bfs : 74, connected component by dfs : 74

Avg BFS time for max num of 2666 edge is 4899 microseconds
Avg DFS time for max num of 2666 edge is 1185 microseconds

connected component by bfs : 121, connected component by dfs : 121
connected component by bfs : 132, connected component by dfs : 132
connected component by bfs : 111, connected component by dfs : 111
connected component by bfs : 106, connected component by dfs : 106
connected component by bfs : 120, connected component by dfs : 120

Avg BFS time for max num of 4333 edge is 5863 microseconds
Avg DFS time for max num of 4333 edge is 1697 microseconds

connected component by bfs : 134, connected component by dfs : 134
connected component by bfs : 145, connected component by dfs : 145
connected component by bfs : 149, connected component by dfs : 149
connected component by bfs : 152, connected component by dfs : 152
connected component by bfs : 145, connected component by dfs : 145

Avg BFS time for max num of 6000 edge is 19456 microseconds
Avg DFS time for max num of 6000 edge is 2190 microseconds

--------End of this size--------
```