MTH6412B Project - Report of Phase 2

Authors:

- **Paul A. Patience**
- **Luis Rojo-González**

Description:

You may find the scripts used in this notebook in the following **GitHub repository**.

Procedure

We have created the file *kruskal.jl* which implement the algorithm described in the book **Introduction to algorithms (3eds.)** in pp. 631. Furthermore, the file **main.jl** is made such that the graph is constructed and then applies the function **kruskal** to it and assign the results into an object.

The function returns the edges within the minimum spanning tree and the distance product of the summation over its edges. This procedure is shown for the instance **Test_1.tsp** which corresponds to the graph seen in class.

Thus, to get the results we create the graph via **test_1 = build_graph("Test_1.tsp)** and then the minimum spanning tree through **test_1_mst = kruskal(test_1)**. Here, the attributes are retrieved via **test_1["MST"]** for the componentes of the tree and **test_1["Distance"]** for the total distance in the tree.

```
begin
    test_1 = build_graph("Test_1.tsp")
    test_1_mst = kruskal(test_1)
end;
```

Furthermore, a test set is included since the response is known for this instance. In this regard, we know that the resulting minimum spanning tree must have eight edges and has a cost (or distance) equals to 37.

```
true
```

```
length(test_1_mst["MST"]) == length(test_1["Nodes"]) - 1
```

```
true
```

```
test_1_mst["Distance"] == 37
```

As we can see, both conditions are satisfied. Therefore, we may claim that the algorithm implemented actually generate a minimum spanning tree.

Then, we can construct the minimum spanning tree for another instance whose solution is not know. In this case, we test this with a full matrix instance. In particular, we show the results for the **bays29.tsp** instance.

```
begin
    bays29 = build_graph("bays29.tsp")
    bays29_mst = kruskal(bays29)
end;
```

We can see here the edges that are part of the minimum spanning tree,

```
Any[
    1:   (10, 20)
    2:   (14, 18)
    3:   (4, 15)
    4:   (26, 29)
    5:   (24, 27)
    6:   (2, 21)
    7:   (4, 10)
    8:   (8, 27)
    9:   (14, 22)
    10:   (1, 28)
    11:   (5, 9)
    12:   (6, 12)
    13:   (16, 27)
    14:   (15, 18)
    15:   (15, 19)
    16:   (5, 26)
    17:   (10, 13)
    18:   (14, 17)
    19:   (6, 28)
    20:   (5, 6)
    21:   (1, 21)
    22:   (16, 19)
    23:   (1, 24)
    24:   (19, 25)
    25:   (3, 29)
    26:   (11, 15)
    27:   (23, 27)
    28:   (7, 25)
]
```

and its distance or cost is

```
1557
```