Samuel Chewarria
4110 21334
Dijkestra's Algorithm:

1.) let n be the number of adjacent vertices, then sequentially searching for the minimum bridged edge would result in a complexity $O(n)$.

2.1) Priority queue using doubly-linked list:
Given that the priority queue would always have the smallest edge as the first element, accessing it would be $O(1)$. On the otherhand, Insertion would be of complexity $O(n)$.

2.2) Priority queue by min-heap:
Maintaining the minimum bridged edge using min heap would always result to the root having the min edge with $O(1)$. However, removing the smallest edge, would result to readjusting the min-heap, wich is of complexity $O(\log(n))$ where n is the number of edges.
Similarly, for the isertion operation, the time complexity would be $O(\log(n))$, involves readjusting the heap.

3. Implementing prims algorithm:

a. let MST_s be the set of vertex included in the sortest path (minimum dist from source is calculated and finalized.

b. let all ~~wight~~ from ~~I~~ source be INFINITE.
let distance value from source be 0 to be picked first.
and initialize all vertices in the graph.

c. While the MST_s does not include all vertices:
-> Pick the minimum ~~edge~~ bridged edge from unselected set
-> ~~ad~~ add it to MST_s
-> update the distance value of all adjecent vertices of ~~it sete~~ the newly selected vertex.

* for every adjacent vertex, if sum of distance value of newly selected vertex (u) and weight of edge u-v is les than v then update the distance value v.

4.) when the unvisited vertices have a distance Infinity from the nodes in the visited vertices set, then the algorithm ends and returns unreachable.