Algorithm Homework 4C

학번 / 이름: 2014310407 / 이 준 혁

1. 구현

0. 우선 두 노드의 위치를 바꾸는 swap position 함수를 만들었다.

1. Min_heapify : 우선 왼쪽 자식이 힙의 크기보다 큰 경우에는 무시했고, 왼쪽 자식이 힙의 크기와 같은 경우에는 왼쪽 자식이 부모보다 작은 경우에만 두 노드와 위치를 바꿨다.

만약 두 자식 노드가 다 있다면, 두 노드중 작은 노드를 구해서 그 작은 자식 노드가 부모 노드보다 키 값이 작다면, 위치와 노드를 바꾼 후, 바뀐 자식에 대해 Min_heapify 함수를 재귀적으로 호출했다.

2. Extract_min: 우선 힙의 가장 첫 번째 원소를 저장한 후, 마지막 원소와 위치와 노드를 바꾼 후에, 남은 노드의 개수를 1 줄였다. 이후 저장한 원소를 반환했다

3. decrease_key: 만약 노드의 키 값이 입력받은 키 값보다 크다면, 입력받은 키 값으로 노드의 키 값을 바꿨다.

4. addvertextoHeap: 힙의 가장 마지막 위치에 그래프의 노드 포인터를 삽입했다. 그리고 해당되는 노드 위치에 몇 번째 노드인지를 기록한 후, 남은 노드의 개수를 1 증가시켰다.

5. primAlgorithm:

우선 그래프 노드의 개수만큼의 저장 공간을 가지는 heap 을 만들었다. 이후 힙의 모든 위치는 초기화 되어있지 않으므로 무한대로 초기화했다. 이후 그래프의 각 노드를 힙에다 넣었다. 그리고 첫 노드의 키 값은 0 으로 초기화했다.

이후 힙이 빌 때 까지, 최솟값을 가지는 노드를 Extract_min 함수를 이용해 뽑아냈다. 그리고 그 노드와 연결되어 있는 모든 노드에 대해, decrease_key 함수를 통해 연결된 모든 노드의 키 값을 연결 가중치로 감소시켰다.

이 과정을 수행하면, 최댓값을 뽑아낸 후부터 heap 이 아니므로, 맨 처음 원소를 최솟값으로 만들기 위해 min_heapify 를 진행했다. 이 때, 매 과정마다 heap 을 만들어야 하기 때문에, 가장 마지막 원소의 부모부터 하나씩 줄여 나가며 min_heapify 를 진행했다. 즉, heap 을 build 했다.

이 과정을 heap 이 빌 때까지 반복하면, 모든 노드에 들어있는 키 값은 해당 노드에 연결된 mst 의 edge 의 가중치 값과 같으므로, 그 값을 모두 더해서 반환했다. 그 결과, mst 의 모든 edge 값의 합을 반환할 수 있었다.