REPORT

숭실대학교

과목명	알고리즘
과제번호	06
담당교수	최재영
학과	컴퓨터학부
학년	2학년
학번	20201852
이름	변서윤
출석번호	314

1번) Prim's Algorithm (v1부터 시작)

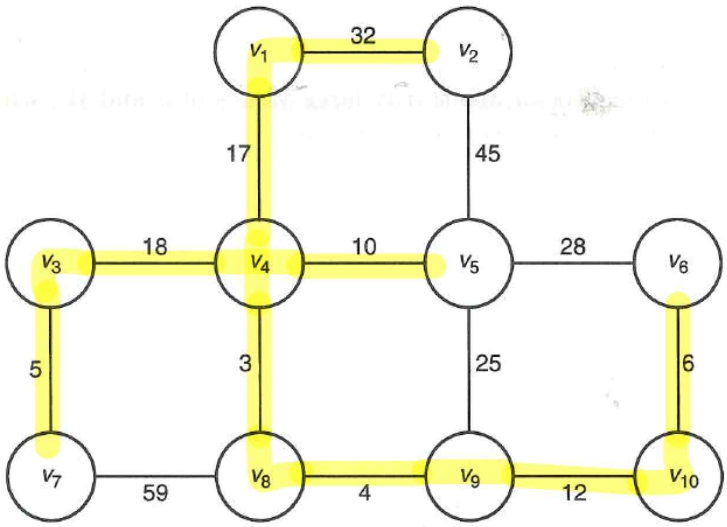
(1) 설계

- **prim 함수**: 시작정점에서 가중치가 가장 낮은 정점 선택->새롭게 연결된 정점과 비교해서 거리가 더 작은 가중치로 distance 가중치 배열 갱신->연결된 정점 집합에서 가중치가 가장 낮은 정점 선택 => 경로가 없을 때까지 반복
- **getMinVertex 함수**: distance 배열을 통해 가중치가 가장 작은 정점 찾는 함수
- **main**: 시작 정점을 prim함수에 넘겨준다.

(2) 실행 결과

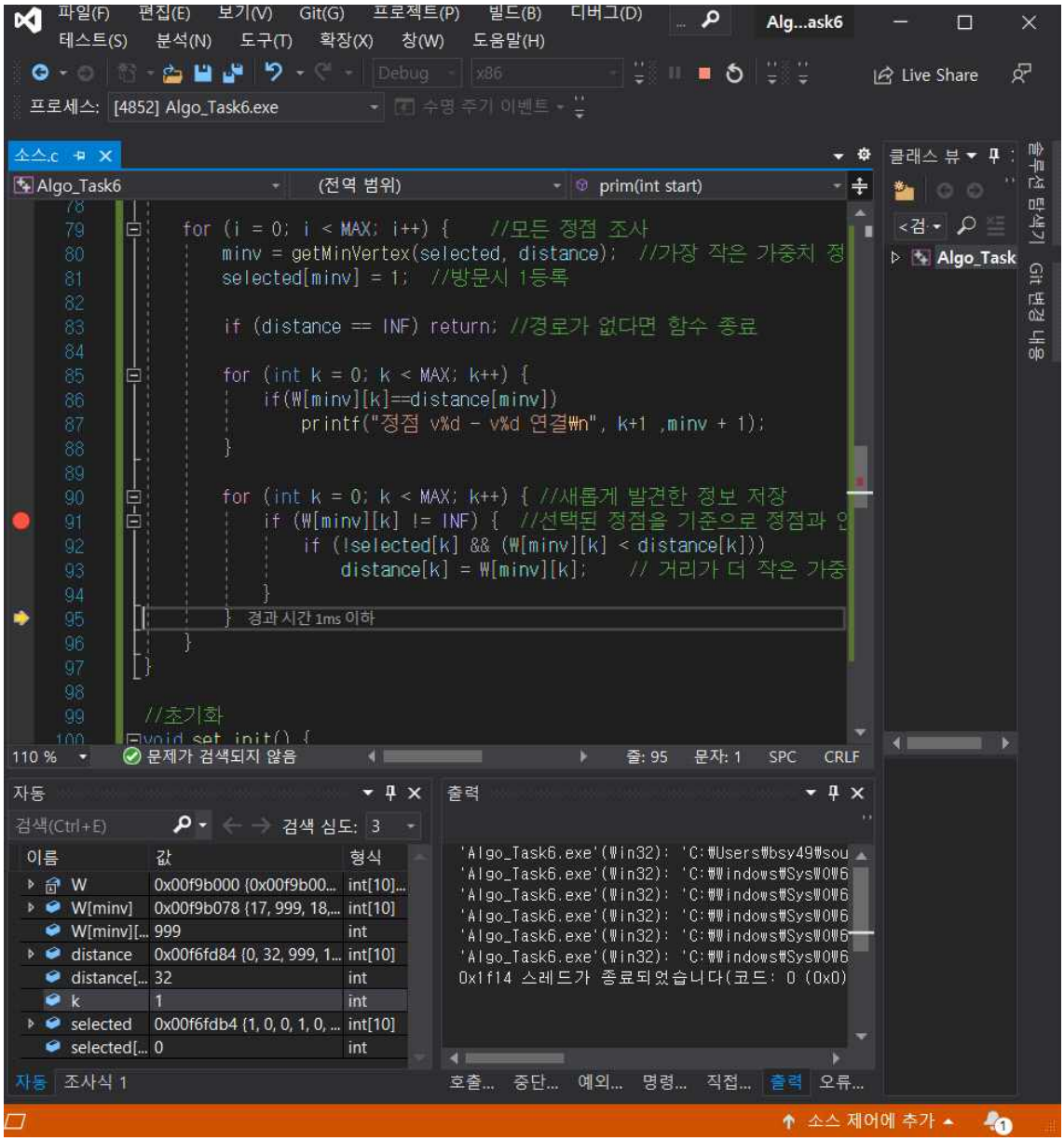
```
<-------v1부터 시작------->
정점 v1 - v1 연결
정점 v1 - v4 연결
정점 v4 - v8 연결
정점 v8 - v9 연결
정점 v4 - v5 연결
정점 v9 - v10 연결
정점 v10 - v6 연결
정점 v4 - v3 연결
정점 v3 - v7 연결
정점 v1 - v2 연결
```

(3) 연결된 간선들(노랑)



(4) 디버그 사용

- 거리가 더 작은 가중치로 갱신할 때 디버그를 사용하였다.



2번) Prim's Algorithm (v8부터 시작)

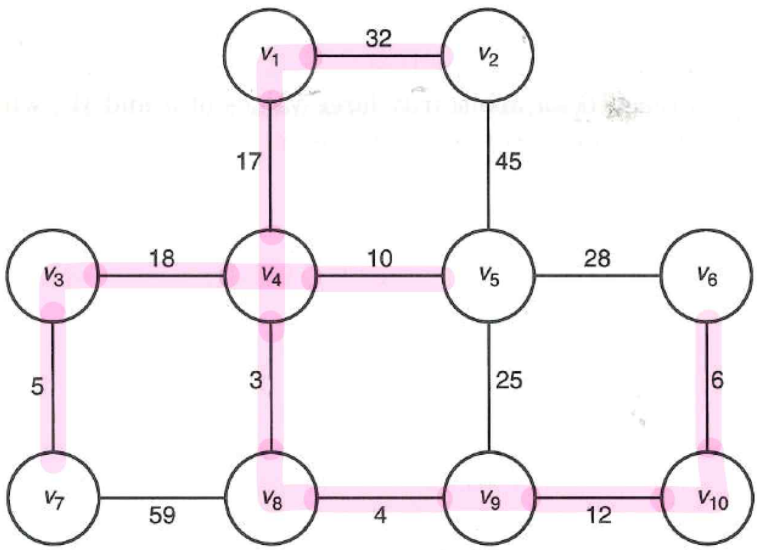
(1) 설계

- 1번 문제의 코드와 동일

(2) 실행 결과

```
<-------v8부터 시작------->
정점 v8 - v8 연결
정점 v8 - v4 연결
정점 v8 - v9 연결
정점 v4 - v5 연결
정점 v9 - v10 연결
정점 v10 - v6 연결
정점 v4 - v1 연결
정점 v4 - v3 연결
정점 v3 - v7 연결
정점 v1 - v2 연결
```

(3) 연결된 간선들(분홍)

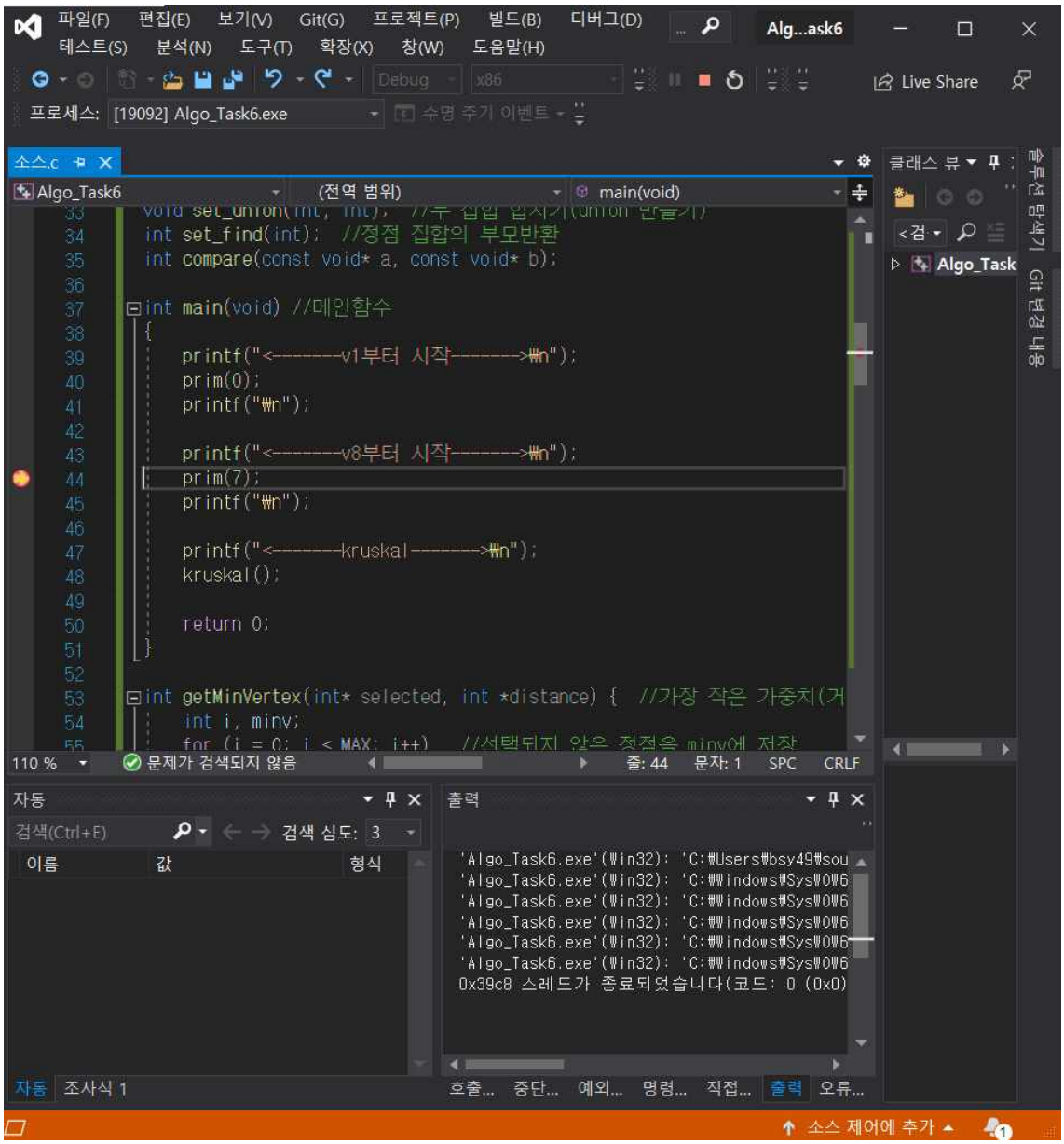


(4) 비교

문제1의 Prim's 알고리즘은 시작점이 v1이고, 문제 2의 시작점은 v8으로 두 문제의 시작점은 달랐지만 연결된 간선의 모습을 보면 최종적인 MST는 똑같다는 것을 알 수 있다.

(4) 디버그 사용

- 시작점을 다르게 넣는 부분에 디버그를 사용하였다.



3번) Kruskal's Algorithm

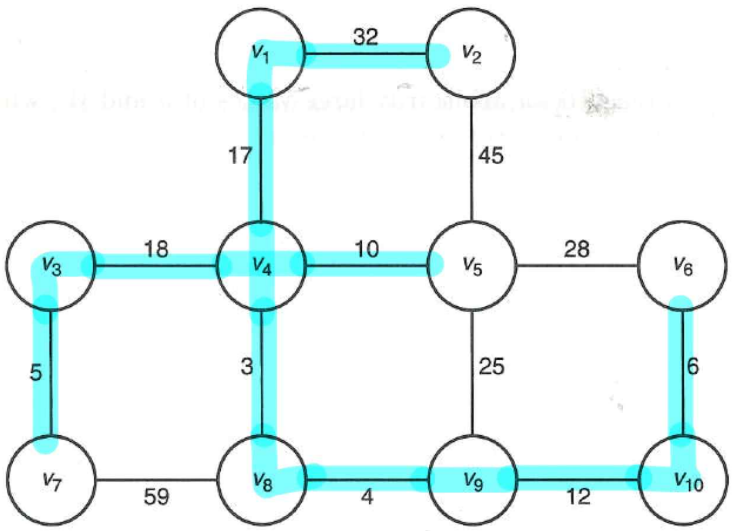
(1) 설계

- **kruskal 함수**: edge_set함수로 간선들의 집합을 만들고-> 오름차순 정렬-> set_find를 이용해 사이클이 생기는지 확인 후 -> 사이클이 생기지 않는다면 set_union 두 정점의 집합을 합쳐준다.
- **union-find 사용=> set_init 함수**: parent 배열 초기화 / **set_union 함수**: 두 정점의 집합 합치기 / **set_find 함수**: 해당 정점의 집합의 부모 반환
- **compare 함수**: qsort를 사용하기 위해 가중치의 차이 반환하는 함수
- **edge_set 함수**: 간선들의 집합을 만들어 주는 함수

(2) 실행 결과

```
<-------kruskal------->
정점 v4 - v8 연결
정점 v8 - v9 연결
정점 v3 - v7 연결
정점 v6 - v10 연결
정점 v4 - v5 연결
정점 v9 - v10 연결
정점 v1 - v4 연결
정점 v3 - v4 연결
정점 v1 - v2 연결
```

(3) 연결된 간선들(파랑)



(4) 디버그 사용

- 사이클을 확인하는 부분에서 생각한 대로 실행이 되는지 확인하기 위해 디버그를 사용했다.