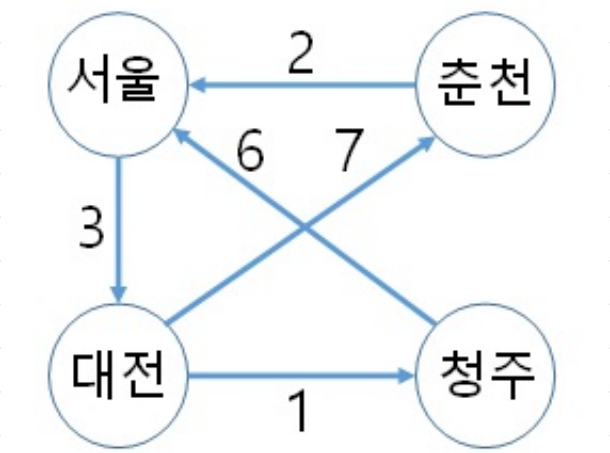
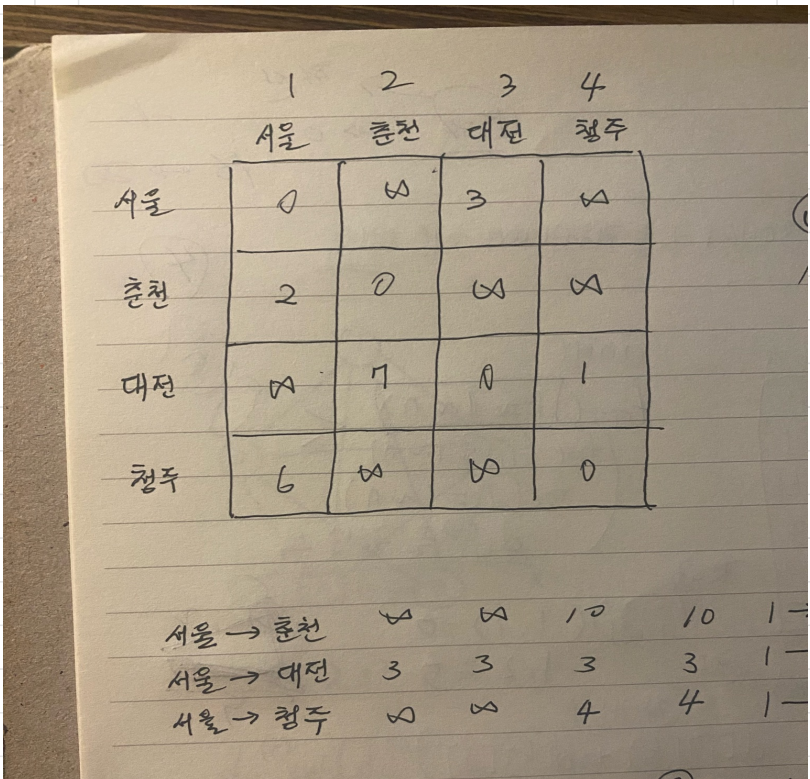
과제 6B

60211579

서호준

1. 강의노트에 있는 모든 쌍 최단 경로 알고리즘을 이용하여 다음 그래프에 대해 행렬 D를 구하라. 행렬 D에서 1행, 2행, 3행, 4행은 각각 서울, 춘천, 대전, 청주를 나타낸다.





2. 강의노트의 모든 쌍 최단경로 찾기 알고리즘은 모든 쌍의 최단 경로의 거리만을 계산한다. 최단 경로를 알 수 있도록 알고리즘을 수정하라.

0으로 초기화시켜 넣으면 경유하는 정점이 없다고 생각하자.

P : n행 n열의 배열

```
public static String[][] path(double W[][], int n){ //입력으로 최초 경로를 나타내는 행렬 W와 정점의 개수 n을 받는다.
    double D[][] = W; // 1. 거리 비교를 위한 최단 거리를 저장할 행렬 D를 선언하고 W를 대입한다.
    String P[][] = new String[n][n]; // 2. n * n 형태의 문자열 행렬 P를 선언한다.

    for(int i = 0; i < n; i++){ // 3. 지역 변수 i가 0에서 n - 1까지 증가할 동안
        for(int j = 0; j < n; j++) { // 3-1. j가 0에서 n - 1까지 증가할 동안
            if(i == j){ // 3-2. i와 j가 같으면 경유하는 지점이 없으므로 0을 넣는다.
                P[i][j] = "0";
            }
            else {
                P[i][j] = ""; // 3-3. 그게 아니라면 P 행렬에 시작점과 도착점을 추가한다. 시작점이 1이고 도착점이 4이면 14를 저장한다.
                P[i][j] += i + 1; // i는 0 ~ n - 1이므로 i + 1이 시작점
                P[i][j] += j + 1; // j는 0 ~ n - 1이므로 i + 1이 시작점
            }
        }
    }

    for(int i = 0; i < n; i++){ // 4. 지역 변수 i가 0에서 n - 1까지 증가할 동안
        for(int j = 0; j < n; j++){ // 4-1. 지역 변수 j가 0에서 n - 1까지 증가할 동안
            for(int k = 0; k < n; k++){ // 4-2. 지역 변수 k가 0에서 n - 1까지 증가할 동안
                if(D[i][k] + D[k][j] < D[i][j]){ // 4-3. D[i][k] + D[k][j] < D[i][j]라면
                    D[i][j] = D[i][k] + D[k][j]; // 4-4. D[i][j] = D[i][k] + D[k][j]
                    P[i][j] = P[i][k] + P[k][j].substring( beginIndex: 1); // 4-5. 그리고 P[i][j]에 대해 P[i][k]의 최단 경로와 P[k][j]경로를 합친다. P[k][j]의 출발점을 떼고 합치면 된다.
                }
            }
        }
    }
    return P; //5. 행렬 P를 반환한다.
}
}
```