알고리즘\_과제\_7

(Traveling Salesperson Problem)



학부:컴퓨터학부

학번:20162518

출석번호 : 156번

이름 : 최승서

**<소스코드>**

//

// main.c

// algo7\_TSP

//

// Created by James Choi on 2019/10/30.

// Copyright © 2019 James Choi. All rights reserved.

#include <stdio.h>

#define MAX 10000000

**int** W[5][5] = {

{0, 8, 13, 18, 20},

{3, 0, 7, 8, 10},

{4, 11, 0, 10, 7},

{6, 6, 7, 0, 11},

{10, 6, 2, 1, 0}};

**int** D[5][1<<5] ={MAX};

**int** minimum(**int** a, **int** b)

{

**return** a > b ? b : a;

}

**int** travel(**int** start, **int** visit)

{

**int** i=0;

**if** (visit == (1 << 5) - 1)

**return** W[start][0];

**int** min = 0;

**if** (min != 0)

**return** min;

min = MAX;

**for** (i = 0; i <= 5; i++)

{

**if** (visit&(1 << i))

**continue**;

**if** (W[start][i] == 0)

**continue**;

min = minimum(min, travel(i, visit | (1 << i)) + W[start][i]);

}

**return** min;

}

**int** main() {

**int** minlen = 0;

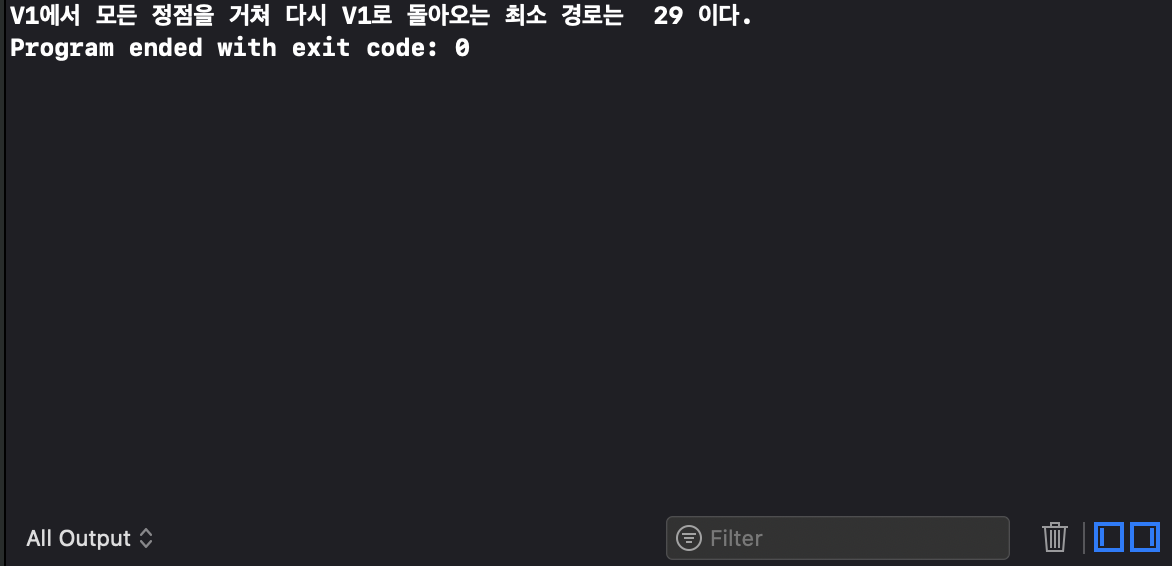
minlen = travel(0, 1);

printf("V1에서 모든 정점을 거쳐 다시 V1로 돌아오는 최소 경로는 %d 이다. \n", minlen);

**return** 0;

}

**<출력 화면>**



위 결과를 통해

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **W** | **V1** | **V2** | **V3** | **V4** | **V5** |
| **V1** | 0 | 8 | 13 | 18 | 20 |
| **V2** | 3 | 0 | 7 | 8 | 10 |
| **V3** | 4 | 11 | 0 | 10 | 7 |
| **V4** | 6 | 6 | 7 | 0 | 11 |
| **V5** | 10 | 6 | 2 | 1 | 0 |

이 인접행렬로 나타나는 가중치 그래프에서 V5까지 5개의 도시가 주어져 있고, 모든 도시들에 대한 가중치가 주어졌을 때,  V1 부터 시작하여 모든 도시를 단 1번씩 만 방문하여 다시 시작점으로 돌아오는 데 최단 거리를 구해보았다. 이 문제를 Dynamic Programing 기법으로 해결 할 수 있었던 이유는 앞서 풀었던 Obtimal binary search Tree나 chained martrix multiplication과 같이 Vk가 최단 경로 상에 V1 다음에 오는 첫번째 도시라고 했을때, Vk에서 V1로 가는 부분경로는 다른 도시를 각 1번씩 만 거치고, Vk에서 V1로 가는 최단경로여야 한다. 따라서 최적의 원칙이 성립하므로 Dynamic Programing 기법을 적용하여 해결 할 수 있었다.

이 알고리즘의 시간복잡도는 이 나온다. 따라서 만약 n이 16을 넘어가면 Dynamic Programing 기법을 사용함에도 실행시간이 너무 길어지므로 사용하기에 비효율적이다. 하지만 현재까지 지수보다 좋은 시간 복잡도를 가진 TSP 알고리즘을 찾아낸 사람은 아직 없다고 한다. 또 그것이 불가능하다고 증명한 사람도 없다. 따라서 이 알고리즘이 더 개선될 가능성이 아직 있다고 생각한다.