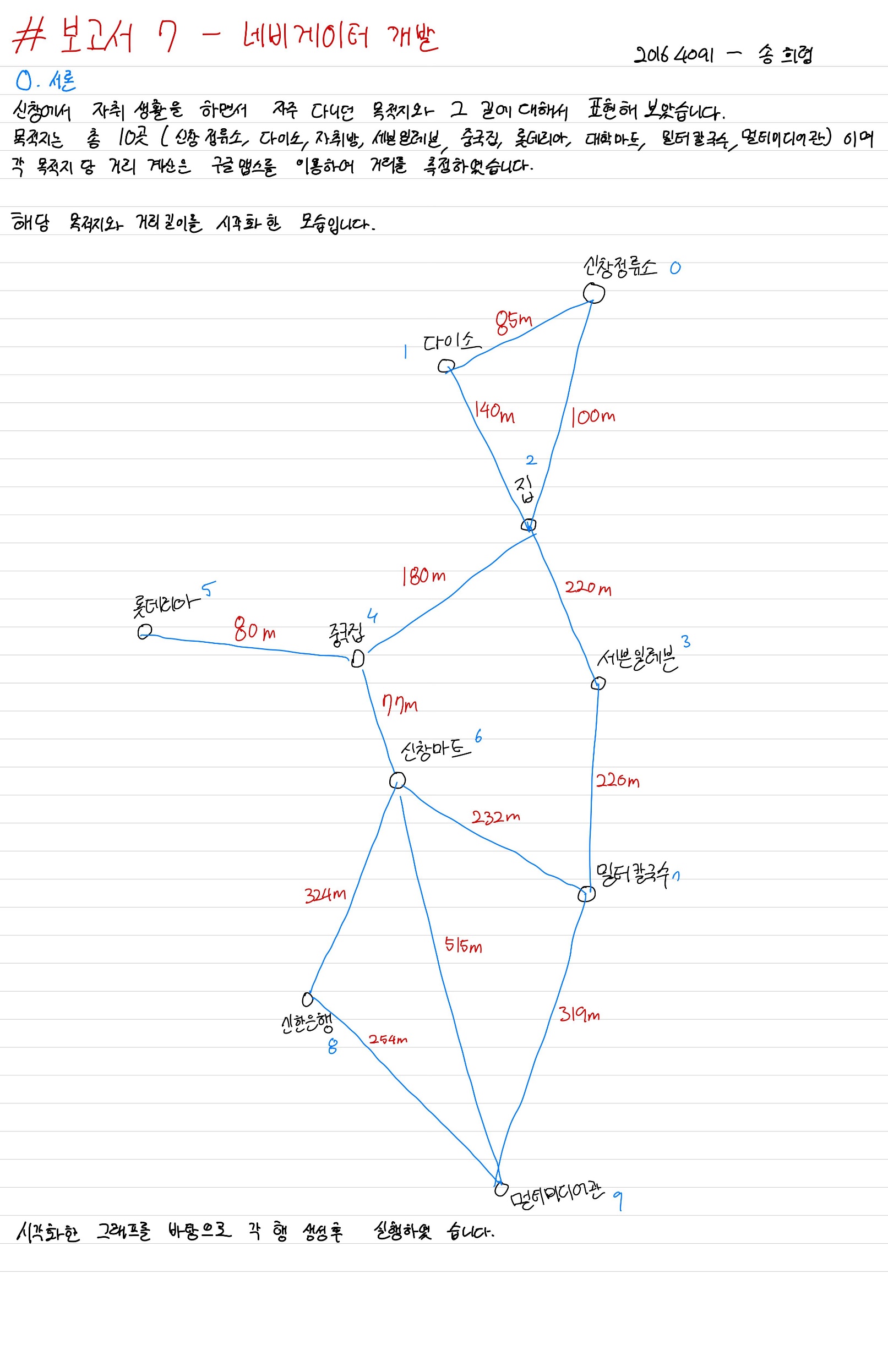
데이터구조 2

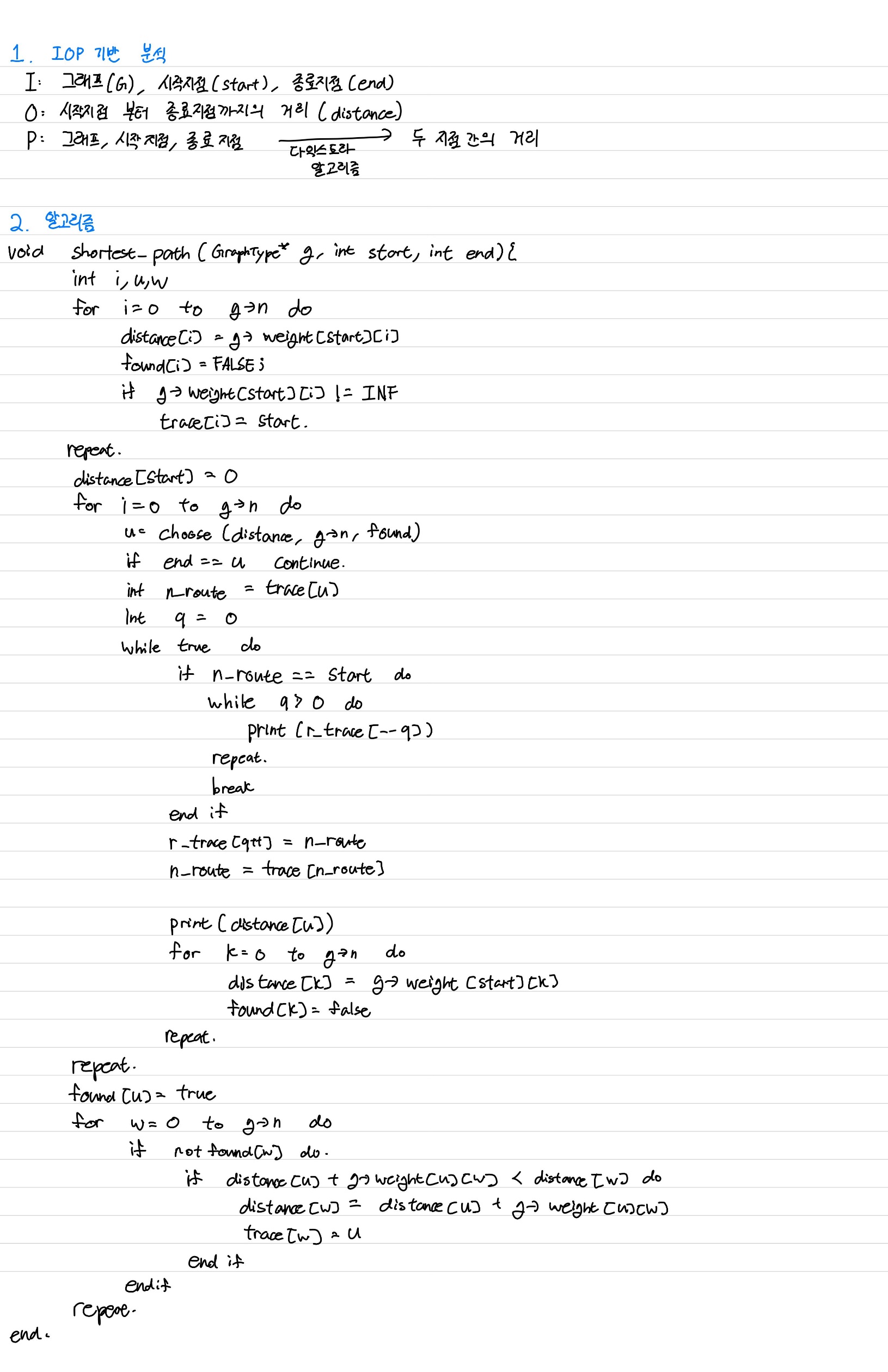
보고서 #6

(Dijkstra 알고리즘 네비게이터 개발)

20164091

송희령





3. 코드

#pragma warning(disable: 4996)

#include <limits.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define TRUE 1

#define FALSE 0

#define MAX\_VERTICES 100

#define INF 1000000 //무한대. (연결이 없는 경우)

프로그램을 작성하면서 필요한 헤더파일과 오류처리 및 사용 전역변수들을 정의해주었습니다.

void ret\_name(int i) {

    switch (i) {

    case 0:

        printf("신창정류소");

        break;

    case 1:

        printf("다이소");

        break;

    case 2:

        printf("자취방");

        break;

    case 3:

        printf("세븐일레븐");

        break;

    case 4:

        printf("중국집");

        break;

    case 5:

        printf("롯데리아");

        break;

    case 6:

        printf("대학마트");

        break;

    case 7:

        printf("밀터칼국수");

        break;

    case 8:

        printf("신한은행");

        break;

    case 9:

        printf("멀티미디어관");

        break;

    default :

        printf("Error!");

        break;

    }

}

각 목적지 코드에 맞는 목적지명을 출력하도록 하는 ret\_name 함수 입니다.

//배열 순서에 따른 목적지의 순서번호값 출력

int ret\_num(char\* a) {

    if (strcmp(a,"신창정류소") == 0) return 0;

    else if (strcmp(a, "다이소") == 0) return 1;

    else if (strcmp(a, "자취방") == 0) return 2;

    else if (strcmp(a, "세븐일레븐") == 0) return 3;

    else if (strcmp(a, "중국집") == 0) return 4;

    else if (strcmp(a, "롯데리아") == 0) return 5;

    else if (strcmp(a, "대학마트") == 0) return 6;

    else if (strcmp(a, "밀터칼국수") == 0) return 7;

    else if (strcmp(a, "신한은행") == 0) return 8;

    else if (strcmp(a, "멀티미디어관") == 0) return 9;

    else printf("error!");

    return;

}

각 목적지 명에 맞는 목적지 코드를 출력하도록 하는 ret\_num 입니다.

typedef struct GraphType {

    int n; //정점의 개수

    int weight[MAX\_VERTICES][MAX\_VERTICES];

} GraphType;

int distance[MAX\_VERTICES];

int found[MAX\_VERTICES];

int trace[MAX\_VERTICES];

int r\_trace[MAX\_VERTICES];

그래프를 정의해주는 GrapyType과 거리 및 탐색에 사용하는 변수들을 정의해준 모습입니다.

int choose(int distance[], int n, int found[]) {

    int i, min, minpos;

    min = INT\_MAX;

    minpos = -1;

    for (i = 0; i < n; i++) {

        if (distance[i] < min && !found[i]) {

            min = distance[i];

            minpos = i;

        }

    }

    return minpos;

}

해당 노드에 대한 최단 경로를 선택해 주는 choose 함수 입니다.

void shortest\_path(GraphType\* g, int start, int end) {

    int i, u, w;

    for (i = 0; i < g->n; i++) {

        distance[i] = g->weight[start][i];

        found[i] = FALSE;

        if (g->weight[start][i] != INF)

            trace[i] = start;

    }

    distance[start] = 0;

    for (i = 0; i < g->n; i++) {

        u = choose(distance, g->n, found);

        if (end == u) {

            if (start == u)

                continue;

            printf("\n");

            ret\_name(start);

            printf(" 에서 ");

            ret\_name(u);

            printf(" 까지의 루트 : ");

            ret\_name(start);

            int n\_route = trace[u];

            int q = 0;

            while (1) {

                if (n\_route == start) {

                    while (q > 0) {

                        printf("-");

                        ret\_name(r\_trace[--q]);

                    }

                    break;

                }

                r\_trace[q++] = n\_route;

                n\_route = trace[n\_route];

            }

            printf("-");

            ret\_name(u);

            printf("\t| 비용 : % d\n", distance[u]);

            for (int k = 0; k < g->n; k++) {

                distance[k] = g->weight[start][k];

                found[k] = FALSE;

            }

        }

        found[u] = TRUE;

        for (w = 0; w < g->n; w++) {//가중치계산

            if (!found[w]) {

                if (distance[u] + g->weight[u][w] < distance[w]) {

                    distance[w] = distance[u] + g->weight[u][w];

                    trace[w] = u;

                }

            }

        }

    }

}

실제 프로그램이 돌아가면서 다익스트라 알고리즘을 통해 경로의 총 거리와 경로상의 거쳐가는 목적지들을 표현해줄 shortest\_path 함수 입니다.

int main(void) {

    GraphType g = { 10,

                   {{0, 85, 100, INF, INF, INF, INF, INF, INF, INF},

                    {85, 0, 140, INF, INF, INF, INF, INF, INF, INF},

                    {100, 140, 0, 220, 180, INF, INF, INF, INF, INF},

                    {INF, INF, 220, 0, INF, INF, INF, 210, INF, INF},

                    {INF, INF, 180, INF, 0, 80, 77, INF, INF, INF},

                    {INF, INF, INF, INF, 80, 0, INF, INF, INF, INF},

                    {INF, INF, INF, INF, 77, INF, 0, 232, 324, 515},

                    {INF, INF, INF, 210, INF, INF, 232, 0, INF, 319},

                    {INF, INF, INF, INF, INF, INF, 324, INF, 0, 254},

                    {INF, INF, INF, INF, INF, INF, 515, 319, 254, 0}} };

    char start[100], end[100];

    while (1) {

        printf("\n출발지점의 이름을 입력해 주세요!(취소 = -1) : ");

        printf("\n(신창정류소, 다이소, 자취방, 세븐일레븐, 중국집)\n(롯데리아, 대학마트, 밀터칼국수, 신한은행, 멀티미디어관) : ");

        scanf("%s", start);

        if (strcmp(start, "-1") == 0)

            break;

        printf("\n도착지점의 이름을 입력해 주세요(취소 = -1) : ");

        printf("\n(신창정류소, 다이소, 자취방, 세븐일레븐, 중국집)\n(롯데리아, 대학마트, 밀터칼국수, 신한은행, 멀티미디어관) : ");

        scanf("%s", end);

        if (strcmp(end, "-1") == 0)

            break;

        //입력오류확인

        if (strcmp(start, end) == 0) {

            printf("\n같은 곳을 선택하셨습니다. 서로 다른 곳을 입력해 주세요!\n");

        }

        else if ((strcmp(start, "신창정류소") == 0 ||

            strcmp(start, "다이소") == 0 ||

            strcmp(start, "자취방") == 0 ||

            strcmp(start, "세븐일레븐") == 0 ||

            strcmp(start, "중국집") == 0 ||

            strcmp(start, "롯데리아") == 0 ||

            strcmp(start, "대학마트") == 0 ||

            strcmp(start, "밀터칼국수") == 0 ||

            strcmp(start, "신한은행") == 0 ||

            strcmp(start, "멀티미디어관") == 0) &&

            (strcmp(end, "신창정류소") == 0 ||

                strcmp(end, "다이소") == 0 ||

                strcmp(end, "자취방") == 0 ||

                strcmp(end, "세븐일레븐") == 0 ||

                strcmp(end, "중국집") == 0 ||

                strcmp(end, "롯데리아") == 0 ||

                strcmp(end, "대학마트") == 0 ||

                strcmp(end, "밀터칼국수") == 0 ||

                strcmp(end, "신한은행") == 0 ||

                strcmp(end, "멀티미디어관") == 0))

            shortest\_path(&g, ret\_num(start), ret\_num(end));

        else

            printf("\n정확한 지명을 입력해 주세요!\n");

    }

    return 0;

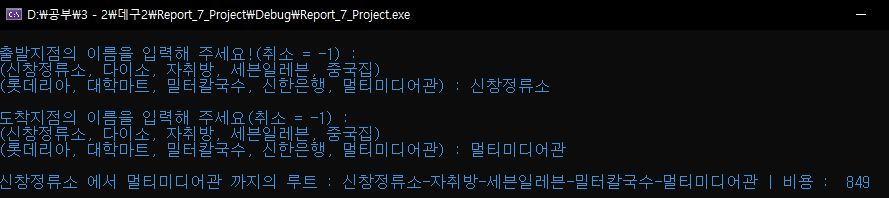
}

프로그램이 실행될 메인함수 입니다.

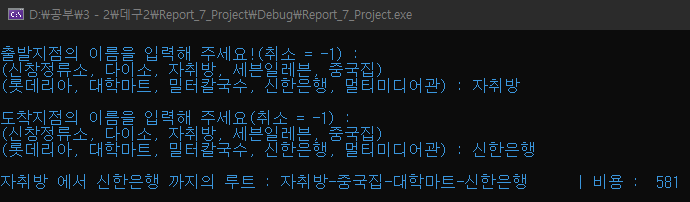
사용자로부터 목적지와 출발지를 입력받고, 해당 값이 정상적인 값일 때 다익스트라 알고리즘을 통해 거리 계산을 하도록 합니다.

4. 실행결과

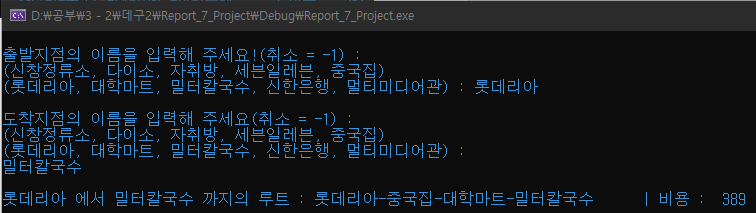
신창정류소 – 멀티미디어관



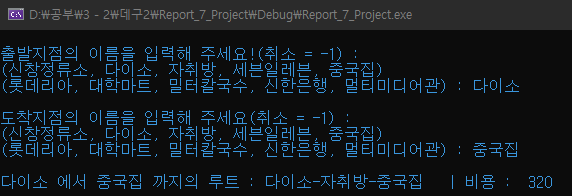
자취방 – 신한은행



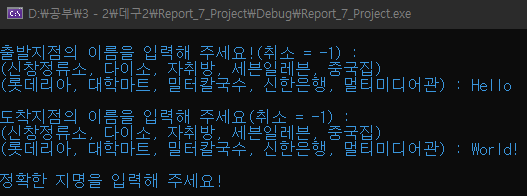
롯데리아 – 밀터칼국수



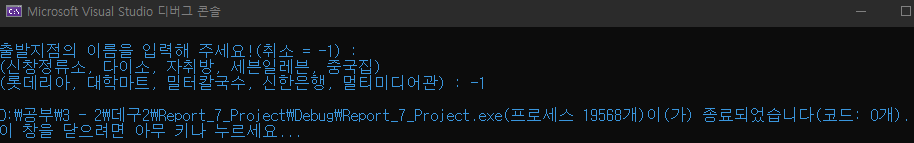
다이소 – 중국집

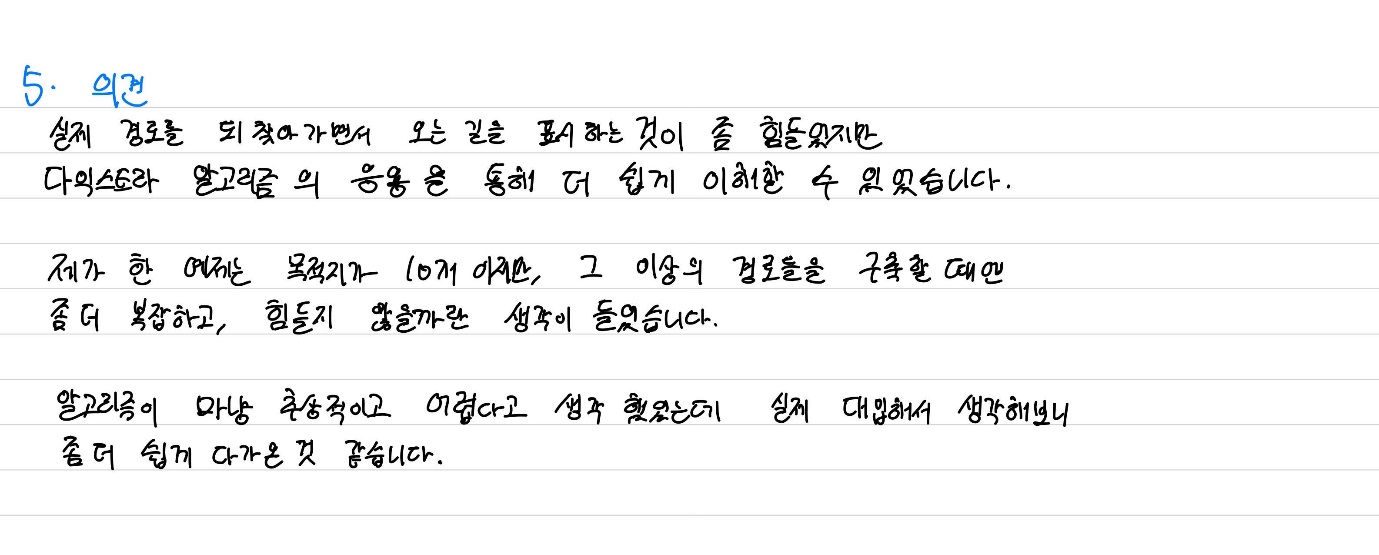


입력 오류시



프로그램 종료





이하 코드 전문입니다.

#pragma warning(disable: 4996)

#include <limits.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define TRUE 1

#define FALSE 0

#define MAX\_VERTICES 100

#define INF 1000000 //무한대. (연결이 없는 경우)

//배열순서에 따른 목적지 이름 출력

void ret\_name(int i) {

    switch (i) {

    case 0:

        printf("신창정류소");

        break;

    case 1:

        printf("다이소");

        break;

    case 2:

        printf("자취방");

        break;

    case 3:

        printf("세븐일레븐");

        break;

    case 4:

        printf("중국집");

        break;

    case 5:

        printf("롯데리아");

        break;

    case 6:

        printf("대학마트");

        break;

    case 7:

        printf("밀터칼국수");

        break;

    case 8:

        printf("신한은행");

        break;

    case 9:

        printf("멀티미디어관");

        break;

    default :

        printf("Error!");

        break;

    }

}

//배열 순서에 따른 목적지의 순서번호값 출력

int ret\_num(char\* a) {

    if (strcmp(a,"신창정류소") == 0) return 0;

    else if (strcmp(a, "다이소") == 0) return 1;

    else if (strcmp(a, "자취방") == 0) return 2;

    else if (strcmp(a, "세븐일레븐") == 0) return 3;

    else if (strcmp(a, "중국집") == 0) return 4;

    else if (strcmp(a, "롯데리아") == 0) return 5;

    else if (strcmp(a, "대학마트") == 0) return 6;

    else if (strcmp(a, "밀터칼국수") == 0) return 7;

    else if (strcmp(a, "신한은행") == 0) return 8;

    else if (strcmp(a, "멀티미디어관") == 0) return 9;

    else printf("error!");

    return;

}

typedef struct GraphType {

    int n; //정점의 개수

    int weight[MAX\_VERTICES][MAX\_VERTICES];

} GraphType;

int distance[MAX\_VERTICES];

int found[MAX\_VERTICES];

int trace[MAX\_VERTICES];

int r\_trace[MAX\_VERTICES];

int choose(int distance[], int n, int found[]) {

    int i, min, minpos;

    min = INT\_MAX;

    minpos = -1;

    for (i = 0; i < n; i++) {

        if (distance[i] < min && !found[i]) {

            min = distance[i];

            minpos = i;

        }

    }

    return minpos;

}

void shortest\_path(GraphType\* g, int start, int end) {

    int i, u, w;

    for (i = 0; i < g->n; i++) {

        distance[i] = g->weight[start][i];

        found[i] = FALSE;

        if (g->weight[start][i] != INF)

            trace[i] = start;

    }

    distance[start] = 0;

    for (i = 0; i < g->n; i++) {

        u = choose(distance, g->n, found);

        if (end == u) {

            if (start == u)

                continue;

            printf("\n");

            ret\_name(start);

            printf(" 에서 ");

            ret\_name(u);

            printf(" 까지의 루트 : ");

            ret\_name(start);

            int n\_route = trace[u];

            int q = 0;

            while (1) {

                if (n\_route == start) {

                    while (q > 0) {

                        printf("-");

                        ret\_name(r\_trace[--q]);

                    }

                    break;

                }

                r\_trace[q++] = n\_route;

                n\_route = trace[n\_route];

            }

            printf("-");

            ret\_name(u);

            printf("\t| 비용 : % d\n", distance[u]);

            for (int k = 0; k < g->n; k++) {

                distance[k] = g->weight[start][k];

                found[k] = FALSE;

            }

        }

        found[u] = TRUE;

        for (w = 0; w < g->n; w++) {//가중치계산

            if (!found[w]) {

                if (distance[u] + g->weight[u][w] < distance[w]) {

                    distance[w] = distance[u] + g->weight[u][w];

                    trace[w] = u;

                }

            }

        }

    }

}

int main(void) {

    GraphType g = { 10,

                   {{0, 85, 100, INF, INF, INF, INF, INF, INF, INF},

                    {85, 0, 140, INF, INF, INF, INF, INF, INF, INF},

                    {100, 140, 0, 220, 180, INF, INF, INF, INF, INF},

                    {INF, INF, 220, 0, INF, INF, INF, 210, INF, INF},

                    {INF, INF, 180, INF, 0, 80, 77, INF, INF, INF},

                    {INF, INF, INF, INF, 80, 0, INF, INF, INF, INF},

                    {INF, INF, INF, INF, 77, INF, 0, 232, 324, 515},

                    {INF, INF, INF, 210, INF, INF, 232, 0, INF, 319},

                    {INF, INF, INF, INF, INF, INF, 324, INF, 0, 254},

                    {INF, INF, INF, INF, INF, INF, 515, 319, 254, 0}} };

    char start[100], end[100];

    while (1) {

        printf("\n출발지점의 이름을 입력해 주세요!(취소 = -1) : ");

        printf("\n(신창정류소, 다이소, 자취방, 세븐일레븐, 중국집)\n(롯데리아, 대학마트, 밀터칼국수, 신한은행, 멀티미디어관) : ");

        scanf("%s", start);

        if (strcmp(start, "-1") == 0)

            break;

        printf("\n도착지점의 이름을 입력해 주세요(취소 = -1) : ");

        printf("\n(신창정류소, 다이소, 자취방, 세븐일레븐, 중국집)\n(롯데리아, 대학마트, 밀터칼국수, 신한은행, 멀티미디어관) : ");

        scanf("%s", end);

        if (strcmp(end, "-1") == 0)

            break;

        //입력오류확인

        if (strcmp(start, end) == 0) {

            printf("\n같은 곳을 선택하셨습니다. 서로 다른 곳을 입력해 주세요!\n");

        }

        else if ((strcmp(start, "신창정류소") == 0 ||

            strcmp(start, "다이소") == 0 ||

            strcmp(start, "자취방") == 0 ||

            strcmp(start, "세븐일레븐") == 0 ||

            strcmp(start, "중국집") == 0 ||

            strcmp(start, "롯데리아") == 0 ||

            strcmp(start, "대학마트") == 0 ||

            strcmp(start, "밀터칼국수") == 0 ||

            strcmp(start, "신한은행") == 0 ||

            strcmp(start, "멀티미디어관") == 0) &&

            (strcmp(end, "신창정류소") == 0 ||

                strcmp(end, "다이소") == 0 ||

                strcmp(end, "자취방") == 0 ||

                strcmp(end, "세븐일레븐") == 0 ||

                strcmp(end, "중국집") == 0 ||

                strcmp(end, "롯데리아") == 0 ||

                strcmp(end, "대학마트") == 0 ||

                strcmp(end, "밀터칼국수") == 0 ||

                strcmp(end, "신한은행") == 0 ||

                strcmp(end, "멀티미디어관") == 0))

            shortest\_path(&g, ret\_num(start), ret\_num(end));

        else

            printf("\n정확한 지명을 입력해 주세요!\n");

    }

    return 0;

}