**ALGORITHM**

**201133216**

**정유석**

**Problem 2 : Fire Station**

#include<stdio.h>

#include<stdlib.h>

#define INF 10000

void first\_Table(int \*\*, int); //make first table function

void last\_Table(int \*\*, int, int); //make second table function

void main(void){

int first,Last;

int current\_Position;

int \*\* table;

while(1){ //receive the intersection and number of health center , and Exception

scanf("%d %d", &first,&Last);

fflush(stdin);

if(Last <=500 && Last > 1 && first <= 100 && first > 0)

break;

printf("Error of intersection or number of Fire station -! Reinput \n");

}

while(1){ //receive the current position in intersection ( Exception current possition <= intersection)

scanf("%d",&current\_Position);

fflush(stdin);

if(current\_Position >= 0 && current\_Position<=Last)

break;

printf("Error of current postion -! Reinput \n");

}

table = (int\*\*)malloc(Last \* sizeof(int\*)); //allocate memory

first\_Table(table, Last); //function call

last\_Table(table, Last, current\_Position);

}

void first\_Table(int \*\* table, int Last){

int i, j;

int start, end, distance;

for (i = 0; i < Last; i++){ //allocate memory

table[i] = (int\*)malloc(Last \* sizeof(int));

}

for (i = 0; i < Last; i++){ //initialize first table

for (j = 0; j < Last; j++)

{

if (i == j)

table[i][j] = 0;

else

table[i][j] = INF;

}

}

for (i = 0; i < Last; i++){ //store the information

scanf("%d %d %d", &start, &end, &distance);

fflush(stdin);

table[start - 1][end - 1] = distance;

table[end - 1][start - 1] = distance;

}

}

void last\_Table(int \*\* table, int Last, int current){

int i, j, k;

int max;

int maxIndex;

//Floyed warshal algorithm

for (k = 0; k < Last; k++){

for (i = 0; i < Last; i++){

for (j = 0; j < Last; j++){

if (table[i][k] + table[k][j] < table[i][j])

table[i][j] = table[i][k] + table[k][j];

}

}

}

for (i = 0; i < Last; i++){

if (i == 0){

max = table[current -1][i];

maxIndex = i;

}

if (max < table[current- 1][i]){

max = table[current - 1][i];

maxIndex = i;

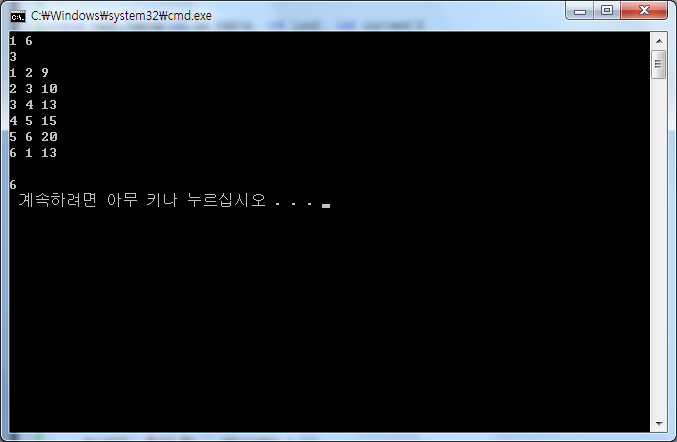
}

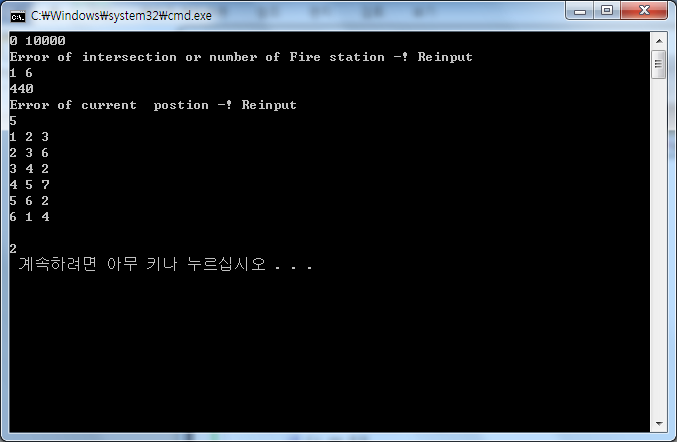
}

//Display distance

printf(" \n%d \n ", maxIndex + 1);

}





PROBLEM 3

#include<stdio.h>

#include<stdlib.h>

typedef struct{

int cen;

int location;

}car;

typedef struct{

int port,starboard;

int count;

int find;

}table;

car t[200];

table f[100][100];

void main(){

int i,j,k;

int table[100][100];

int length\_ferry,check=0,max,max\_N,comp\_P,comp\_S;

int port=0,starboard=0;

while(1){//Exception && receive ferry length

scanf("%d",&length\_ferry);

fflush(stdin);

if(length\_ferry >=1 && length\_ferry<100)

break;

printf("Error! length of ferry!\n");

}

length\_ferry = length\_ferry \* 100;

i=0;

while(1){//Exception && receive car length

scanf("%d",&t[i].cen);

fflush(stdin);

if(t[i].cen == 0)

break;

if(t[i].cen >=100 && t[i].cen <=3000){

check++;

i++;

t[i].location = 0;

continue;

}

else

printf("Error! length of car!\n");

}

//initialized

for(i=0; i<check; i++){

for(j=0; j<check; j++){

f[i][j].port = 0;

f[i][j].starboard = 0;

f[i][j].count = 0;

}

}

//dynamic programming

for(i=0; i<check; i++){

k=0;

comp\_P = 0;

comp\_S = 0;

for(j=i; j<check; j++){

if(j==i){

if(t[j].cen <= length\_ferry){

k++;

f[i][0].port = t[j].cen;

f[i][0].count = k;

f[i][0].find = 1;

comp\_P = t[j].cen;

continue;

}

else

continue;

}

if(comp\_P+t[j].cen <= length\_ferry){

k++;

f[i][j].port = f[i][j-1].port + t[j].cen;

f[i][j].find = 1;

f[i][j].count = k;

comp\_P = comp\_P + t[j].cen;

continue;

}

if(f[i][j-1].starboard + t[j].cen <= length\_ferry){

k++;

f[i][j].starboard = f[i][j-1].starboard + t[j].cen;

f[i][j].find = 2;

f[i][j].count = k;

comp\_S = comp\_S + t[j].cen;

continue;

}

if(comp\_P + t[j].cen > length\_ferry && comp\_S+ t[j].cen > length\_ferry)

break;

if(j==check-1){

if(comp\_P <= length\_ferry || comp\_S <= length\_ferry){

j = 0;

}

}

}

}

max =0;

max\_N=0;

//find max count

for(i=0; i<check; i++){

for(j=0; j<check; j++){

printf("%d ",f[i][j].count);

if(max < f[i][j].count){

max = f[i][j].count;

max\_N = i;

}

}

printf("\n");

}

//display

printf("\n%d \n",max-1);

for(i=0; i<check-1; i++){

if(f[max\_N][i].find == 1)

printf("port\n");

else if(f[max\_N][i].find == 2)

printf("starboard\n");

else

continue;

}

}

