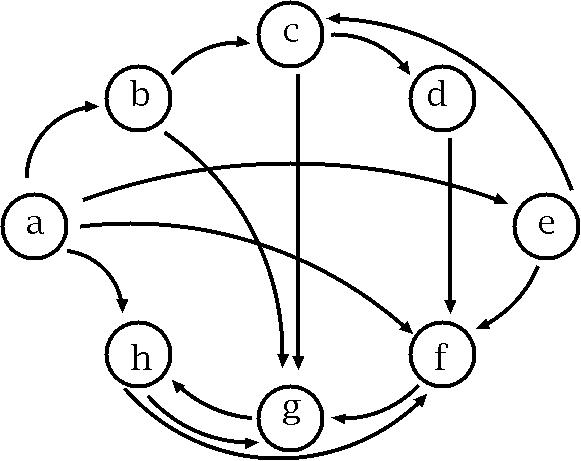
**Exercise 6. Answer Sheet**

Student's Name: Tsuyoshi Kumamoto Student's ID: s1250050

***Problem 1.*** Given the graph below



a) *(10 points)* Fill the following matrix by putting 1 if there is an edge between nodes. Put 0 otherwise.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h |
| a | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| b | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| c | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| d | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| e | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| f | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| g | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| h | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

b) *(40 points)* Write a program implementing Warshal's algorithm. Upload your code. Use your program to create a transitive closure G\* of the graph above and show it in the space below.

Transitive closure defined by adjacency table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h |
| a |  |  |  |  |  |  |  |  |
| b |  |  |  |  |  |  |  |  |
| c |  |  |  |  |  |  |  |  |
| d |  |  |  |  |  |  |  |  |
| e |  |  |  |  |  |  |  |  |
| f |  |  |  |  |  |  |  |  |
| g |  |  |  |  |  |  |  |  |
| h |  |  |  |  |  |  |  |  |

#include <stdio.h>

#include <stdlib.h>

#include <limits.h>

#define min(a, b) ((a < b) ? a : b)

int G[8][8];

int b\_num=0;

int v;

void init(){

int i, j;

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

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

G[i][j]=0;

}

}

}

void warshal\_floid(int n){

int i,j,k;

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

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

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

G[i][j]=min(G[i][j], G[i][k] + G[k][j]);

}

}

}

}

int main(){

int i,j;

char from,to;

init();

printf("Input vaertex number.\n");

scanf("%d",&v);

printf("Input branch number.\n");

scanf("%d",&b\_num);

for(i=0; i<b\_num; i++){

printf("Inupt From(char) and To(char).\n");

scanf(" %c %c",&from,&to);

printf("From:%d\n",(int)from);

G[from-10][to-10]=1;

}

warshal\_floid(v);

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

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

if(G[i][j]!=0){

printf("From %d to %d\n",(char)i+'a',(char)j+'a');

}

}

}

return 0;

}

***Problem 2.*** *(50 points)*Consider the following weight adjacency matrix.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h |
| a | 0 | 48 | ∞ | 8 | 20 | ∞ | 20 | ∞ |
| b | ∞ | 0 | 24 | ∞ | 9 | ∞ | 76 | 29 |
| c | 97 | ∞ | 0 | ∞ | ∞ | ∞ | 18 | 1 |
| d | ∞ | 52 | 34 | 0 | 29 | ∞ | ∞ | ∞ |
| e | ∞ | ∞ | ∞ | ∞ | 0 | 10 | ∞ | ∞ |
| f | ∞ | 10 | 85 | 43 | ∞ | 0 | 41 | 29 |
| g | ∞ | ∞ | ∞ | 76 | 38 | ∞ | 0 | ∞ |
| h | 28 | 42 | ∞ | 77 | 21 | ∞ | 11 | 0 |

Write a program implementing Floyd's algorithm. Upload your code. Given the matrix above, calculate all pairs shortest paths using your program and fill the table below:

All pairs shortest path table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h |
| a |  |  |  |  |  |  |  |  |
| b |  |  |  |  |  |  |  |  |
| c |  |  |  |  |  |  |  |  |
| d |  |  |  |  |  |  |  |  |
| e |  |  |  |  |  |  |  |  |
| f |  |  |  |  |  |  |  |  |
| g |  |  |  |  |  |  |  |  |
| h |  |  |  |  |  |  |  |  |