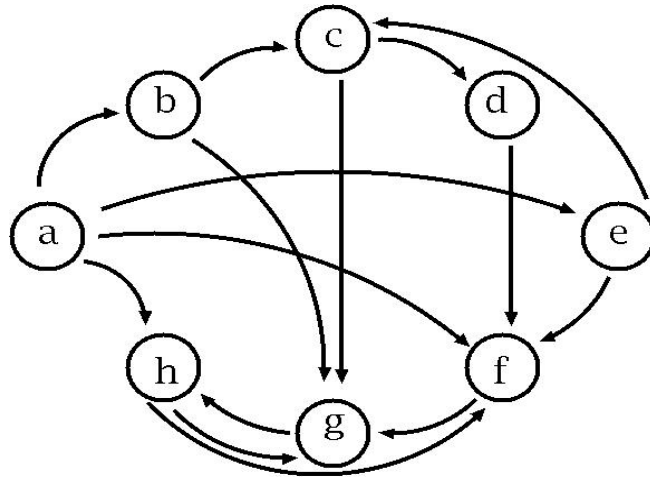


## Exercise 6. Answer Sheet

Student's Name: \_\_\_\_\_Tran Thi Thoa\_\_\_\_\_

Student's ID: \_\_\_\_\_s1242006\_\_\_\_\_

**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	1	1	0	0	1	1	0	1
b	0	1	1	0	0	0	1	0
c	0	0	1	1	0	0	1	0
d	0	0	0	1	0	1	0	0
e	0	0	1	0	1	1	0	0
f	0	0	0	0	0	1	1	0
g	0	0	0	0	0	0	1	1
h	0	0	0	0	0	1	1	1

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	1	1	1	1	1	1	1	1
b	0	1	1	1	0	1	1	1
c	0	0	1	1	0	1	1	1
d	0	0	0	1	0	1	1	1
e	0	0	1	1	1	1	1	1
f	0	0	0	0	0	1	1	1
g	0	0	0	0	0	1	1	1
h	0	0	0	0	0	1	1	1

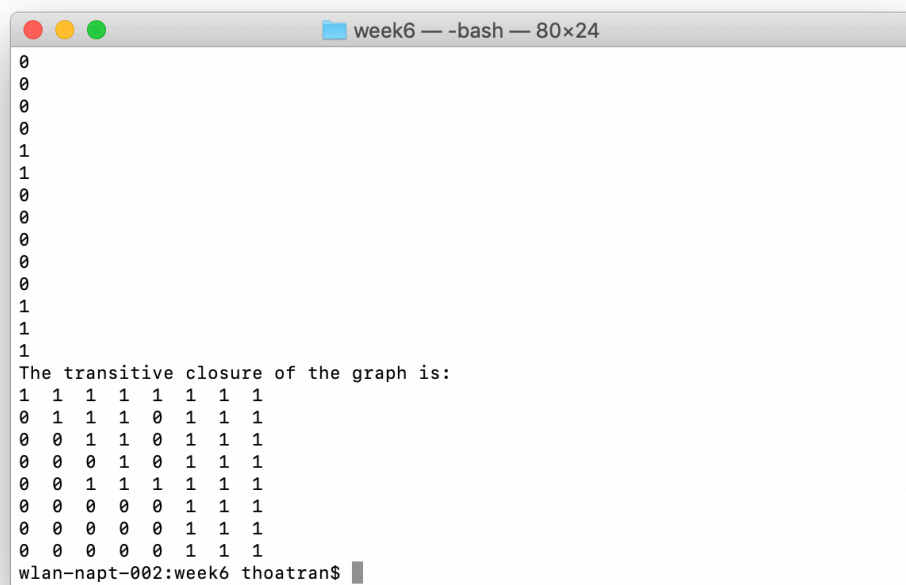
The implementation of Warshall's algorithm in C++

Input: firstly input the number of rows(equal to the number of columns) in the adjacent matrix  
then input each cell in that matrix following the rule of the warshall's algorithm

Output: the transitive closure matrix of the graph

To compile and run file, run the following command lines

```
g++ -std=c++11 -o warshall.o warshall.cpp  
./warshall.o
```



```
0  
0  
0  
0  
1  
1  
0  
0  
0  
0  
0  
1  
1  
1  
The transitive closure of the graph is:  
1 1 1 1 1 1 1 1  
0 1 1 1 0 1 1 1  
0 0 1 1 0 1 1 1  
0 0 0 1 0 1 1 1  
0 0 1 1 1 1 1 1  
0 0 0 0 0 1 1 1  
0 0 0 0 0 1 1 1  
0 0 0 0 0 1 1 1  
wlan-napt-002:week6 thoatran$
```

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

	a	b	c	d	e	f	g	h
a	0	48	$\infty$	8	20	$\infty$	20	$\infty$
b	$\infty$	0	24	$\infty$	9	$\infty$	76	29
c	97	$\infty$	0	$\infty$	$\infty$	$\infty$	18	1
d	$\infty$	52	34	0	29	$\infty$	$\infty$	$\infty$
e	$\infty$	$\infty$	$\infty$	$\infty$	0	10	$\infty$	$\infty$
f	$\infty$	10	85	43	$\infty$	0	41	29
g	$\infty$	$\infty$	$\infty$	76	38	$\infty$	0	$\infty$
h	28	42	$\infty$	77	21	$\infty$	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	0	40	42	8	20	30	20	43
b	53	0	24	61	9	19	36	25
c	29	42	0	37	22	32	12	1
d	63	49	34	0	29	39	46	35
e	67	20	44	53	0	10	50	39
f	57	10	34	43	19	0	40	29
g	105	58	82	76	38	48	0	77
h	28	41	65	36	21	31	11	0

The Floyd's algorithm is implemented in C++ in the file floyd.cpp

To compile and run the file, change the directory to the folder where you saved it and run the following command line:

```
g++ -std=c++11 -o floyd.o floyd.cpp
./floyd.o
```

Input: first input the number of rows (equal to the number of columns) of the weight matrix.

Then input the value of each cell in the weight matrix respectively.

The output will be the APSP matrix D ( the weight matrix of all pairs with the shortest path)

```
week6 — -bash — 80x24
inf
76
38
inf
0
inf
28
42
inf
77
21
inf
11
0
All pairs shortest table:
0 40 42 8 20 30 20 43
53 0 24 61 9 19 36 25
29 42 0 37 22 32 12 1
63 49 34 0 29 39 46 35
67 20 44 53 0 10 50 39
57 10 34 43 19 0 40 29
105 58 82 76 38 48 0 77
28 41 65 36 21 31 11 0
wlan-napt-002:week6 thoatran$
```