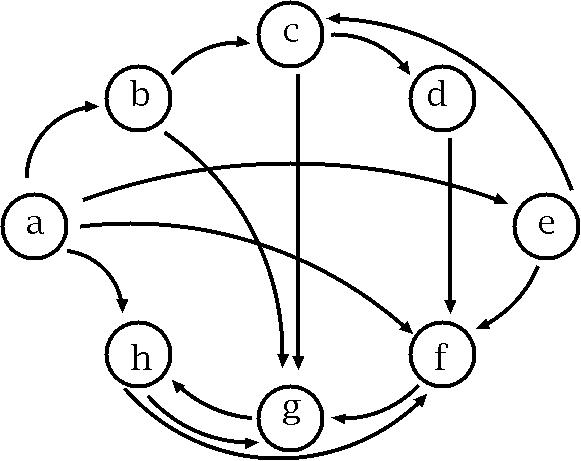
**Exercise 6. Answer Sheet**

Student's Name: Yuta Nemoto Student's ID: s1240234

***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 | 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 |

<How to compile/ run>

Input of the graph data is already did in the program (initialize() method in Warshal\_Graph class).

Please enter the command (Program source name is “Warshall\_Graph.java”:

**javac Warshall\_Graph.java**

**java Warshall\_Graph**

***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 | 0 | 40 | 42 | 8 | 20 | 30 | 20 | 43 |
| b | 53 | 0 | 24 | 62 | 9 | 19 | 36 | 25 |
| c | 29 | 42 | 0 | 75 | 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 | 74 | 21 | 31 | 11 | 0 |

<How to compile/ run>

Input of the graph data is already did in the program

(adjacencyMatrixOfProblem2() method in Floyd\_Graph class).

Please enter the command (Program source name is “Floyd\_Graph.java”:

**javac Floyd\_Graph.java**

**java Floyd\_Graph**