ENGG1330 Computer Programming I

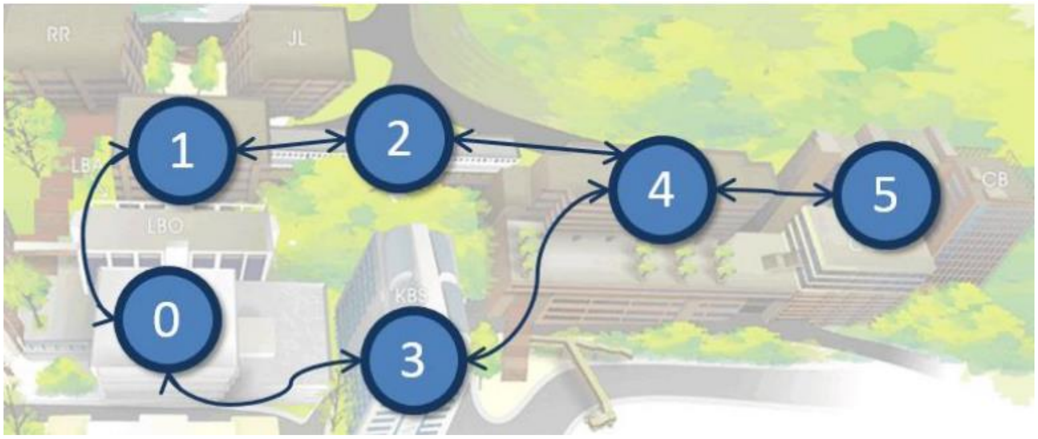
Assignment 3

Test cases to help you check your program.

You can check your program according to the following test cases.

Please note that we will add “END” command at the end of the checker program so that the program will end the while loop in the main() function.

We will call the `InsertNode` function in ascending order of the node ID. The following input is the same as the running example used in the specification.



Input 1	(Run the executable)
	`InsertNode 0 Library_Building`
	`InsertNode 1 Hui_Oi_Chow_Science_Building`
	`InsertNode 2 University_Street`
	`InsertNode 3 Kadoorie_Biological_Sciences_Building`
	`InsertNode 4 Haking_Wong_Building`
	`InsertNode 5 Chow_Yei_Ching_Building`

Output 1	(No screen output, the nodes are inserted into the graph.)

We will call the `InsertNode` function in any random order of the node ID.

Input 2	(Run the executable)
	`InsertNode 1 Hui_Oi_Chow_Science_Building`
	`InsertNode 5 Chow_Yei_Ching_Building`
	`InsertNode 0 Library_Building`
	`InsertNode 3 Kadoorie_Biological_Sciences_Building`
	`InsertNode 2 University_Street`
	`InsertNode 4 Haking_Wong_Building`

Output 2	(No screen output, the nodes are inserted into the graph.)

We will call the `InsertNode` function with node ID not starting from 0.

Input 3	(Run the executable)
	`InsertNode 10 Hui_Oi_Chow_Science_Building`
	`InsertNode 22 Chow_Yei_Ching_Building`
	`InsertNode 32 Library_Building`
	`InsertNode 25 Kadoorie_Biological_Sciences_Building`
	`InsertNode 11 University_Street`
	`InsertNode 9 Haking_Wong_Building`

Output 3	(No screen output, the nodes are inserted into the graph.)

We will call the `InsertNode` function with duplicate node ID, the node will not be inserted into the graph. The `InsertNode` function will output "`ID exists.`". The program should continue to process any upcoming commands after displaying the error message.

Input 4	(Run the executable)
	`InsertNode 0 Library_Building`
	`InsertNode 1 Hui_Oi_Chow_Science_Building`
	`InsertNode 2 University_Street`
	`InsertNode 3 Kadoorie_Biological_Sciences_Building`
	`InsertNode 4 Haking_Wong_Building`
	`InsertNode 5 Chow_Yei_Ching_Building`
	`InsertNode 5 Main_Building`

Output 4	`ID exists.`

- Note that in the above test case the graph is built for the first 6 calls of `InsertNode` function.
- The program is still active to receive upcoming commands.

We will call the `InsertEdge(x,y)` function with both `x` and `y` exist in the graph.

Input 5	Assume that we continue with the inputs of test case 1.
	`InsertEdge 0 1`
	`InsertEdge 1 0`
	`InsertEdge 1 2`
	`InsertEdge 2 1`
	`InsertEdge 0 3`
	`InsertEdge 3 0`
	`InsertEdge 2 4`
	`InsertEdge 4 2`
	`InsertEdge 3 4`
	`InsertEdge 4 3`
	`InsertEdge 4 5`
	`InsertEdge 5 4`

Output 5	(No screen output, the edges are inserted into the graph.)

We will call the `InsertEdge(x,y)` function with either `x` or `y`, or both does not exists in the graph. `InsertEdge(x,y)` should output “`No such node.`” once on screen.

Input 6	Assume that we continue with the inputs of test case 1 except END. `InsertEdge 100 1`
Output 6	`No such node.`

Input 7	Assume that we continue with the inputs of test case 1.
	`InsertEdge 1 100`
	`END`

Output 7	`No such node.`

Input 8	Assume that we continue with the inputs of test case 1.
	`InsertEdge 100 100`
	`END`

Output 8	`No such node.`

We will call the `InsertEdge(x,y)` function with duplicate edge, and the duplicated one will not be inserted into graph. In this case, `InsertEdge` function will output “`Edge exists.`”

Input 9	Assume that we continue with the inputs of test case 5. `InsertEdge 0 1`
Output 9	`Edge exists.`
Input 10	Assume that we continue with the inputs of test case 5. `InsertEdge 4 3`
Output 10	`Edge exists.`

We will call the `CommonNeighbor(x,y)` function where node `x` and `y` have common neighbors. If there are more than one common neighbors, output them in ascending order of the node ID, line by line.

Input 11	Assume that we continue with the inputs of test case 5.
	`InsertEdge 1 3`
	`InsertEdge 3 1`
	`CommonNeighbor 2 3`

Output 11	`1 Hui_Oi_Chow_Science_Building`
	`4 Haking_Wong_Building`

We will call the `CommonNeighbor(x,y)` function where node `x` and `y` do not have common neighbors. The function outputs “`No common neighbor.".`

Input 12	Assume that we continue with the inputs of test case 5. `CommonNeighbor 1 5`
Output 12	`No common neighbor.`

Input 13	Assume that we continue with the inputs of test case 5.
	`InsertNode 6 University_Hall`
	`CommonNeighbor 6 1`

Output 13	`No common neighbor.`

Input 14	Assume that we continue with the inputs of test case 5.
	`InsertNode 6 University_Hall`
	`InsertNode 7 Clinical_Pathology_Building`
	`CommonNeighbor 6 7`

Output 14	`No common neighbor.`

We will call the `CommonNeighbor(x,y)` function where node `x` and `y` are the same.

Input 15	Assume that we continue with the inputs of test case 5. `CommonNeighbor 0 0`

Output 15	`1 Hui_Oi_Chow_Science_Building`
	`3 Kadoorie_Biological_Sciences_Building`

Input 16	Assume that we continue with the inputs of test case 5.
	`InsertNode 6 University_Hall`
	`CommonNeighbor 6 6`

Output 16	`No common neighbor.`

We will call the `CommonNeighbor(x,y)` function with `x` or `y` or both does not exists in the graph. `CommonNeighbor (x,y)` should output "`No such node.`" once.

Input 17	Assume that we continue with the inputs of test case 1. `CommonNeighbor 100 0`
Output 17	`No such node.`
Input 18	Assume that we continue with the inputs of test case 1. `CommonNeighbor 0 100`
Output 18	`No such node.`
Input 19	Assume that we continue with the inputs of test case 1. `CommonNeighbor 100 100`
Output 19	`No such node.`

We will call the `ShortestPath(x,y)` function where node `x` and `y` have a path to print. If there are more than one shortest paths, output any one of them.

Input 20	Assume that we continue with the inputs of test case 5. `ShortestPath 0 4`

Output 20	`0 Library_Building`
	`3 Kadoorie_Biological_Sciences_Building`
	`4 Haking_Wong_Building`

Input 21	Assume that we continue with the inputs of test case 5. `ShortestPath 1 5`

Output 21	`1 Hui_Oi_Chow_Science_Building`
	`2 University_Street`
	`4 Haking_Wong_Building`
	`5 Chow_Yei_Ching_Building`

Input 22	Assume that we continue with the inputs of test case 5. `ShortestPath 5 0`

Output 22	`5 Chow_Yei_Ching_Building`
	`4 Haking_Wong_Building`
	`3 Kadoorie_Biological_Sciences_Building`
	`0 Library_Building`

We will call the `ShortestPath(x,y)` function where node `x` and `y` are not connected in the graph (i.e., there are no path to reach from `x` to `y` in the graph.). In this case, `ShortestPath(x,y)` outputs `“No path found.”`.

Input 23	Assume that we continue with the inputs of test case 5.
	`InsertNode 6 University_Hall`
	`ShortestPath 0 6`

Output 23	`No path found.`

Input 24	Assume that we continue with the inputs of test case 5.
	`InsertNode 7 Clinical_Pathology_Building`
	`InsertNode 8 Faculty_of_Medicine_Building`
	`InsertEdge 7 8`
	`InsertEdge 8 7`
	`ShortestPath 8 0`

Output 24	`No path found.`

Input 25	Assume that we continue with the inputs of test case 5.
	`InsertNode 7 Clinical_Pathology_Building`
	`InsertNode 8 Faculty_of_Medicine_Building`
	`InsertEdge 7 8`
	`InsertEdge 8 7`
	`InsertEdge 3 8`
	`ShortestPath 7 0`

Output 25	`No path found.`

Input 26 (say, if we have a one way shuttle from KBS to Medicine building)	Assume that we continue with the inputs of test case 5.
	`InsertNode 7 Clinical_Pathology_Building`
	`InsertNode 8 Faculty_of_Medicine_Building`
	`InsertEdge 7 8`
	`InsertEdge 8 7`
	`InsertEdge 3 8`
	`ShortestPath 0 7`

Output 26	`0 Library_Building`
	`3 Kadoorie_Biological_Sciences_Building`
	`8 Faculty_of_Medicine_Building`
	`7 Clinical_Pathology_Building`

Input 27	Assume that we continue with the inputs of test case 5.
	`InsertNode 7 Clinical_Pathology_Building`
	`ShortestPath 7 0`

Output 27	`No path found.`

We will call the `ShortestPath(x,y)` function where node `x` and `y` are the same node.

Input 28	Assume that we continue with the inputs of test case 5. `ShortestPath 0 0`
Output 28	`0 Library_Building`
Input 29	Assume that we continue with the inputs of test case 5. `ShortestPath 4 4`
Output 29	`4  Haking_Wong_Building`

Input 30	Assume that we continue with the inputs of test case 5.
	`InsertNode 6 University_Hall`
	`ShortestPath 6 6`

Output 30	`6 University_Hall`

Dear Students.

We wish you enjoy this assignment. Please feel free to let our student TAs / Kit know if you face any difficulties when working on this assignment. We are happy to help you ☺. We wish you enjoy learning programming technologies and tools in this course!

Best regards,

Kit and Dirk

-- END --