Introduction to Programming EE2310　Homework 11

103061142 楊淳佑

Problem

Based on homework 10, a student ID is added to every node, and let every node represents a student. The network becomes a friendship network. Use the Find Length function to find the shortest route from one student to the farthest friend in the starting student's network, and find how many trees needed to connect all the students in a selected department.

Solution, Additional Feature, Program Flow & Structure

Classes and Structures

- `class Node` : Include below data of a node in a ring-shaped network:
  - id(integer): The id of the node starts from 0.
  - level(integer): The level or the distance between node 0 and the node. Used in BFS.
  - neighbor(Student pointer vector): The pointers to the neighbor students.
- `class Student: public Node` : Extend the functions of Node with additional information of students' ID by inheriting from Node.
- `struct Link`: Save two student's node id to represent the link between those two nodes.

Functions, data structure and program flow

- `main()`
  - Data structure setup
    1. Get the number of students. (We set it "n+1" here.)
    2. Get the start node ID of the student network. (We set it "s" here.)
    3. Get the department number (00~08, we set it "d" here) which is going to analysis separately.
    4. Define the students vector (students), neighbors' pointer vector array (sptr) and links vector (links).
  - Target file setup
  - Create students
    1. Randomly create an ID of a student by using the student ID rule of NTHU.
    2. Assign the node ID and the student ID.

3. Repeat 1.~2. until every student has been assigned their IDs.

- Build the links
  1. Build the links vector:
     a. Save every link between every node (from node 0 to n-1) with nodes behind.
  2. Randomly swap the vector.
- Assign links
  1. Assign the first link in the links vector to the students' network vector and pop it back from the links vector. Repeat this until 8% of the links were assigned.
  2. Push the pointer of all the links on the network to `sptr`.
  3. Run `FindLengths` function again.
  4. Repeat 2.~3., until the links list is empty (all links are added to the network).
- Output the whole network [Additional]
  1. Print all of the students and their neighbors to a .csv file.
- Find the distance between `startNode` and `endNode`.
  1. Call `FindLengths` function and send the two nodes in, while the `process` is 2.
- Reset levels of all students
  1. Assign 0 to every students' level.
  2. To prepare for the calculation by department next.
- Count trees for selected department
  1. For every student, if the department is the selected one, then call `FindLength` function.
  2. Add 1 to the trees counter.
- Print out the results.

- `FindLengths(vector<Student> students, int startNode, int endNode, int department, int process)`
  - Create two `Student` pointer vector called `currentShell` (to save the nodes to be processed this level) and `nextShell` (to save the nodes going to be processed next level).
  - Push starting node (`startNode`) into `currentShell`
  - While there is nodes in `currentShell`, repeat below steps:
    1. Clear `nextShell`.
    2. Add 1 to `distance`.
    3. For each neighbor of a node in the `currentShell` :
       a. If `process` is 0 or 1, process the neighbor if the neighbor's

`level` is zero and the neighbor isn't `startNode`.

b. If `process` is 2, if the node processing matches the `endNode`, jump out of the while loop.

4. Repeat 3. until every node in the `currentShell` has been processed, copy `nextShell` to `currentShell`.

- If `process` is 2 and there is a node that matches the `endNode`, or `process` is 0/1, return the max distance. Else, return -1.

Output Result

On screen

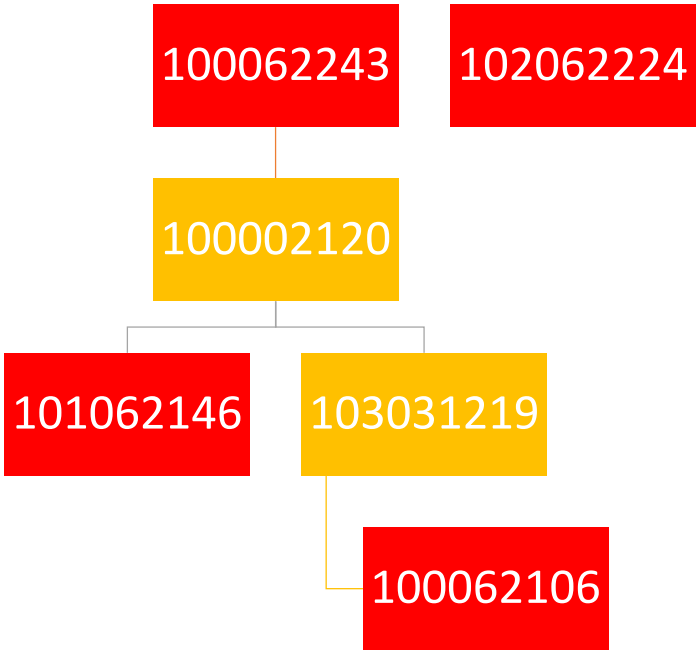
```
Total students number: 30
Select a student as start node: 0
Select a student as end node: 21
Select a department: 6
The distance between student 100081120 and student 99002237 is
2 .
Need 2 trees to link all the students in department 6.


--------------------------------
Process exited after 28.57 seconds with return value 0
請按任意鍵繼續 . . .
```

In file

STUDENT ID	FRIENDS ID			
100081120	100072237	103020229	101062146	101031203
103030115	102080242	103020229		
99021212	100082121			
100072237	100081120			
100062243	99032238	101002120		
101062146	101002120	100081120		
101002120	103031219	100082121	103050107	101062146
101081245	99030249			
102062224				
103031219	101002120	100062106		
101051238	99030249	101031203	99032238	
103020229	100081120	103030115	99002237	
100022139				

100042149	103032238	102080242	99072209	
100002221				
99030249	99070117	101081245	101051238	100081120
103050107	101002120			
99072209	100042149			
103032238	100082121	100042149	99032238	
102080242	103030115	100042149	99002237	
102021105	103072239			
99002237	99032238	103072239	102080242	103020229
100082121	101002120	103032238	99021212	103020213
103020213	100082121			
100062106	103072239	103031219		
101031203	103072239	100081120	101051238	
99070117	99030249			
99032238	101020246	100062243	99002237	103032238
101020246	99032238			
103072239	100062106	101031203	102021105	99002237

Tree List for Department Calculation



Tree List for Two Friends' Distance Calculation