





Quiz Section for Program Design (II)

Exercise #3

In this exercise, we will write a program together! This program shows how to find the person A from the person B. For example, Bella has two friends, Jon and Kelly. Jon doesn't know Kelly and Kelly doesn't know Jon either, but we still can find Kelly from Jon since he could reach out Kelly through Bella (Jon   Bella   Kelly).

We provide all source files of this program on eCourse2. Your task is trying to understand all source files and write corresponding header files. Please see the table below for the content of each source file. **Notice: Do not modify any file which we provide, if you do so, you will get zero point for this exercise.**

File Name	Content / Corresponding header file
main.c	The main procedure of this program. No corresponding header file. Yet, there's one macro " <code>_print()</code> " you have to write, and you need to decide which header file is suitable for it.
myIO.c	Deal with the major inputs and outputs of this program, and check whether the arguments are correct. Corresponding header file: " <code>myIO.h</code> ".
myDS.c	Implement stack and circular queue by using arrays. Corresponding header file: " <code>myDS.h</code> ".
myAlgo.c	Provide DFS algorithm and BFS algorithm and initialization before algorithm. Corresponding header file: " <code>myAlgo.h</code> ".

We won't use DOMjudge as a checker because once this program can be compiled by entering "`gcc main.c myIO.c myDS.c myAlgo.c`" in the command line without warnings, the program's output should be correct, because we write most of the IO part. The table below shows how to execute this program and example input and output, in case you write the "`_print()`" part incorrectly.

Command Line	Input	Output
./a.out DFS BFS	Build map Enter a person(-1 for stop): <u>1</u> Enter the number of person's friend count: <u>2</u> Enter friends separated by space: <u>2 3</u> Enter a person: <u>-1</u> Enter query: <u>2 3</u> Enter query: <u>1 4</u>	DFS: 2 > 1 > 3 BFS: 2 > 1 > 3 DFS: Stranger! BFS: Stranger!
./a.out BFS	Build map Enter a person(-1 for stop): <u>1</u> Enter the number of person's friend count: <u>1</u> Enter friends separated by space: <u>2</u> Enter a person: <u>-1</u> Enter query: <u>1 2</u>	BFS: 1 > 2
./a.out BFS DFS	Build map Enter a person(-1 for stop): <u>1</u> Enter the number of person's friend count: <u>2</u> Enter friends separated by space: <u>2 3</u> Enter a person: <u>2</u> Enter the number of person's friend count: <u>2</u> Enter friends separated by space: <u>3 4</u> Enter a person: <u>3</u> Enter the number of person's friend count: <u>1</u> Enter friends separated by space: <u>4</u> Enter a person: <u>-1</u> Enter query: <u>1 4</u>	BFS: 1 > 2 > 4 DFS: 1 > 2 > 3 > 4

./a.out DFS BFS BFS

no input

too many arguments!
usage: a.out
algorithm1
[algorithm2]