CSI 2372 – Assignment 2

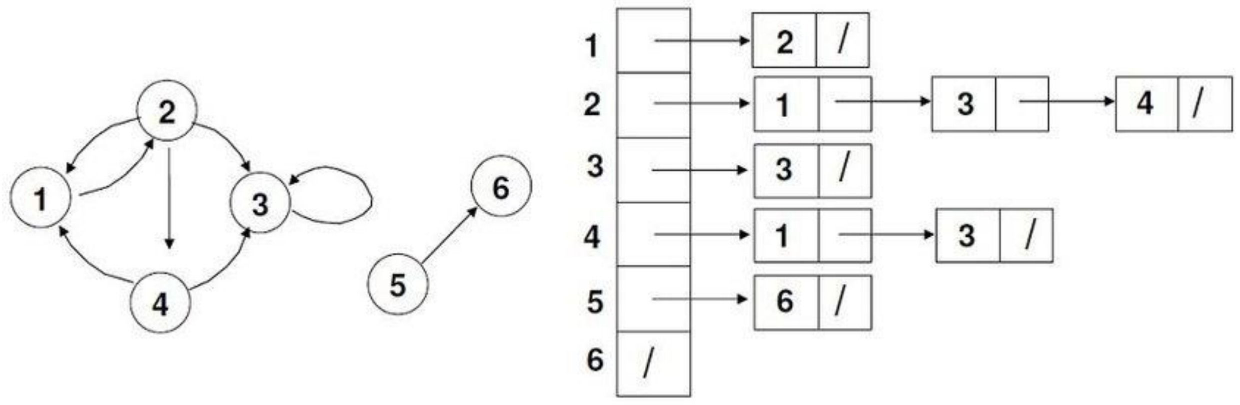
Abdorrahim Bahrami

Linked lists in C++

Your task in this assignment is to apply the concepts you learned so far about object oriented programming, and dynamic memory for linked lists in C++. Make sure you have C++ installed, and you are familiar with the header files, and coding files.

Then, you should do the following programming task. Each programming task in this assignment is a design based on the subjects you learned during lectures.

Design a class for the concept of directed graphs. There are two ways of representing a graph, which are adjacent matrix and adjacent list. Here, we want to represent a graph using adjacent lists as shown below.



You can implement it with having an array of headers of linked lists, or a linked list of headers of linked lists. If you have an array, you need to use dynamic memory and resizing. Make sure you add each node to the list of a node only once. Nodes are numbered from 1 to n, which is the number of nodes. You can use single or double linked lists. Using the programs, I wrote for the class is fine.

You are not allowed to use vector of STL. You should use dynamic memory and pointers.

Your class should have the following methods.

Class BigInteger	
Method	**Description**
Graph	The default constructor that initializes a graph of any default number of nodes without any edges
Graph	The user constructor, which initializes the graph with a given number of nodes without any edges
Graph	The copy constructor
~Graph	The destructor
add_edge	Add an edge between two given vertices
remove_edge	Remove an edge between two given vertices
edge_exist	Check if there is an edge between two given vertices
get_degree	returns the degree of a vertex (The number of vertices it is connected to)
operator ++	In both forms for adding a vertex to the graph
operator --	In both forms for removing the last vertex from the graph
path_exist	Checks if a path exists between two given vertices
connectivity_type	For checking what type of connectivity, the graph has. The method returns one of the following integers 0 for not connected 1 for weakly connected 2 for unilaterally connected 3 for strongly connected
BFS	Returns a list of integers, which shows the breadth first search of the graph starting from a given node
DFS	Returns a list of integers, which shows the depth first search of the graph starting from a given node

operator <<	For printing the graph in the following format,
	The example from the graph in the picture,
	V = {1, 2, 3, 4, 5, 6}
	E =
	{
	    1 => 2
	    2 => 1, 3, 4
	    3 => 3
	    4 => 1, 3
	    5 => 6
	    6 => None
	}

You can add any method you need.

Operators marked in red have bonus mark.

Rules

This is an individual assignment. No team work, you have to do it yourself. You are allowed to collaborate in thinking and brainstorming about problems with your classmates but you have to write the programs yourself. Any similarity between your programs is considered plagiarism. Do not use any code or program from the Internet because it is also considered plagiarism. See the university policies for plagiarism in the following link.

https://www2.uottawa.ca/about-us/provost

Measures that we take to detect plagiarism

Teaching assistants have been instructed to report to the professor any suspicion of plagiarism they find when they mark assignments.

If plagiarism has been detected in any part or in the whole assignment, the professor will take appropriate measures. Recall that it is equally bad to copy a solution and to let someone else copy your solution.