Author: Vortez Wohl
Mail: vortezwohl@proton.me
Lang: Later than C++98
Namespace : namespace wohl;
Including : "stdio.h", "string.h", "iostream"

Classes:

Class Graph.
//An object for a digraph based on adjacent chain tables.
Class GStack.
//An object for a stack based on int.
Class GQueue.
//An object for a queue based on int.

In class **Graph**:

14 methods in total.

Graph* setv(int idx);
//create a new node

Graph* setv(int* idxs);
//create a series of nodes according to an array
//remember to set the last unit END or MAXINT to mention the funtion where the end of array is

Graph* sete(int idx1,int idx2,int weit);
//set an edge linking idx2 to idx1

Graph* rmv(int idx);
//remove an existing node

Graph edgeof(int idx);**
//return an array of address and weight of an edge
//better use it with print_adjv(int idx,Graph* vert)

Graph* locate(int idx);
//return the address of current Graph in Graph list

Graph* locate(int idx1,int idx2);
//return the address of current Graph (idx2) in adjacency list of Graph idx1

void init_traversal_status();
//initialize traversal status
//if not initalized, the traversal algorithms dont work

friend void traverse_around(Graph& graph,Graph* target);
//traverse around a Graph and output to cli

friend void greedy_traversal(Graph& graph,Graph* target);
//greedily traverse the graph and output to cli

friend void DFS(Graph& graph,Graph* target);
//depth first search and output to cli

void print_Graphes();
//shows you the list of Graphes

void print_adjv(int idx,Graph* vert)
//print one edge of Graph idx
//need to be used with the method edgeof(int idx) and use it as an array

----

In class **GStack**:

2 methods in total.

//push and pop operations
int g_push(int idx);
int g_pop();

In class **GQueue**:

2 methods in total.

//enqueue and dequeue oprations
int g_enqueue(int idx);
int g_dequeue();