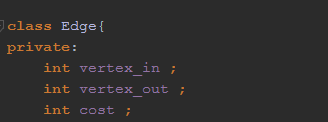
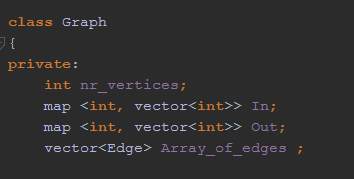
**Graphs: practical work no. 1**

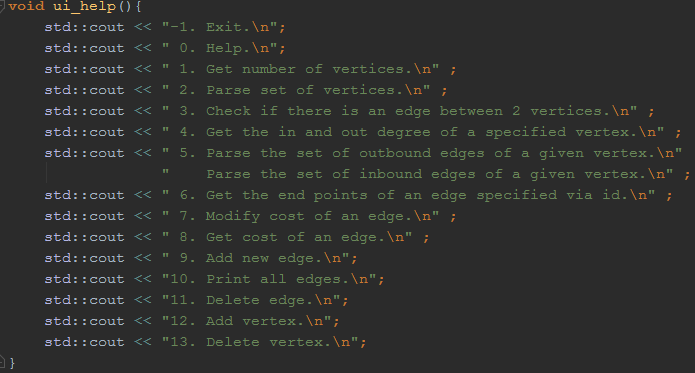
**Toadere Andreas-Robert C++**

For this implementation I chose the C++ programming language. The directed graph is represented by the number of vertices, two maps ( the in and out maps where we store which vertices are connected ) and an array of edges where we store the edges with their cost. Also, for representing an edge I have implemented a class.

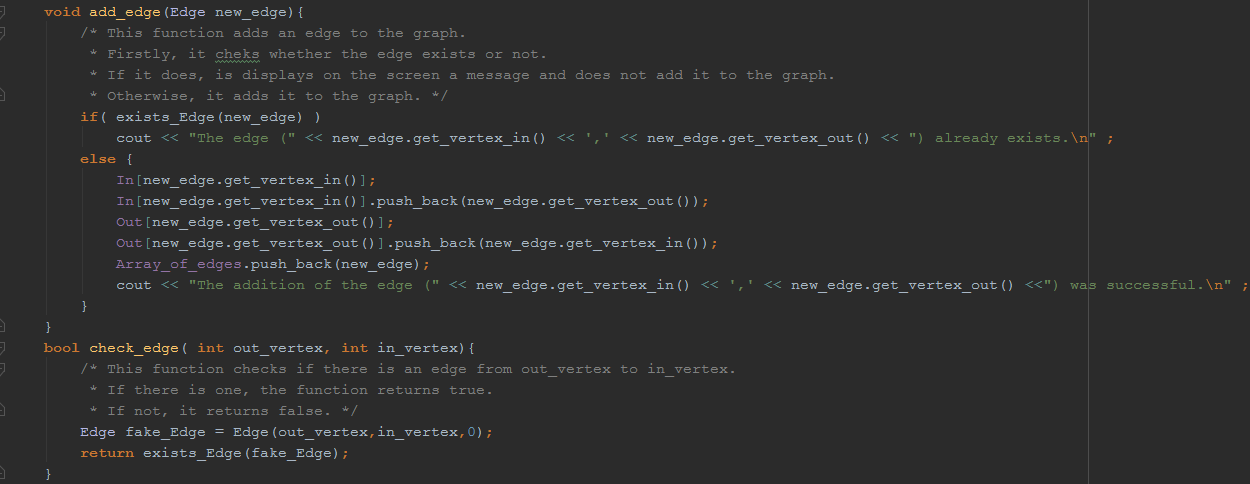




For the user-interface the program has a basic command menu, where each command is represented by a number. This is the list of commands:

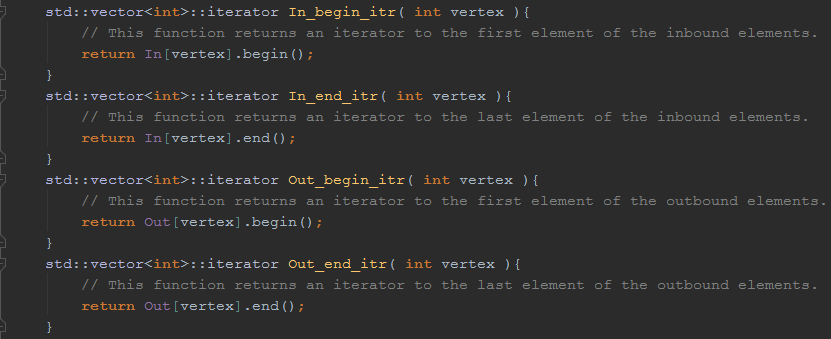


The following pictures will show the key functions of the implementations:

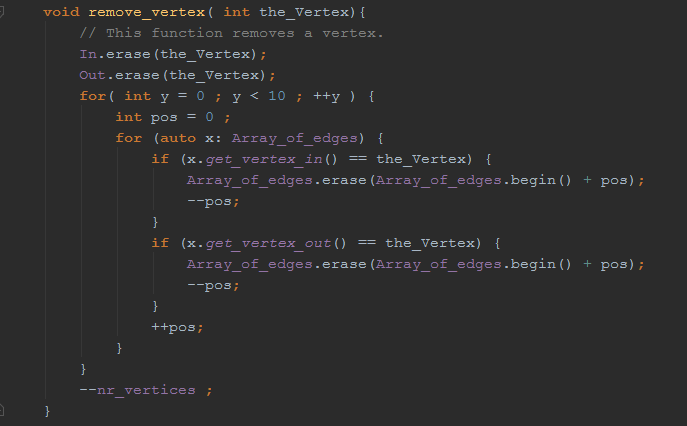


To add a new edge we need to add its vertices to the two maps and the add it in the array of edges.

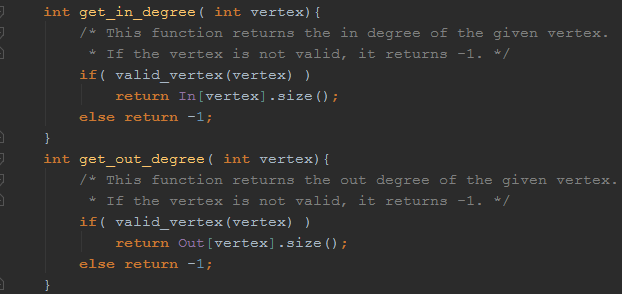
As shown above, to check if an edge exists we only need the in and out vertex.

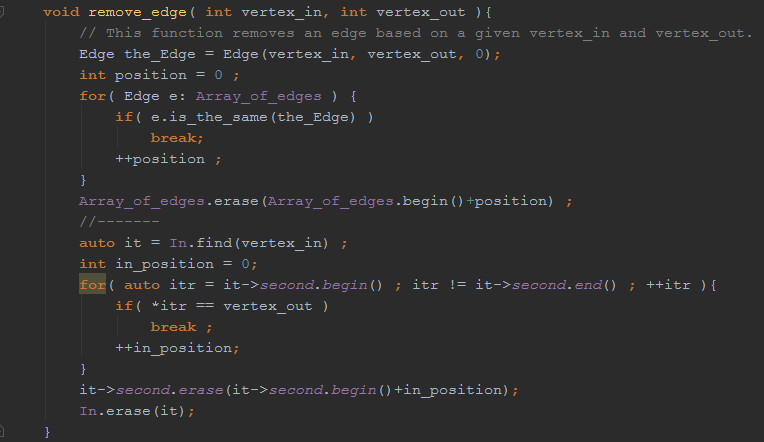


This are the functions that are used to return the proper iterators for parsing the list of in and out vertices for a given vertex.

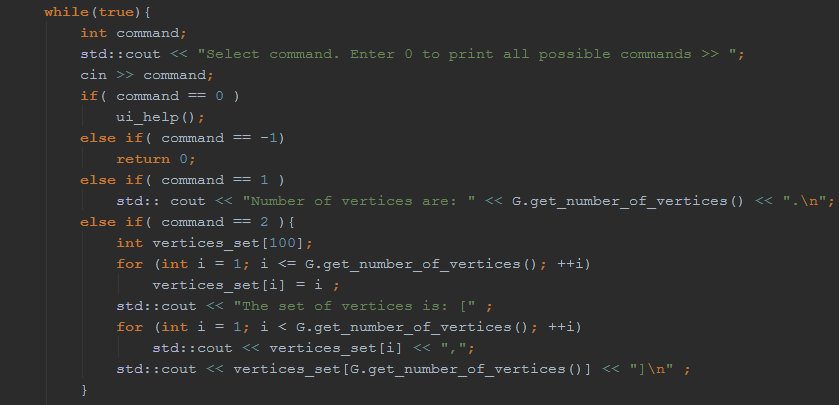


To remove a vertex we have to erase the keys and vector of vertices from the maps and the edges from the array of edges.





To remove an edge we need to go through the array of edges and eliminate the corresponding edge and then remove the vertices for the respective keys in the map.



A little bit of code for the implementation of the menu.