

Sheet 4 (Car Workshop)

Write a C program that keeps track of the customers visiting a car workshop. The program utilizes two data structures, a stack and a queue to have customers' data in particular orders.

The main program should display the following menu:

1. Add a New Customer.
 2. Serve a Customer.
 3. Display Customers Information.
 4. Display Customers information in the "most-recent" Order.
 5. Exit menu
- By choosing "Add a New Customer" you should enter the data of the new arriving customer and save it such that he has the least priority among others. → queue
 - By choosing "Serve a Customer" you should display the data of the first arriving customer then dismiss them from the system.
 - "Display Customers Information" prints on screen the data of the current waiting customers without serving them.
 - "Display Customers in the most-recent Order" without serving them - should be done by copying the data to a structure that reverses the order. → stack

Hints:

- Before coding the program first link the three files, stack.c, queue.c, and main.c; follow the following guide:
- Since the element type of the stack, queue, and the main program is the same, we need to be more structured by defining this common element type in a separate file global.h. In this file we should have all the definitions that are common to all of the three modules; these definitions are: the maximum stack or queue size, and the element_type, which will be the type of a customer and contains the following data: Name, ID.
- Now, stack.c must include stack.h and the latter includes global.h; why? Also, queue.c must include queue.h and the latter includes global.h. Finally, main.c includes all the three header files; why?

- However, this will cause a "redefinition error" since the definitions in global.h will appear again in the other two included header files because they also include global.h. To resolve this problem, we need to start global.h by:

```
#ifndef GLOBAL_H  
#define GLOBAL_H
```

then end it by ***#endif***

These statements are "Preprocessor Commands" that are processed before compilation. For more explanation refer to your C language text book.