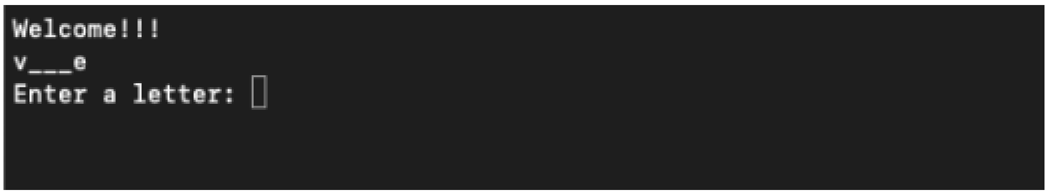
Exercises:

1.
In the 1st question you are asked to implement the game "Hangman" using Message Queues, Shared Memory and Signals.
More specifically, you need to implement two different programs, where the first will have the role of server, while the second will have the role of client.
In the beginning, the server will read a file (.txt) which will be the dictionary and will be given as a parameter by the command line. The words contained in the file will be stored in a 2D character matrix.
The initial communication between the client and the server will be achieved through a queue of messages while all other communication during the game as well as the synchronization between the two processes will be achieved using shared memory and signals respectively.
More specifically, the server will create a message queue using one known key and then the client will retrieve the queue ID using the same key. Also, the server will bind a partition of the shared memory and will be connected to it. After completing the above actions, the client will make a connection request to the server by sending the message "hi". Then, the server will select a random word from the dictionary, it will update the client for the number of letters of the word, the first as well as the last letter of the word, the number of attempts to find the word, and the identifier of the shared memory that was binded. The client will then is also connected to the shared memory and the game will start.



```
Welcome!!!
v___e
Enter a letter: 
```

After the client displays the appropriate information on the screen, as shown for example in Figure 1, will ask the user to enter a letter.
Alongside, at this point the server should be waiting for a signal to continue its execution.
After the client reads the letter from the user it will write this letter in the shared memory and will notify the server via a signal. When the server receives the signal, it will continue to run while the client will wait to receive a signal in order to continue.

The server will then read from the shared memory the letter entered by the user, will check if the letter is in the word and will write in the shared memory the positions in which the letter was found. It will also write the number of the tries left in the shared memory of the user as well as how many letters he has found in total. After writing the above in the shared memory, it will notify via one signal the client to continue its execution. The above procedure will be repeated until the user has no more attempts to find the word, or the user manages to find the word before he runs out of tries.

2.
In the 2nd question you are asked to implement a program in which a process will read letters from the keyboard, while another process will write these letters in a file. More specifically, the parent process will bind one shared memory part which will relate to a matrix of 10 characters as well a semaphore and then create a new process. This process will be responsible for reading the characters from the keyboard. In the beginning the parental process will block and wait until the child-process reads 10 characters. When the child process completes the reading of 10 characters, it should block and "notify" the parent process to continue its execution. The parent process will then write the characters read in a file and “notify” the child process to continue reading characters from the keyboard. The process will be repeated until the user enters the character ‘.’ (dot), in which case both processes should terminate their execution.

<u>The files should be .c and .h only!</u>