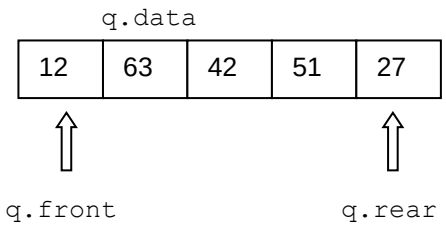
Queue Exercises

Example: Given 5 integers, insert them into a queue.

- First of all, we must initialize our queue as empty. Then, we need to read the input data using a loop. We must first read each integer into a variable, then we can insert this variable into the queue:

```
queue_t q;
int num, k;
initializeQ(&q);
for (k = 1; k <= 5; k++) {
      scanf("%d", &num);
      insert(&q, num);
}
```

- Assume that our input data is 12, 63, 42, 51, 27. After the loop terminates, all data will be put into the queue, and the queue will look like the following, if the QUEUE_SIZE was 5.



- If you are asked to fill a queue with data, no matter what the size of the queue is or no matter how many items the queue already contains, you can go on reading data until the queue becomes full. Therefore our program segment should be as follows:

```
while (!isFullQ(&q)) {
      scanf("%d", &num);
      insert(&q, num);
}
```

Example: Output the data in the queue.

- When outputting, there is an important decision to make: Do you want the data to remain in the queue, or do you want to empty the queue as you output? If you want to empty the queue each time you output the value, you need to remove it, and don't forget that this value will be lost after outputting.

```
while (!isEmptyQ(&q))
      printf("%5d\n", remove(&q));
printf("\n");
```

- After the loop terminates, we will see the following on the screen:

```
12   63   42   51   27
```

- Notice that, the output is in the same order as the input data.

- Since we removed each value before outputting, the queue is empty after the loop terminates. Thus, we destroyed the queue.

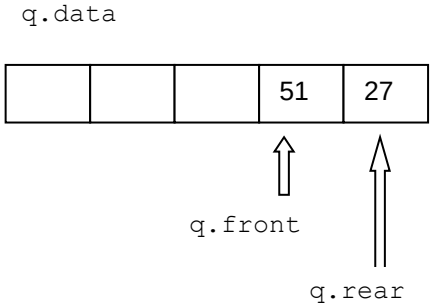
- If we don't want to destroy the queue, we can define the above program segment as a function, sending the queue as a value parameter, so that even if it becomes empty in the function, the actual queue will remain the same:

```
void displayQ(queue_t q) {
      while (!isEmptyQ(q))
           printf("%5d\n", remove(&q));
      printf("\n");
}
```

Example: Remove the first 3 items (means we want to remove 12, 63 and 42).

```
for (k = 1; k <= 3; k++)
      num = remove(&q);
```

- After the loop, the queue is as follows:



- Remember that, with two variables of the same structure, direct assignment is possible. Thus, we can copy all values in queue `q` back into queue `e` with a single assignment statement as:

```
e = q;
```

Home Exercises:

1) Define a function that removes all occurrences of a certain item from a queue, without changing the order of the other items.

2) Define a function that inserts an item after a certain item in a queue, without changing the order of the other items. Assume that there is enough space in the queue.

Example: Define a function that reverses a queue.

```
#include <stack_int.h>
#include <queue_int.h>

void reverseQueue(queue_t *q) {
      stack_t s; //local stack structure
      initializeS(&s);
      /* Remove all items from the queue and push them onto the stack */
      while (!isEmptyQ(q))
           push(&s, remove(q));
      /* Pop all items from the stack and insert them into the queue */
      while (!isEmptyS(&s))
           insert(q, pop(&s));
}
```

Example: Trace the following program segment, and show the output. Assume that `QUEUE_SIZE` is 5. (Remember that `q.rear` is initialized to -1 in `queue_int.h`)

```
#include <stack_int.h>
#include <queue_int.h>
//main
...
stack_t s;
queue_t q;
int n;
initializeS(&s);
initializeQ(&q);
while (!is_full_q(&q)) {
      insert(&q, q.rear);
      push(&s, s.top);
}
while (!isEmptyS(&s)) {
      n = pop(&s);
      printf("%d   %d\n", n, s.top);
      n = remove(&q);
      printf("%d   %d\n", n, q.front);
}
```

Homework:

Write a C program that will get the queue values from the user until the user enters a sentinel value (-9). Then the program will display a menu with the options below:

- Print Queue
- Clear Queue
- Count Queue
- Remove Maximum Element
- Send Nth To End
- Exit

After getting the choice of the user, the program will do the operations according to the choice by using the functions following:

PrintQueue	: Display the elements of the queue (Queue content will not change)
ClearQueue	: Remove all elements
CountQueue	: Count the elements in the queue
RemMaxQueue	: Remove the maximum element from the queue
SendNthToEnd	: Send the Nth element from the start to the end of the queue

Example Run:

```
Enter the numbers for the queue (-9 to stop): 3 4 5 2 -9

1) Print Queue
2) Clear Queue
3) Count Queue
4) Remove Maximum Element
5) Send Nth To End
6) Exit
Enter your choice: 1
3  4  5  2

Enter your choice: 3
Number of elements in the queue: 4

Enter your choice: 5
Enter N: 5
N must be between 1 and 4

Enter your choice: 5
Enter N: 2

Enter your choice: 1
3  5  2  4

Enter your choice: 4

Enter your choice: 1
3  2  4

Enter your choice: 2
The queue is empty!!

Enter your choice: 3
Number of elements in the queue: 0

Enter your choice: 7

Enter your choice: 6
```