CENG205 DATA STRUCTURES

ASSIGNMENT 2

Write a C program to implement **two circular queues** on **a single integer array**. You are **not allowed** to more than one array or any other data structure. Your integer array must have a fixed size of 20 elements. Your first and second queues (i.e. their front values) **must start from** indices 0 and 10, respectively. You **should consider** that any of the queues can have more elements than half size of the array (i.e. 10) at any time, depending on the commands executed.

- Read an input file (with a fixed name of input.txt) that is located near your executable.
- Input file contains 3 different commands:
    - ENQUEUE<tab><queue><tab><value>

      This command inserts the specified value to the specified queue.

      <queue> can have values of FIRST or SECOND. <value> can have any integer value.

      The output of this command is:

        - If the insertion is successful;

          <value><tab>is inserted to queue<tab><queue>

        - If the specified queue is full;

          Queue<tab><queue><tab>is full

    - DEQUEUE<tab><queue>

      This command deletes the front item from the specified queue.

      <queue> can have values of FIRST or SECOND.

      The output of this command is:

        - If the deletion is successful;

          <value><tab>is deleted from queue<tab><queue>

        - If the specified queue is empty;

          Queue<tab><queue><tab>is empty

    - PRINT <queue>

This command prints the content(s) of the specified queue(s). <queue> can have values of FIRST or SECOND.

The output of this command is:

QUEUE:<queue><tab>FRONT:<front_index><tab>REAR:<rear_index><tab>SIZE:<queue_size>

<queue_content>

<front_index> and <rear_index> is the current front and rear index values of the specified queue, respectively. <queue_size> is the number of items currently stored in the specified queue.

<queue_content> is a tab separated list of values currently stored in the specified queue, if the queue is currently not empty. Otherwise, print NO_CONTENT instead of <queue_content>.

- Example input file content:

```
DEQUEUE	FIRST
PRINT FIRST
ENQUEUE	FIRST 5
PRINT FIRST
ENQUEUE	SECOND	7
PRINT FIRST
PRINT SECOND
ENQUEUE	FIRST 3
PRINT FIRST
PRINT SECOND
DEQUEUE	SECOND
PRINT FIRST
PRINT SECOND
ENQUEUE	SECOND	9
DEQUEUE	SECOND
ENQUEUE	FIRST 10
ENQUEUE	FIRST 11
ENQUEUE	FIRST 12
ENQUEUE	FIRST 13
ENQUEUE	FIRST 14
ENQUEUE	FIRST 15
ENQUEUE	FIRST 16
ENQUEUE	FIRST 17
ENQUEUE	FIRST 18
ENQUEUE	FIRST 19
PRINT FIRST
PRINT SECOND
```

- Print the output of your program to the console. You **must strictly conform** the output formats given and you **must not print** anything else.

- Example output of the above given commands:

```
Queue FIRST is empty
QUEUE:FIRST     FRONT:0    REAR:0     SIZE:0
NO_CONTENT
5    is inserted to queue  FIRST
QUEUE:FIRST     FRONT:0    REAR:1     SIZE:1
5
7    is inserted to queue  SECOND
QUEUE:FIRST     FRONT:0    REAR:1     SIZE:1
5
QUEUE:SECOND    FRONT:10   REAR:11    SIZE:1
7
3    is inserted to queue  FIRST
QUEUE:FIRST     FRONT:0    REAR:2     SIZE:2
5    3
QUEUE:SECOND    FRONT:10   REAR:11    SIZE:1
7
7    is deleted from queue SECOND
QUEUE:FIRST     FRONT:0    REAR:2     SIZE:2
5    3
QUEUE:SECOND    FRONT:11   REAR:11    SIZE:0
NO_CONTENT
9    is inserted to queue  SECOND
9    is deleted from queue SECOND
10   is inserted to queue  FIRST
11   is inserted to queue  FIRST
12   is inserted to queue  FIRST
13   is inserted to queue  FIRST
14   is inserted to queue  FIRST
15   is inserted to queue  FIRST
16   is inserted to queue  FIRST
17   is inserted to queue  FIRST
18   is inserted to queue  FIRST
Queue FIRST is full
QUEUE:FIRST     FRONT:0    REAR:11    SIZE:11
5    3    10    11   12    13   14    15   16    17   18
QUEUE:SECOND    FRONT:12   REAR:12    SIZE:0
NO_CONTENT
```

Matters needing attention:

- Submission file structure must conform the template given below:

  < NameSurname_Studentnumber>.zip

  |---*.c

  (Name and surname information must be included in the file. Necessary information should be written separately as a description line (comment line) in the code file submission.)

- Submissions should contain only the code files but not the executables or project files.
- Assignments confronting with these submission rules will be penalized.
- Assignments should be uploaded via Gazi LMS. No other submission methods (e-mail, message) will be accepted.
- Assignments must be submitted on time. No late submissions will be accepted.
- Codes in assignments should be written in standard C programming language. Code::Blocks IDE with GCC will be used for evaluation and it is your responsibility to be sure your code works properly on this setup.
- Assignments should be prepared in English.
- Coding standards should be followed, and full attention should be paid to details such as indentation, variable/function naming. It should be detailed with comment lines.
- All external ideas, algorithms and codes must be referenced properly as comment in the code. Otherwise, assignments will be considered as copies.
- Sloppy or copy-paste assignments will be penalized.
- Must no copy, zero will be given when a copy is detected.

  (Assignments will be subject to pairwise code comparison and similar assignments will be considered as copies.)