

|              |                                  |
|--------------|----------------------------------|
| Started on   | Monday, 23 October 2023, 1:10 PM |
| State        | Finished                         |
| Completed on | Monday, 30 October 2023, 6:44 AM |
| Time taken   | 6 days 17 hours                  |
| Grade        | 300.00 out of 300.00 (100%)      |

Question **1**  
Correct  
Mark 100.00 out of 100.00

|              |       |
|--------------|-------|
| Time limit   | 1 s   |
| Memory limit | 64 MB |

Implementasikanlah ADT Queue dengan representasi array secara eksplisit dan alokasi statik. Diberikan file berikut:

- [boolean.h](#)
- [queue.h](#)

Submit file queue.c yang merupakan hasil implementasi queue.h

C

 [queue.c](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

| No | Score | Verdict  | Description       |
|----|-------|----------|-------------------|
| 1  | 10    | Accepted | 0.00 sec, 1.55 MB |
| 2  | 10    | Accepted | 0.00 sec, 1.63 MB |
| 3  | 10    | Accepted | 0.00 sec, 1.66 MB |
| 4  | 10    | Accepted | 0.00 sec, 1.60 MB |
| 5  | 10    | Accepted | 0.00 sec, 1.64 MB |
| 6  | 10    | Accepted | 0.00 sec, 1.71 MB |
| 7  | 10    | Accepted | 0.00 sec, 1.66 MB |
| 8  | 10    | Accepted | 0.00 sec, 1.60 MB |
| 9  | 10    | Accepted | 0.00 sec, 1.55 MB |
| 10 | 10    | Accepted | 0.00 sec, 1.60 MB |

Question **2**

Correct

Mark 100.00 out of 100.00

|              |       |
|--------------|-------|
| Time limit   | 1 s   |
| Memory limit | 64 MB |

Submit file **dividequeue.c** yang merupakan implementasi dari [dividequeue.h](#).  
Gunakan file yang telah disediakan dan juga submisi **queue.c** pada soal 1 untuk membantu pengerjaan soal ini.

Implementasi hanya boleh menggunakan fungsi yang sudah disediakan (CreateQueue, isEmpty, isFull, enqueue, dll) dan makro(IDX\_HEAD, IDX\_TAIL, HEAD, TAIL). Tidak boleh mengubah atau mengakses Buffer, idxHead dan idxTail.



[dividequeue.c](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

| No | Score | Verdict  | Description       |
|----|-------|----------|-------------------|
| 1  | 10    | Accepted | 0.00 sec, 1.71 MB |
| 2  | 10    | Accepted | 0.00 sec, 1.50 MB |
| 3  | 10    | Accepted | 0.00 sec, 1.64 MB |
| 4  | 10    | Accepted | 0.00 sec, 1.63 MB |
| 5  | 10    | Accepted | 0.00 sec, 1.66 MB |
| 6  | 10    | Accepted | 0.00 sec, 1.71 MB |
| 7  | 10    | Accepted | 0.00 sec, 1.71 MB |
| 8  | 10    | Accepted | 0.00 sec, 1.60 MB |
| 9  | 10    | Accepted | 0.00 sec, 1.64 MB |
| 10 | 10    | Accepted | 0.00 sec, 1.60 MB |

Question **3**

Correct

Mark 100.00 out of 100.00

|              |       |
|--------------|-------|
| Time limit   | 1 s   |
| Memory limit | 64 MB |

**Simulasi Pemutar Lagu**

**Latar Belakang**

Kamu diminta untuk membuat simulasi pemutar lagu sederhana yang bisa mengatur queue lagu, memutarnya, dan mengaktifkan fitur loop menggunakan antrian melingkar.

Silahkan gunakan file implementasi circular\_queue berikut

[circular\\_queue.h](#)

[circular\\_queue.c](#)

**Permasalahan**

Buatlah sebuah pemutar lagu yang bisa:

- Memutar lagu dari queue.
- Mengaktifkan dan menonaktifkan fitur loop.
- Menambahkan lagu ke queue berdasarkan ID-nya.
- Menambahkan beberapa lagu dari playlist ke queue.

Pemutar lagu ini menggunakan antrian melingkar untuk mengatur lagu-lagu. Saat fitur loop diaktifkan, lagu yang diputar harus kembali ke akhir queue.

**Format Masukan**

- Dua bilangan bulat  $N$  dan  $M$ .  $N$  menunjukkan jumlah lagu dan  $M$  menunjukkan jumlah playlist.
- Diikuti oleh  $M$  baris, masing-masing berisi 3 bilangan bulat yang mewakili ID lagu dalam playlist.
- Sebuah rangkaian operasi yang akan dikerjakan pada pemutar:

- `1`: Memutar lagu selanjutnya dari queue.
- `2`: Mengaktifkan/menonaktifkan fitur loop.
- `3 `: Menambahkan lagu ke queue berdasarkan ID-nya.
- `4 `: Menambahkan lagu dari playlist ke queue.
- `0`: Keluar dari program.

**Format Keluaran**

- Saat memutar lagu: `Play lagu `
- Saat queue kosong: `Tidak ada lagu`
- Saat loop diaktifkan: `Looping lagu`
- Saat loop dinonaktifkan: `Tidak looping lagu`
- Saat queue penuh saat mencoba menambahkan lagu: `Queue penuh`

Jika terdapat playlist yang memiliki lagu yang tidak ada, ( $id\_lagu < 0$  atau  $id\_lagu > N - 1$ ) maka keluarkan `Playlist tidak valid` dan berhentikan program tanpa memproses masukan lainnya.

**Batasan**

- $1 \leq N, M \leq 100$
- ID lagu dan ID playlist dimulai dari angka 0.

**Contoh 1**

Masukan:

```
10 3
1 2 3
4 5 6
7 8 9
3 1
2
1
1
2
1
1
0
```

Keluaran:

```
Looping lagu
Play lagu 1
Play lagu 1
Tidak looping lagu
Play lagu 1
Tidak ada lagu
```

Catatan

Dalam contoh di atas, pengguna menambahkan lagu `1` ke queue, memutarnya, mengaktifkan fitur loop, memutar lagu tersebut dua kali (karena di-loop), menonaktifkan loop, memutar lagu tersebut sekali lagi, dan setelah itu mencoba memutar lagi tapi queue-nya kosong.

Contoh 2

Masukan:

```
5 1
4 1 2
1
3 1
4 0
1
1
1
1
0
```

Keluaran:

```
Tidak ada lagu
Play lagu 1
Play lagu 4
Play lagu 1
Play lagu 2
```



Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

| No | Score | Verdict  | Description       |
|----|-------|----------|-------------------|
| 1  | 10    | Accepted | 0.00 sec, 1.54 MB |
| 2  | 10    | Accepted | 0.00 sec, 1.71 MB |
| 3  | 10    | Accepted | 0.00 sec, 1.66 MB |
| 4  | 10    | Accepted | 0.00 sec, 1.63 MB |
| 5  | 10    | Accepted | 0.00 sec, 1.63 MB |
| 6  | 10    | Accepted | 0.00 sec, 1.54 MB |
| 7  | 10    | Accepted | 0.00 sec, 1.65 MB |
| 8  | 10    | Accepted | 0.00 sec, 1.67 MB |
| 9  | 10    | Accepted | 0.00 sec, 1.55 MB |
| 10 | 10    | Accepted | 0.00 sec, 1.65 MB |