Circular Array "Queue"

++front;
front \% = size; \phi \left( \frac{\text{size}}{2} \right)
array[front] = \sim 5 \text{ op} \& \text{ fetches}

(++front) \% = \text{size}
1) 

```
LLL - Queue
```

2) 

```
if (front)
{
    temp = front->next;
    delete front;
    front = temp;
    if (front == NULL)
        head = NULL;
}
```

```
if (rear)
{
    rear->next = new node;
    rear = rear->next;
    rear->data = ...;
    rear->next = NULL;
}
```
2) `head` if (front) {
    `temp = front->next`;
    delete front;
    front = temp;
    if (front == NULL)
        `rear = NULL`;
} else {
    `front = new node`;
    `front->data = ...`;
    `front->next = NULL`;
    `rear = front`;
}

`front` if (rear) {
    `rear->next = new node`;
    `rear = rear->next`;
    `rear->data = ...`;
    `rear->next = NULL`;
} else {
    `front = new node`;
    `front->data = ...`;
    `front->next = NULL`;
    `rear = front`;
}
LLL - Queue

Enqueue: $10 \text{ ops} + 5f \approx 16 \text{ op/fetch}$

Dequeue: $6 \text{ ops} + 5f \approx 11 \text{ op/fetch}$
CLL - Queue

1 node

Empty List

```
pointer qptr
```

```
temp = qptr->next;
qptr->next = temp->next;
delete temp;
```

```
qptr
```

```
temp = qptr->next;
qptr->next = new node;
qptr = qptr->next;
qptr->next = temp;
```
DLL - Deque

Dequeue @ tail

\[ P \]

\[ \text{temp = tail; tail = tail -> prev}; \]

\[ \text{tail -> next = NULL; delete temp;} \]
node * current = head;
node * previous = NULL;
while (current && current->data != match) {
    previous = current;
    current = current->next;
}
6 fetches + 7 ≥ 13
Node *current = head;
if (current)
while (current->next &&
       current->next->data != match)
    current = current->next;

10 op + 7 = ≈ 17