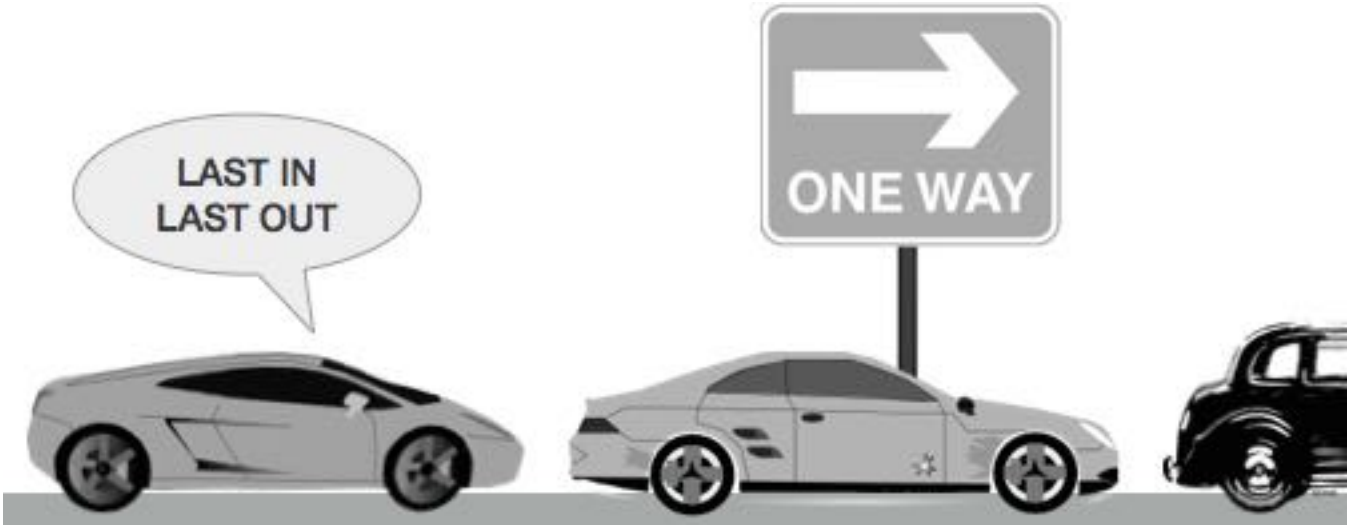
Cấu trúc dữ liệu hàng đợi (Queue)

Cấu trúc dữ liệu hàng đợi (Queue) là gì ?

Hàng đợi (Queue) là một cấu trúc dữ liệu trừu tượng, là một cái gì đó tương tự như hàng đợi trong đời sống hàng ngày (xếp hàng).



Khác với ngăn xếp, hàng đợi là mở ở cả hai đầu. Một đầu luôn luôn được sử dụng để chèn dữ liệu vào (hay còn gọi là sắp vào hàng) và đầu kia được sử dụng để xóa dữ liệu (rời hàng). Cấu trúc dữ liệu hàng đợi tuân theo phương pháp First-In-First-Out, tức là dữ liệu được nhập vào đầu tiên sẽ được truy cập đầu tiên.

Trong đời sống thực chúng ta có rất nhiều ví dụ về hàng đợi, chẳng hạn như hàng xe ô tô trên đường một chiều (đặc biệt là khi tắc xe), trong đó xe nào vào đầu tiên sẽ thoát ra đầu tiên. Một vài ví dụ khác là xếp hàng học sinh, xếp hàng mua vé, …

Biểu diễn cấu trúc dữ liệu hàng đợi (Queue)

Giờ thì có lẽ bạn đã tưởng tượng ra hàng đợi là gì rồi. Chúng ta có thể truy cập cả hai đầu của hàng đợi. Dưới đây là biểu diễn hàng đợi dưới dạng cấu trúc dữ liệu:

Tương tự như cấu trúc dữ liệu ngăn xếp, thì cấu trúc dữ liệu hàng đợi cũng có thể được triển khai bởi sử dụng Mảng (Array), Danh sách liên kết (Linked List), Con trỏ (Pointer) và Cấu trúc (Struct). Để đơn giản, phần tiếp theo chúng ta sẽ tìm hiểu tiếp về hàng đợi được triển khai bởi sử dụng mảng một chiều.

Các hoạt động cơ bản trên cấu trúc dữ liệu hàng đợi

Các hoạt động trên cấu trúc dữ liệu hàng đợi có thể liên quan tới việc khởi tạo hàng đợi, sử dụng dữ liệu trên hàng đợi và sau đó là xóa dữ liệu khỏi bộ nhớ. Danh sách dưới đây là một số hoạt động cơ bản có thể thực hiện trên cấu trúc dữ liệu hàng đợi:

- **Hoạt động enqueue()**: thêm (hay lưu trữ) một phần tử vào trong hàng đợi.
- **Hoạt động dequeue()**: xóa một phần tử từ hàng đợi.

Để sử dụng hàng đợi một cách hiệu quả, chúng ta cũng cần kiểm tra trạng thái của hàng đợi. Để phục vụ cho mục đích này, dưới đây là một số tính năng hỗ trợ khác của hàng đợi:

- **Phương thức peek()**: lấy phần tử ở đầu hàng đợi, mà không xóa phần tử này.
- **Phương thức isFull()**: kiểm tra xem hàng đợi là đầy hay không.
- **Phương thức isEmpty()**: kiểm tra xem hàng đợi là trống hay hay không.

Trong cấu trúc dữ liệu hàng đợi, chúng ta luôn luôn: (1) dequeue (xóa) dữ liệu được trỏ bởi con trỏ **front** và (2) enqueue (nhập) dữ liệu vào trong hàng đợi bởi sự giúp đỡ của con trỏ**rear**.

Trong phần tiếp chúng ta sẽ tìm hiểu về các tính năng hỗ trợ của cấu trúc dữ liệu hàng đợi:

Phương thức peek() của cấu trúc dữ liệu hàng đợi

Giống như trong cấu trúc dữ liệu ngăn xếp, hàm này giúp chúng ta quan sát dữ liệu tại đầu hàng đợi. Giải thuật của hàm peek() là:

```
bắt đầu hàm peek     return queue[front]     kết thúc hàm
```

Sự triển khai của hàm peek() trong ngôn ngữ C:

```
int peek() {    return queue[front]; }
```

Phương thức isFull() trong cấu trúc dữ liệu hàng đợi

Nếu khi chúng ta đang sử dụng mảng một chiều để triển khai hàng đợi, chúng ta chỉ cần kiểm tra con trỏ rear có tiến đến giá trị MAXSIZE hay không để xác định hàng đợi là đầy hay không. Trong trường hợp triển khai hàng đợi bởi sử dụng Danh sách liên kết vòng (Circular Linked List), giải thuật cho hàm isFull() sẽ khác.

Phần dưới đây là giải thuật của hàm isFull():

```
bắt đầu hàm isfull     if rear equals to MAXSIZE       return true    else
return false    endif     kết thúc hàm
```

Sự triển khai giải thuật của hàm isFull() trong ngôn ngữ C:

```
bool isfull() {    if(rear == MAXSIZE - 1)       return true;    else
return false; }
```

Phương thức isEmpty() trong cấu trúc dữ liệu hàng đợi

Giải thuật của hàm isEmpty():

```
bắt đầu hàm isempty     if front là nhỏ hơn MIN  OR front là lớn hơn rear
return true    else       return false    kết thúc if     kết thúc hàm
```

Nếu giá trị của **front** là nhỏ hơn **MIN** hoặc 0 thì tức là hàng đợi vẫn chưa được khởi tạo, vì thế hàng đợi là trống.

Dưới đây là sự triển khai code trong ngôn ngữ C:

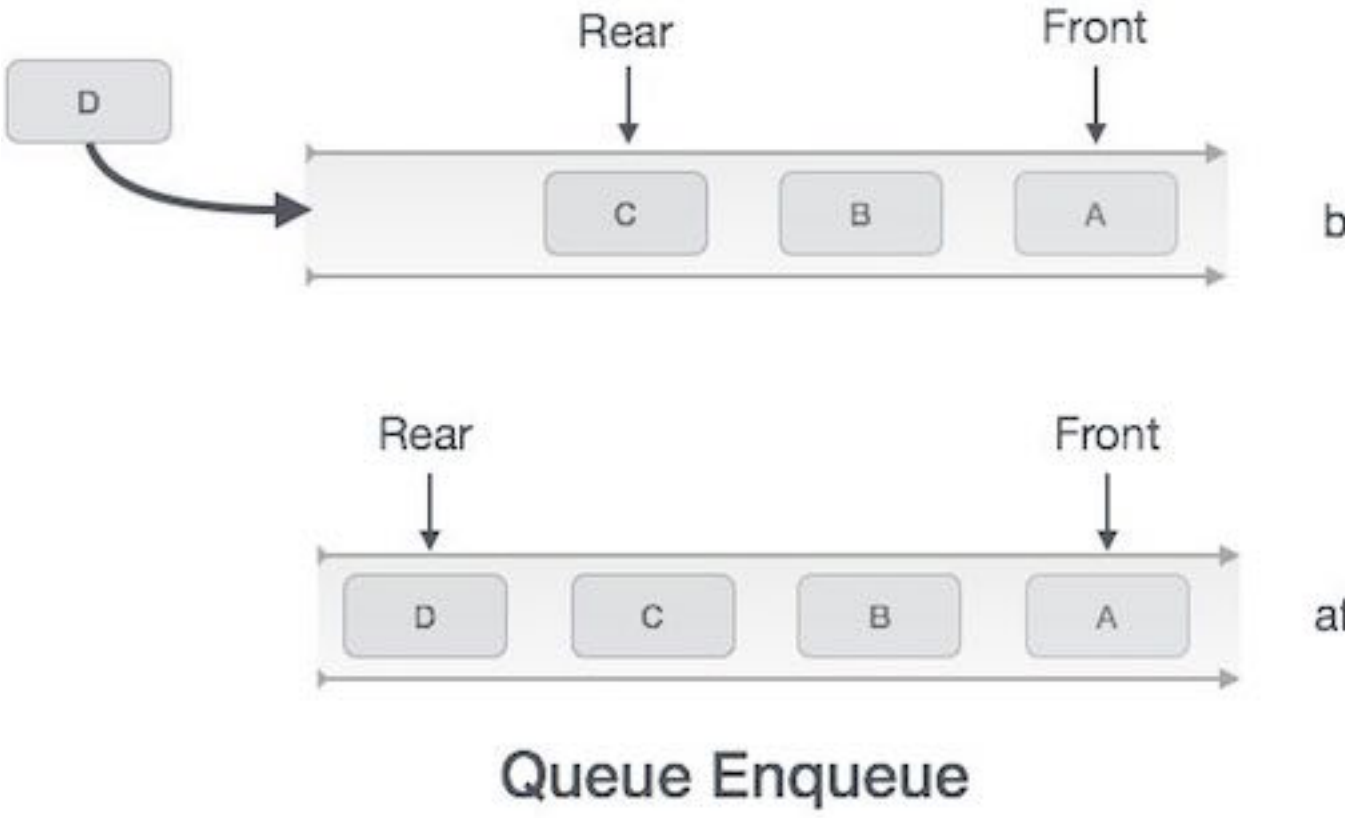
```
bool isempty() {    if(front < 0 || front > rear)       return true;    else
return false; }
```

Hoạt động enqueue trong cấu trúc dữ liệu hàng đợi

Bởi vì cấu trúc dữ liệu hàng đợi duy trì hai con trỏ dữ liệu: front và rear, do đó các hoạt động của loại cấu trúc dữ liệu này là khá phức tạp khi so sánh với cấu trúc dữ liệu ngăn xếp.

Dưới đây là các bước để enqueue (chèn) dữ liệu vào trong hàng đợi:

- **Bước 1**: kiểm tra xem hàng đợi là có đầy không.
- **Bước 2**: nếu hàng đợi là đầy, tiến trình bị lỗi và bị thoát.
- **Bước 3**: nếu hàng đợi không đầy, tăng con trỏ rear để trỏ tới vị trí bộ nhớ trống tiếp theo.
- **Bước 4**: thêm phần tử dữ liệu vào vị trí con trỏ rear đang trỏ tới trong hàng đợi.
- **Bước 5**: trả về success.



Đôi khi chúng ta cũng cần kiểm tra xem hàng đợi đã được khởi tạo hay chưa để xử lý các tình huống không mong đợi.

Giải thuật cho hoạt động enqueue trong cấu trúc dữ liệu hàng đợi

```
bắt đầu enqueue(data)          if queue là đầy       return overflow    endif
rear ← rear + 1        queue[rear] ← data        return true     kết thúc hàm
```

Sự triển khai giải thuật của hoạt động enqueue() trong ngôn ngữ C:

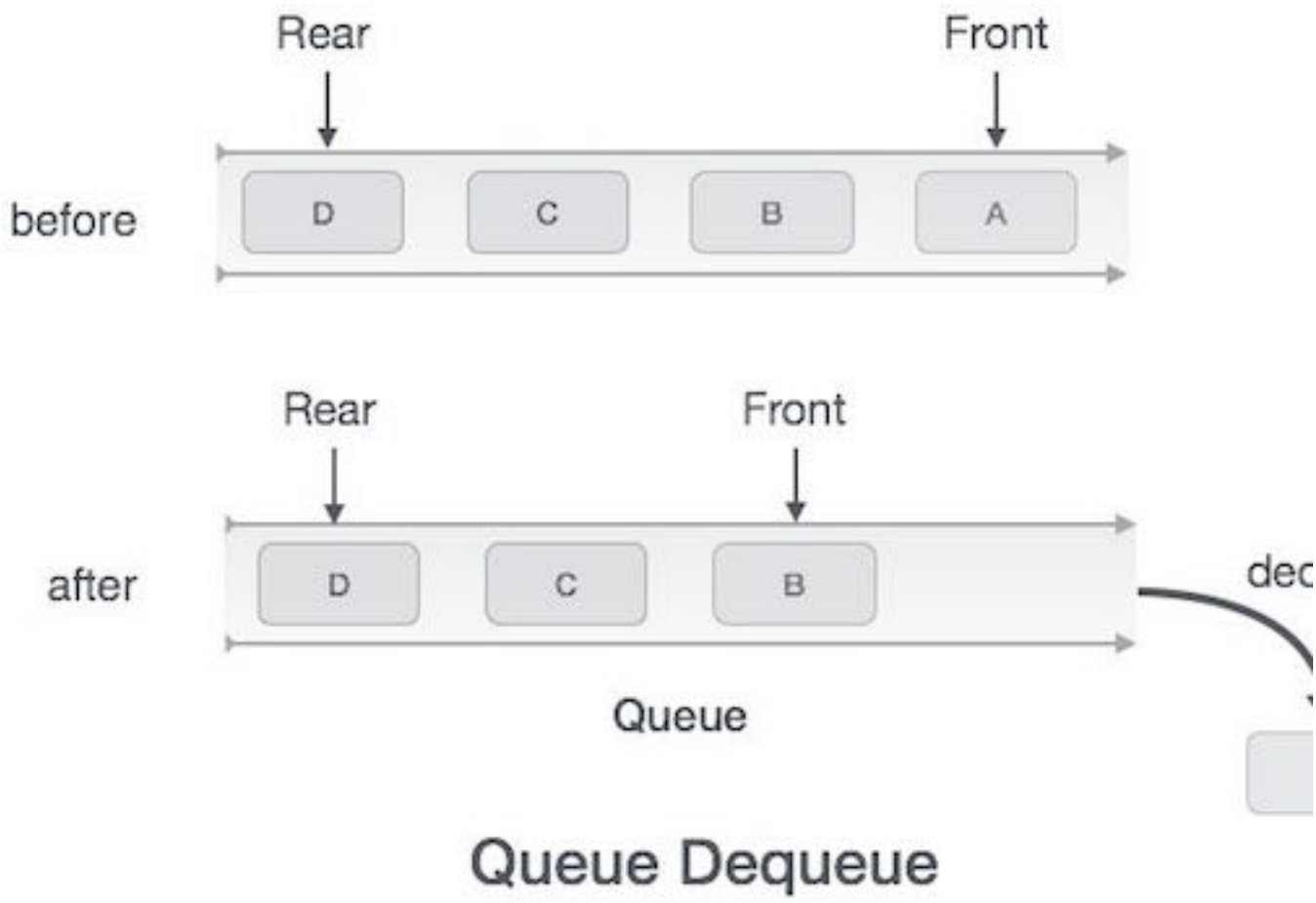
```
int enqueue(int data)          if(isfull())       return 0;        rear = rear
+ 1;    queue[rear] = data;        return 1; kết thúc hàm
```

Để theo dõi sự triển khai code đầy đủ của các hoạt động trên trong ngôn ngữ C, mời bạn click chuột vào chương: **Hàng đợi trong C**.

Hoạt động dequeue trong cấu trúc dữ liệu hàng đợi

Việc truy cập dữ liệu từ hàng đợi là một tiến trình gồm hai tác vụ: truy cập dữ liệu tại nơi con trỏ **front** đang trỏ tới và xóa dữ liệu sau khi đã truy cập đó. Dưới đây là các bước để thực hiện hoạt động **dequeue**:

- **Bước 1**: kiểm tra xem hàng đợi là trống hay không.
- **Bước 2**: nếu hàng đợi là trống, tiến trình bị lỗi và bị thoát.
- **Bước 3**: nếu hàng đợi không trống, truy cập dữ liệu tại nơi con trỏ **front** đang trỏ.
- **Bước 4**: tăng con trỏ front để trỏ tới vị trí chứa phần tử tiếp theo.
- **Bước 5**: trả về success.



Queue Dequeue

Giải thuật cho hoạt động dequeue

```
bắt đầu hàm dequeue    if queue là trống       return underflow    end if
data = queue[front]    front ← front + 1         return true kết thúc hàm
```

Sự triển khai hoạt động dequeue() trong ngôn ngữ C:

```
int dequeue() {     if(isempty())       return 0;     int data = queue[front];
front = front + 1;     return data; }
```

Để theo dõi sự triển khai code đầy đủ của các hoạt động trên trong ngôn ngữ C, mời bạn click chuột vào chương: **Hàng đợi trong C**.