



Queues

Queue is a container datatype that stores items in "First In First Out" order (FIFO)

جزء من insert list (أو list) و list هو ما يسمى Queue

(queue) rear item (آخر) ↑
Front item (أول) ↑

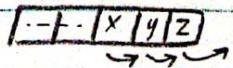
- Front item → first item that enters queue and waits longer than others in queue
- rear item → the most recent addition to queue

يكتسب يوزع سترجع من الأول و لا يجوز بياخد من الآخر.

Difference between queues & Lists

- 1- queue is a restricted form of a list
- 2- addition to the queue (enqueue) must occur to the rear
- 3- deletion from the queue (dequeue) must occur to the front

و لكن كذا، OS يتحكم بشكل كبير في queues & buffers



buffers

Queue theory

ذلك ليس يعني أن queue يتابع behaviorApps أو buffer behaviorApps، بل هو يشير إلى أن هناك تشتت المطلبة أو في البيئة أو زر ما يقولنا قلل في OS (أو buffers) كل ذلك



Difference between queues & Lists

Queue

هنا العكس بـ insert من آخر
الـ queue في الـ rear وـ remove
(front) delete من الأول (الـ front)

List

لزム أصل insert من أول الـ list
وـ remove من الآخر
وـ push/pop من linked list

Inorder List ← Special Case of priority Queue
والـ priority elements حواه هو الـ order
والـ order يحدد الـ priority

Static Array

- Very Common (in embedded sys)
 - the queue has fixed size
- لذلك كل ما حوز مكان يجوز
مكان جديد غير heap

Dynamic Linked Structure

Note:

الـ Queue حاجة مختلفة عن الـ Stack
وـ one input one read ولكن الـ queue
دخل من الـ front وـ يقرأ من الـ rear

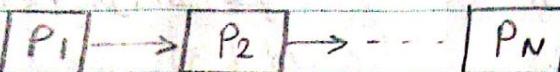
Queue in applications

* لو هناك مطبعة printer وعاليز ابعتها حاجة من الكمبيوتر
تطبعها فالحاجة الى بتبع دوى بتغير سرعتها عالية والـ printer
محتجة تستقبلا سرعة أقل عما يستقبلها صبح فبحافظ
نخزن فيه الحاجات الجاهزة من الكمبيوتر والـ printer يستقبل
ال الحاجات واحدة واحدة من الـ buffer على مدها فكل ٥ ثواني
حاجتين الـ printer لـ استقبل صبح ومضيحتش وقت النهار
(الـ buffer هو الـ queue)

* حاجة كمان لـ printer لـ one resource يعنيSingle resource
user يستخدم queue رى سباق تذاكر المترو يستخدم كـ token من شخص
واقفين في طابور

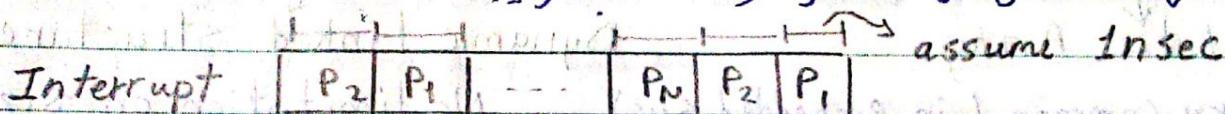


* Cpu Scheduling \leftarrow زر الـ tasks الكثيرة اللي بتتلقى خالدة من نفس الوقت ضرالـ CPU وحتاج الدلينا كويس عبايه تستعمل كلها ص



CPU

دلو قتن عندى أكتر من process حالية فبنقسم الـ time slots على يوم بحيث كل task تأخذ time slot واحدة ولفتره sec ولو الـ task محتاجة أكتر من ١٢ time slot المهمية دي سبلاها من أول الـ queue وأرجعوها من الآخر وافتقد task غيرها في الـ sec و لكن ارجع للـ sec مكمانتش في الآخر وهكذا الترتيب



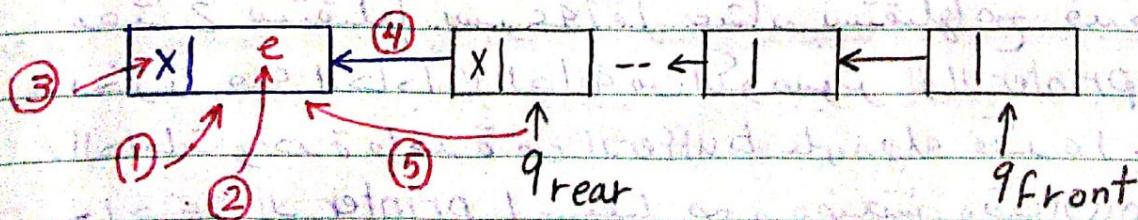
task Scheduling

ال interruption * task queue وارجح اكمال ال schedule

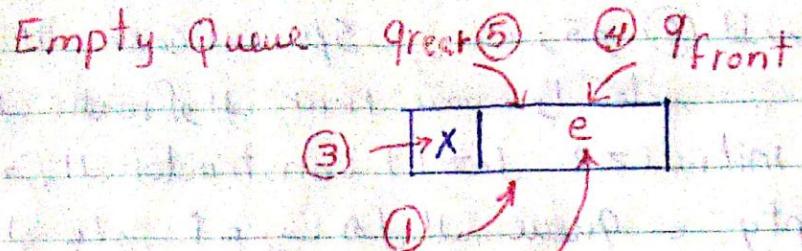
designing applications كفايات

Queue Design

General way to insert in queue

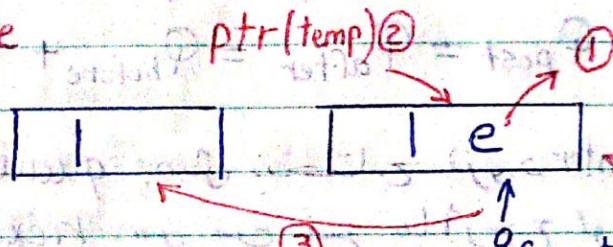


- (1) allocate new node (2) Store the element in the node
(3) Set next of added node to Null
(4) Set next of rear to added Node
(5) Set rear to the added Node



- (1) allocate new Node
 - (2) store element in the added Node
 - (3) set next of added Node to Null
 - (4) set front to added Node
 - (5) set rear to added Node
- (4) ملخص الケース؛ empty queue ||| general case

Dequeue:

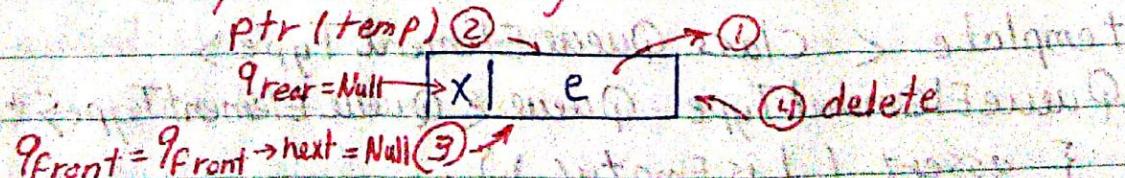


- ① take element from front
- ② set pointer temp = qfront
- ③ set next of the front to qfront
- ④ delete qfront (the front Node)

بعد إدخال next of front = 5, 3, 4 قبل 4 إلأى front Note

all item الـ item يبقى في Stack والـ queue الـ item . Note
item الـ item and its Value الـ item list الـ item و

Special Case: queue has only one element



- ① take element
- ② set pointer temp = qfront
- ③ Set next of front to qfront (Set qfront to Null)
- ④ delete this Node
- ⑤ Set qrear to Null



Sat	Sun	Mon	Tue	Wed	Thu	Fri
-----	-----	-----	-----	-----	-----	-----

خواص queue لذم زودت رقم ⑤ Special Case ال rear front كذا rear front

- * من ال بداعي queue'll initialize با Constructor
- * لو عاوز تعرف هل ال empty < queue هل ساواها NULL
- (rear - front) pointers'll use check if

Queue Operation:

→ Enqueue :

pre condition: $\text{Node} \neq \text{empty}$ و مكونش ال empty < queue

post condition: $Q_{\text{post}} = Q_{\text{after}} = Q_{\text{before}} + x$ added to the rear

و دلالة من prev pointer من بحتاج تزود من next ما ارجع لورا نفس محتاجين double Linked List

first و من first dequeue و enqueue فيه queue خواص first و last و current

→ Qfront "Get the data without removing it"

pre condition: Queue is not empty

post Condition: $Q_{\text{after}} = Q_{\text{before}}$

Return: return item stored in queue front

template < class Queue Element Type >

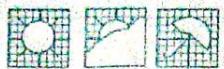
QueueElementType Queue < Queue ElementType > :: front()

{ assert(! isEmpty());

return qfront → element;

}

1mon2011



Sat Sun Mon Tue Wed Thu Fri

Page _____

Date _____

11. h file

```
template < class QueueElemType >
```

```
class Queue {
```

```
public:
```

```
bool enqueue ( const QueueElemType & e )
```

```
QueueElemType dequeue ();
```

```
QueueElemType front();
```

```
bool isEmpty();
```

```
private:
```

```
struct Node
```

```
typedef Node* Link;
```

```
struct Node {
```

```
QueueElemType elem;
```

```
Link next;
```

```
};
```

```
Link qfront;
```

```
Link qrear;
```

```
}
```

11. Cpp file

Function 11.1 Pass by ref

```
template < class QueueElemType >
```

```
bool Queue < QueueElemType > enqueue ( QueueElemType & e )
```

```
{ class & I & template < Link > b = & p; } = T
```

```
link addedNode;
```

```
(1) addedNode = new Node;
```

```
if (addedNode == Null)
```

```
return false;
```

```
(2) addedNode->element = e;
```

```
(3) addedNode->next = Null;
```



Sat	Sun	Mon	Tue	Wed	Thu	Fri
-----	-----	-----	-----	-----	-----	-----

Page / /
Date / /

```

(4) if (qfront == Null)
    qfront = addedNode; // Empty queue
else
    qrear->next = addedNode; // General Case
}
list *j;
queue *front;
Note

```

(5) qrear = addedNode;
return true;
}

```

main()
{
    Queue <int> iQueue;
    int i, j;
    for (i=0; i<4; i++)
        cin >> j; // queue ناتج از این خروجی
    flag = iQueue.enqueue(j);
    while (!iQueue.isEmpty())
    {
        j = iQueue.dequeue();
        cout << j
    }
    j = iQueue.dequeue();
    cout << j
    j = iQueue.front();
    cout << j
    j = iQueue.dequeue();
    cout << j
    enqueue(10);
}

```

نفرض این داده ها در قوامی $15 \leftarrow 6 \leftarrow 5 \leftarrow 10 \leftarrow$ قوامی $15 \leftarrow 6 \leftarrow 5 \leftarrow 10$ باشند

(2) ایک ایک (1) و مردہ مارے (1) کر کر 15 \leftarrow 6 \leftarrow 5 \leftarrow 10 طبع output 11 (1) is front() ہے اور queue 11 سے بچھوئی 15 \leftarrow 6 \leftarrow 10 طبع output 11 (2) is 5 ہے اور dequeue کیا ہے، queue 11 سے بچھوئی 5 ہے اسے 5 پہنچا دیا جائے تو enqueue لے 15 \leftarrow 6 \leftarrow 10 سے queue 11 سے بچھوئی 10 ہے اسے 10 پہنچا دیا جائے فیسا 10 \leftarrow 15 \leftarrow 6 \leftarrow 10 queue 11 سے بچھوئی