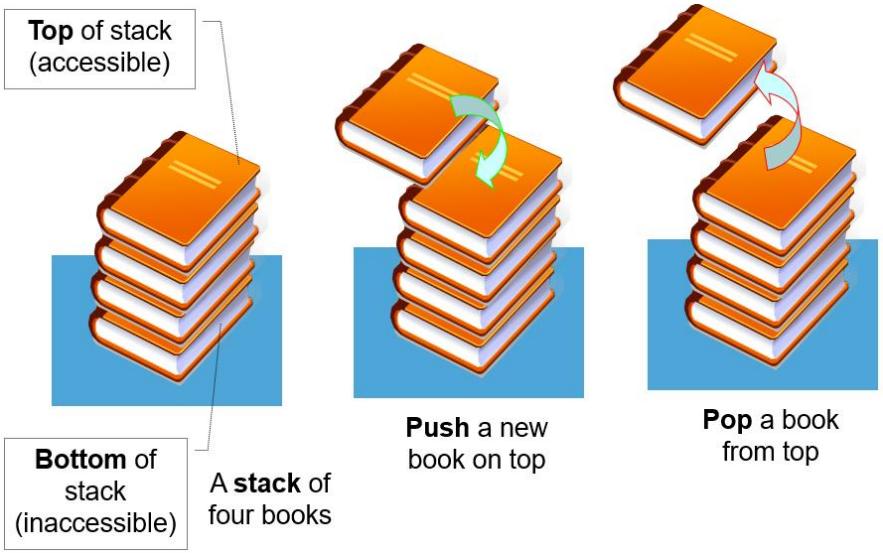
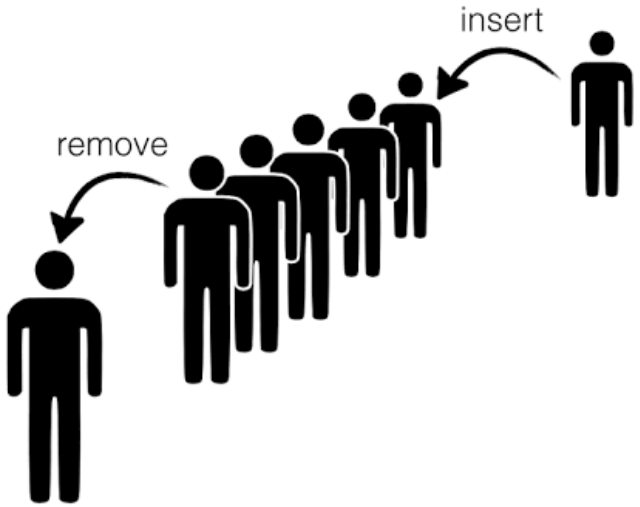
ساختمان‌های داده

Data Structures

پشته و صف

Stack and Queue



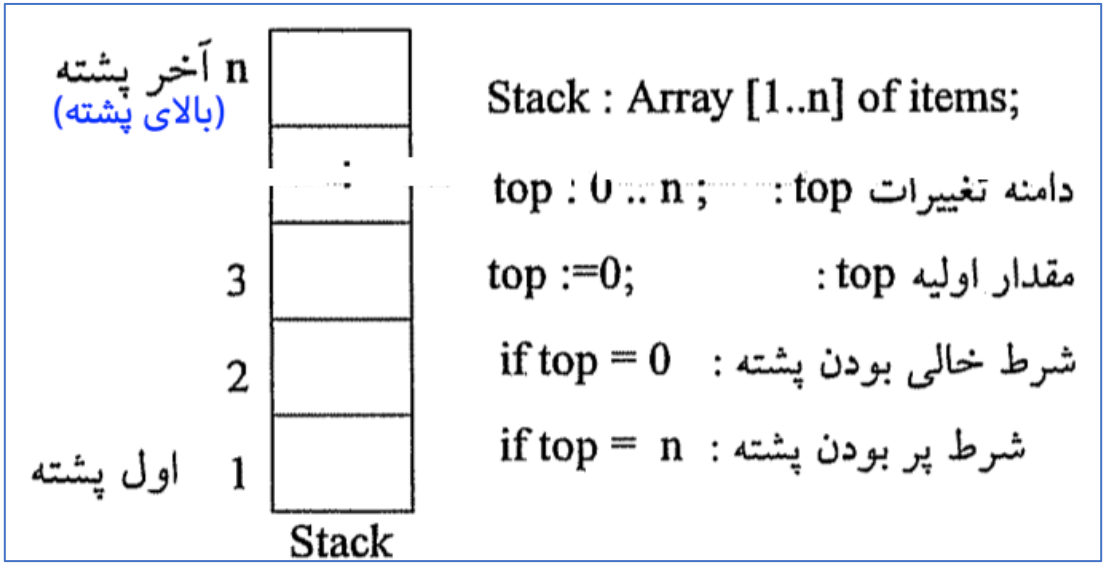


پشته (Stack)

- پشته، لیست مرتبی است که عملیات اضافه و حذف در آن *از یک ظرف* انجام می شود.
- پشته به صورت Last In First Out (LIFO) است.
  - یعنی آخرین عنصری وارد شده، اولین عنصری است که خارج می شود.
- همانند گذاشتن کتاب یا بشقاب بر روی همدیگر است.
- از کاربردهای پشته، ذخیره سازی آدرس بازگشت و ساخت متغیرهای محلی در صدا زدن توابع می باشد.
- ساده ترین راه نمایش پشته، استفاده از آرایه یک بعدی است.

پشته

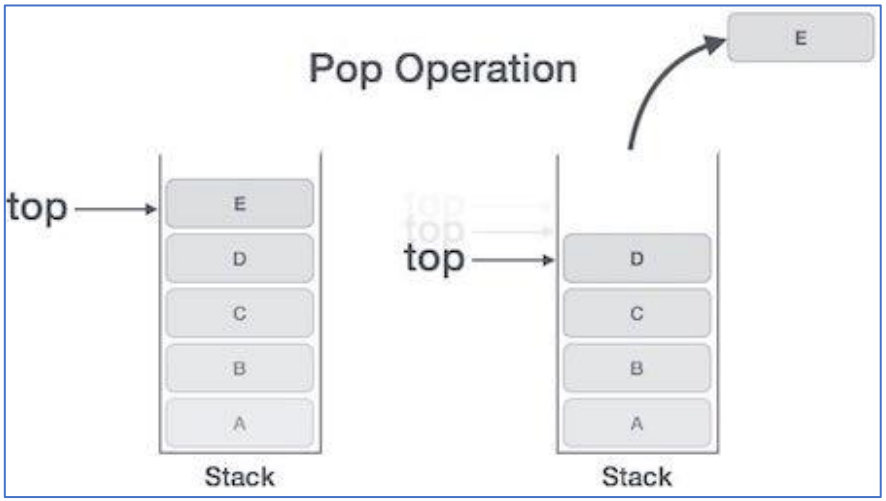
- خانه های آرایه پشته اغلب از 1 تا n شماره گذاری می شوند.
- متغیر top *اشاره به عنصر بالایی* پشته دارد.
- Top از 1 تا n تغییر می کند.
- در ابتدای کار پشته، top=0 است.
- Item نوع داده است.
  - مثلا int

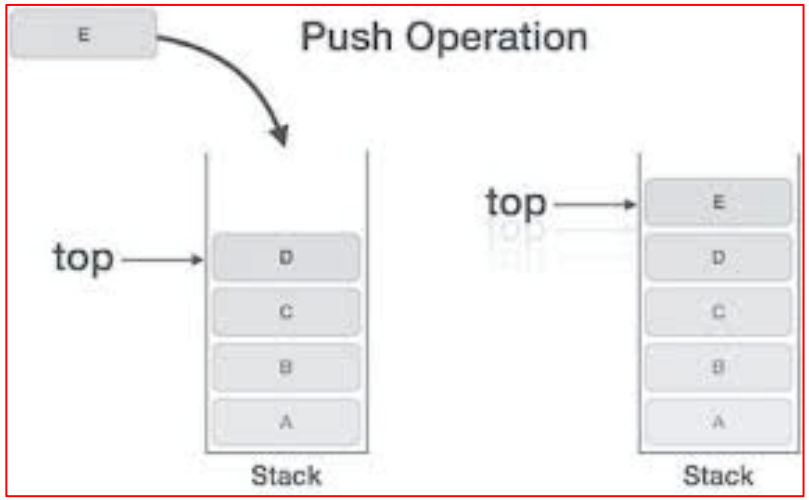


پشته

❑در پشته دو عمل قابل انجام است:

- خواندن و حذف از پشته: **pop**
- اضافه کردن یا نوشتن در پشته: **push**





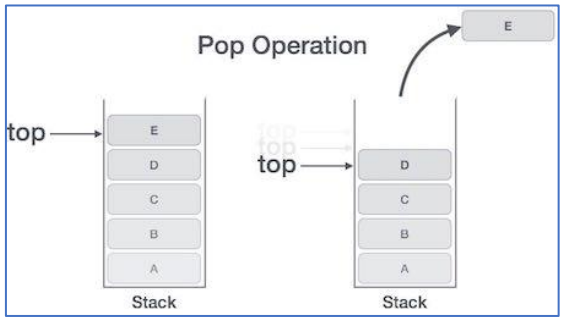
پشته

❑کدهای دستورات مرتبط با پشته

- خواندن و حذف از پشته: **pop**
- اضافه کردن یا نوشتن در پشته: **push**

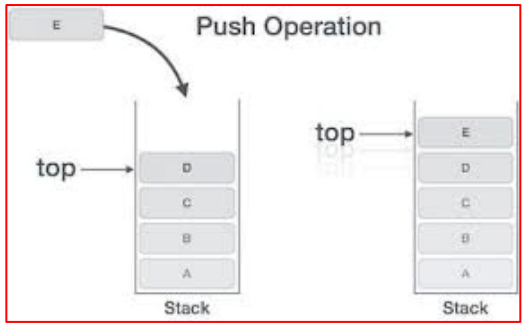
خروجی

```
pop(var K:items);
{
    if top=0 then
    write('stackempty')
    else {
        K:=stack[top];
        top:=top-1;
     }
}
```



ورودی

```
push (K:items);
{
    if top = n then
     write ('stackfull')
    else  {
        top:=top+1;
        stack[top]:=K;
     }
}
```

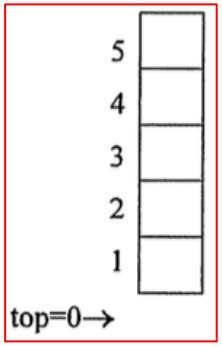


پشته

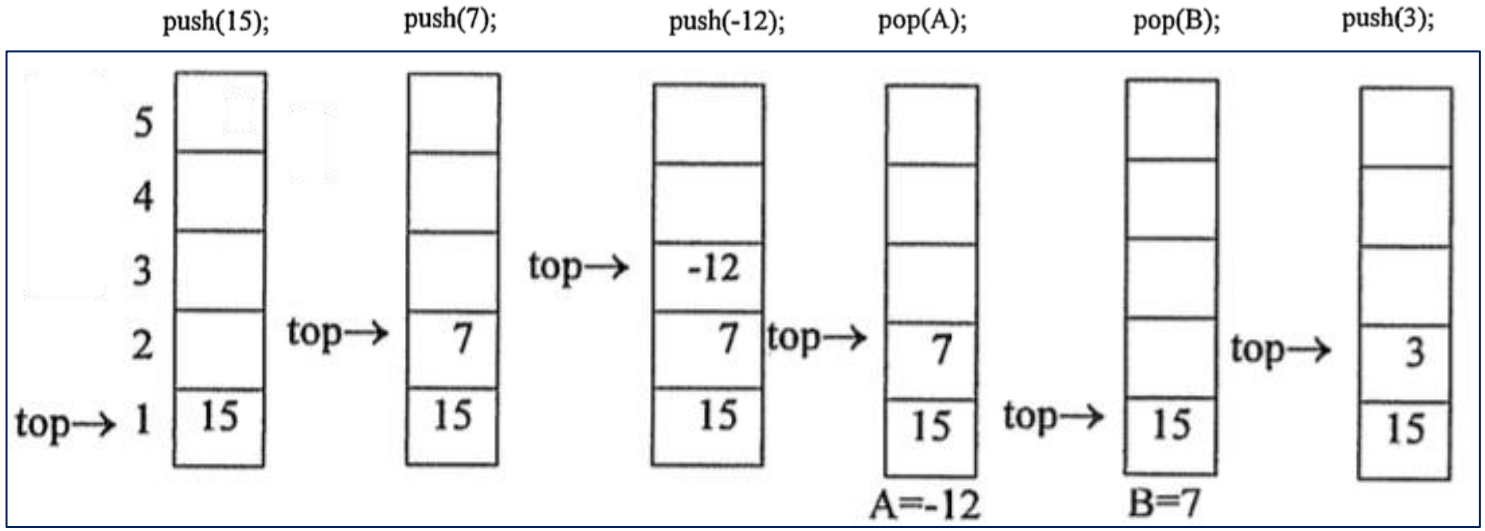
- مثال : عملیات زیر را به ترتیب (از چپ به راست) روی شکل نشان دهید. (n=5)

push(15); push(7); push(-12); pop(A); pop(B); push(3);

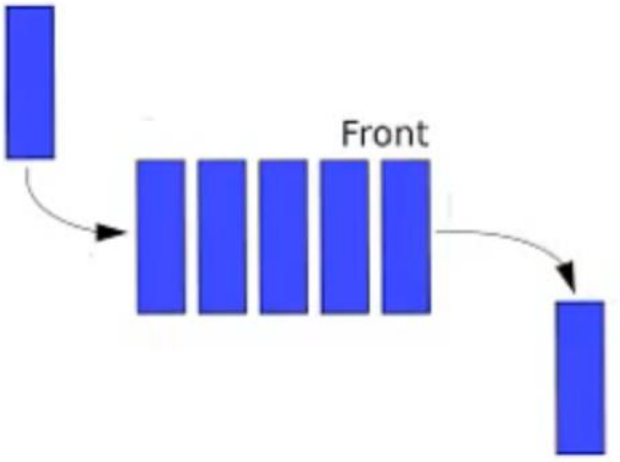
- در لحظه شروع پشته به صورت زیر است:

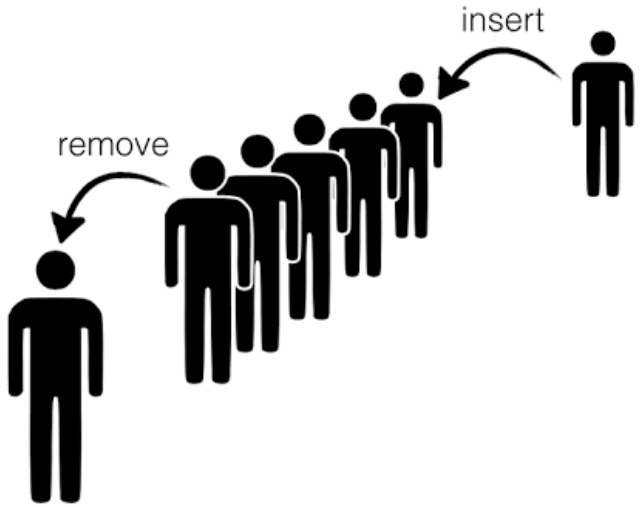


- مراحل عملیات:



صف
(queue)





صف (queue)

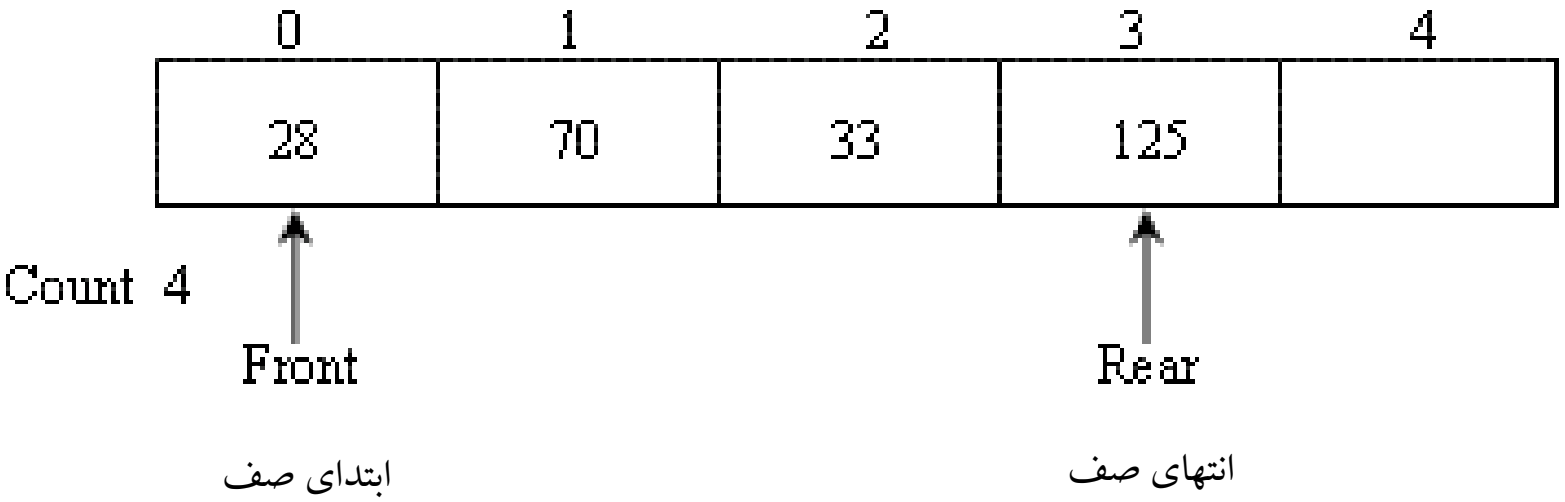
- صف، لیست مرتبی است که:
  - عمل اضافه کردن (نوشتن) از یک طرف آن به نام انتهای صف (rear)
  - عمل خواندن (حذف کردن) از طرف دیگر آن به نام ابتدای صف (front) انجام می شود.
- صف به صورت FIFO (First In First Out) می باشد
  - یعنی اولین عنصر وارد شده، اولین عنصر خارج شده می باشد.

صف (queue)

- یکی از کاربردهای مهم صف در زمانبندی برنامه ها در سیستم عامل است.
- ساده ترین راه نمایش صف استفاده از آرایه یک بعدی به طول n می باشد.
  - Queue[0..n-1]
- برای کار با صف معمولی به دو اشاره گر نیاز داریم:
  - Front: که همیشه به عنصر قبل از عنصر ابتدایی اشاره می کند
  - Rear:که همیشه به آخرین عنصر اشاره دارد

(queue) **صف**

- پیاده سازی صف با آرایه



صف (queue)

❑مثال : یک صف با آرایه با ۴ عنصر q[0..3] (اندیس ها از صفر شروع شده اند)

- که در ابتدا خالی است.

front = -1 , rear = -1

front = -1 , rear = 0 درج A

front = -1 , rear = 1 درج B

front = -1 , rear = 2 درج C

front = 0 , rear = 2 حذف A

front = 1 , rear = 2 حذف B

front = 2 rear = 2 حذف C

صف (queue)

- نوشتن و خواندن از صف: (اگر اندیس ها از ۱ شروع شوند تا n)
  - delq خواندن (حذف) از صف (با نام **dequeue** هم شناخته می شود)
  - addq اضافه کردن (نوشتن) در صف (با نام **enqueue** هم شناخته می شود)

```
addq (K:items);
{
        if rear = n then
        write ('queuefull')
        else {
                rear:=rear+1;
               q[rear]:=K;
         }
}
```

```
delq (var K:items);
{
        if front = rear then
         write ('queueempty')
        else {
                front:=front+1;
                k:=q[front];
         }
}
```

صف (queue)

- نوشتن و خواندن از صف: (اگر اندیس ها از 0 شروع شوند تا n-1)

تذکر: در صف پر مقدار rear برابر n-1 می باشد.

تذکر: در صف خالی مقدار front با rear برابر است.

```
addq (rear , item)
{
  if (rear == n-1)
       {
        queue-full( );
        return;
       }
   rear= rear+1;
   queue[rear] = item;
}
```

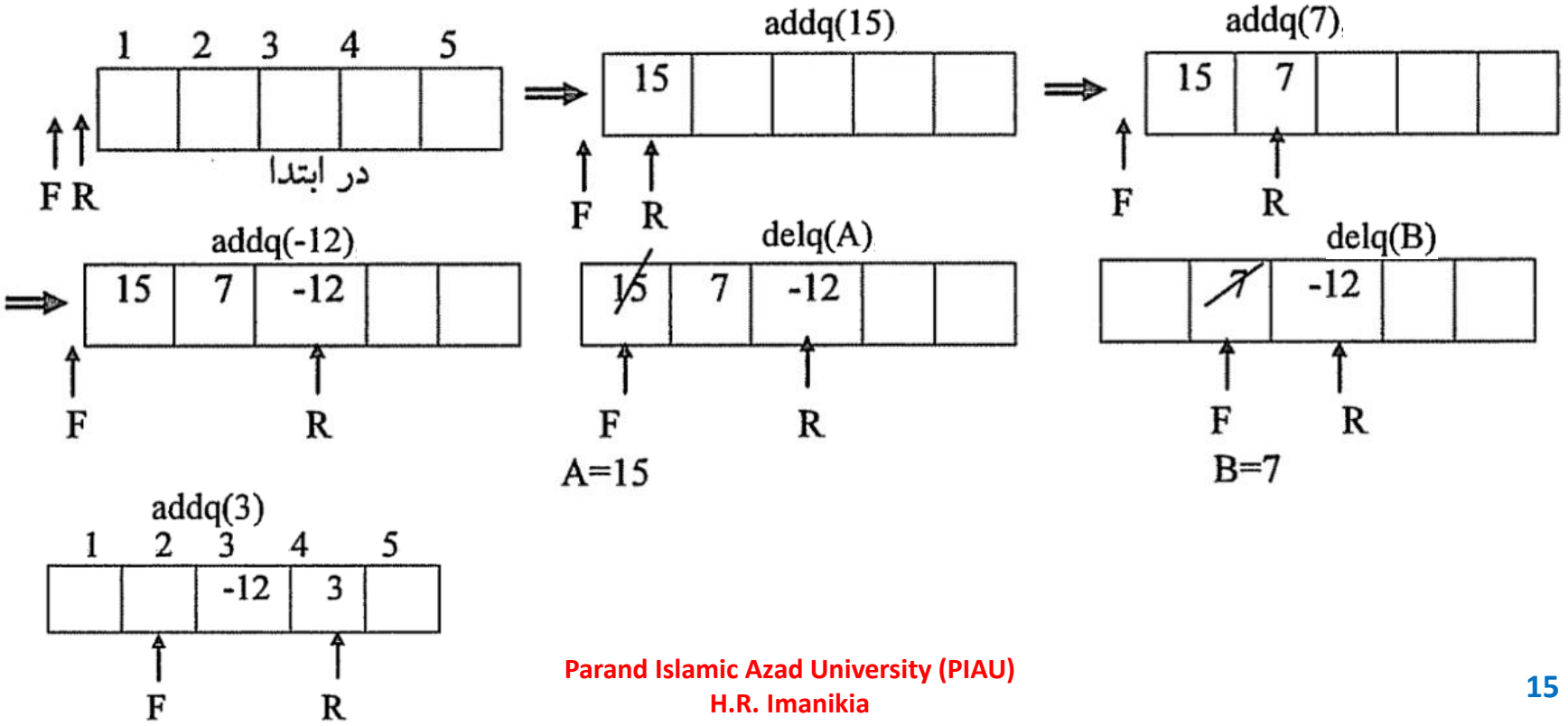
```
delq (front , rear)
{
  if (front == rear)
       return queue-empty( );
  return   q[++front] ;
}
```

صف

- مثال : عملیات زیر را به ترتیب (از چپ به راست) روی شکل نشان دهید. (n=5)

addq(15), addq(7), addq(-12), delq(A), delq(B), addq(3)

- مراحل عملیات:



صف

- تعداد عناصر موجود در صف : R-F
- تعداد خانه های خالی : n- (R-F)
- همواره $F \leq R$

صف حلقوی یا چرخشی (Circular Queue)

- مشکل اصلی صف (معمولی) : فقط یکبار قابل استفاده است.
  - با رسیدن Rear به انتها دیگر چیزی نمی توان در صف ذخیره کرد.

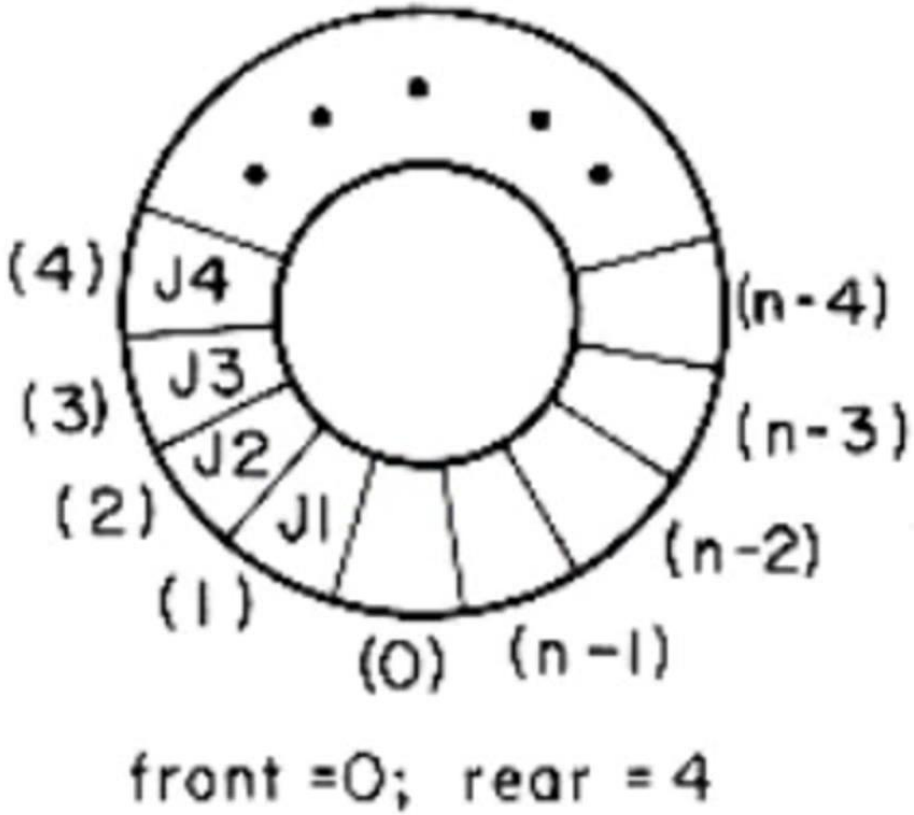
- رفع مشکل: استفاده از صف حلقوی

صف حلقوی یا چرخشی (Circular Queue)

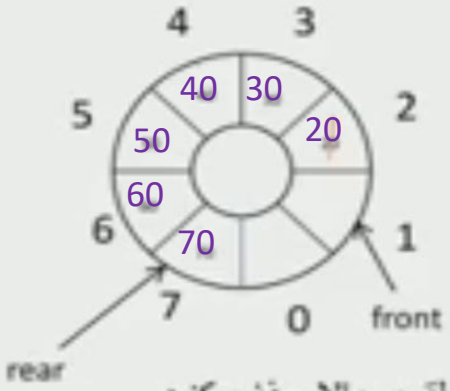
- در صف چرخشی هم مانند صف معمولی:
  - Front : به یک خانه قبل از اولین عنصر (ابتدای صف) اشاره دارد. (در برخی پیاده سازی ها به خود ابتدای صف اشاره دارد)
  - Rear : به آخرین عنصر (انتهای صف) اشاره دارد.
- حرکت در جهت عقربه های ساعت
- *در صف چرخشی با n عنصر، فقط از n-1 آن استفاده می شود. (یک خانه خالی خواهد ماند)*
  - اگر از آن یک خانه هم استفاده شود در آنصورت Front=Rear می شود و نمی توانیم صف خالی را از صف پر تشخیص دهیم.

صف حلقوی یا چرخشی (Circular Queue)

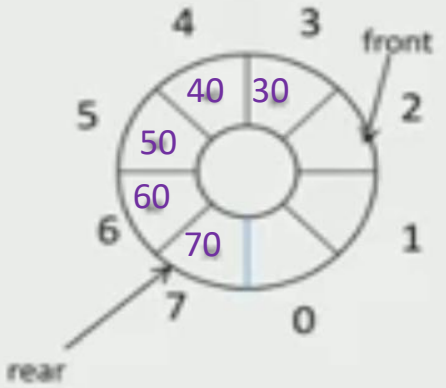
- ابتدا و انتهای صف چرخشی(مانند صف معمولی)
  - Front: یکی قبل از اولین عنصر (ابتدای صف)
  - Rear : اشاره به خود عنصر آخر (انتهای صف)



صف حلقوی یا چرخشی (Circular Queue)



عدد ۲۰ را از صف حلقوی بالا حذف کنید



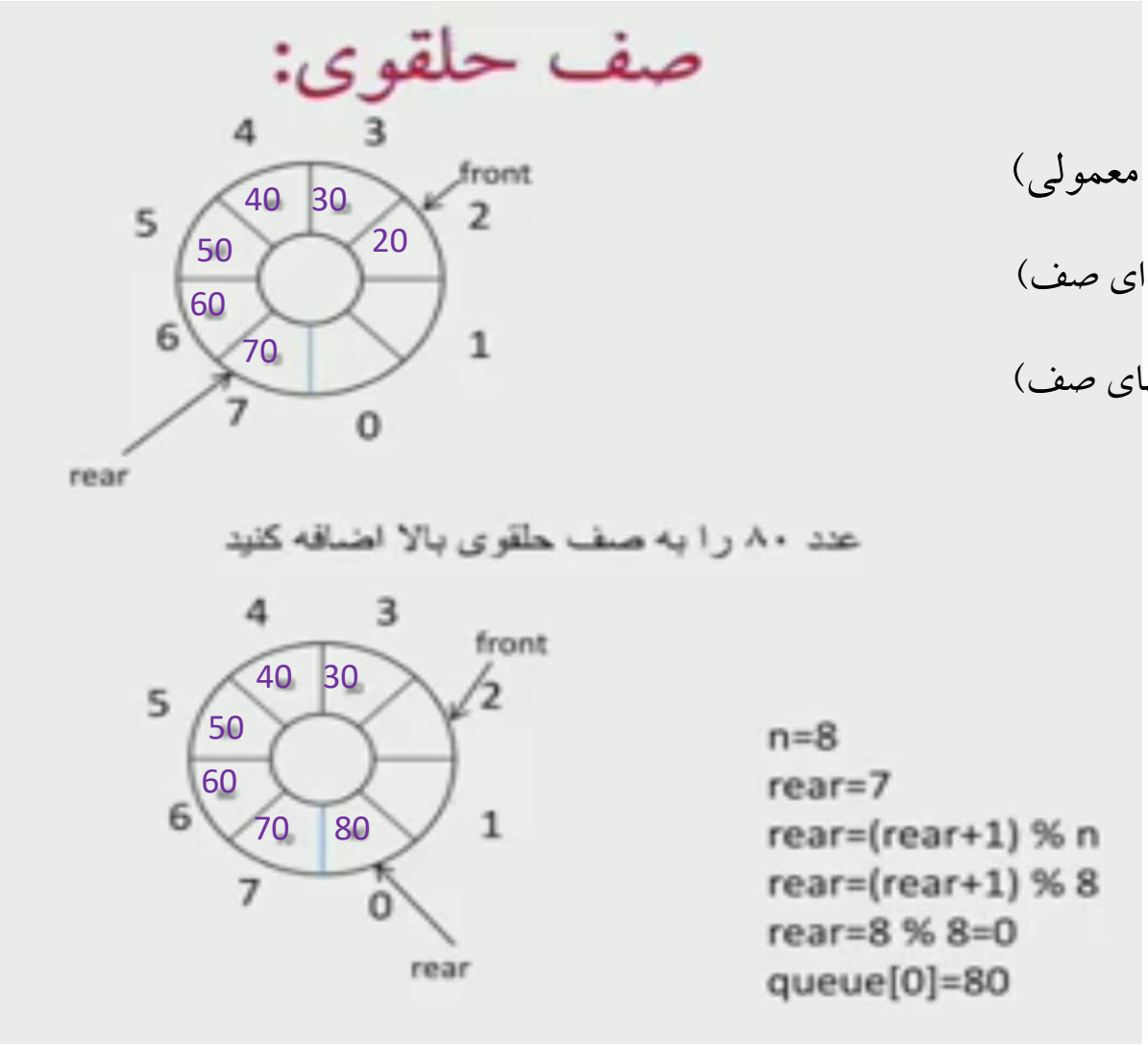

```
n=8
front=1
front=(front+1) % n
front=(front+1) % 8
front=2%8=2
return  queue[2];
```

❑ حذف از صف حلقوی

❑ ابتدا و انتهای صف چرخشی(مانند صف معمولی)

- Front: یکی قبل از اولین عنصر (ابتدای صف)
- Rear : اشاره به خود عنصر آخر (انتهای صف)

صف حلقوی یا چرخشی (Circular Queue)



- اضافه کردن به صف حلقوی
- ابتدا و انتهای صف چرخشی(مانند صف معمولی)
  - Front: یکی قبل از اولین عنصر (ابتدای صف)
  - Rear : اشاره به خود عنصر آخر (انتهای صف)

اضافه کردن عنصری به صف چرخشی

```
addq (front , rear , item)
{
    rear = (rear+1) % n;
     if ( rear == front )
         {
            queue-full(rear);
            return;
         }
    queue[rear] = item ;
}
```

- ابتدا و انتهای صف چرخشی(مانند صف معمولی)
  - Front: یکی قبل از اولین عنصر (ابتدای صف)
  - Rear : اشاره به خود عنصر آخر (انتهای صف)

حذف کردن عنصری از صف چرخشی

- ابتدا و انتهای صف چرخشی(مانند صف معمولی)
  - Front: یکی قبل از اولین عنصر (ابتدای صف)
  - Rear : اشاره به خود عنصر آخر (انتهای صف)

```
deleteq (front , rear)
{
  if (front == rear)
      return  queue-empty( );
  front = (front+1)  %  n ;
   return   queue[front] ;
}
```

تعداد خانه های خالی و پر

- با توجه به مثال قبل به سادگی می توان به فرمول های زیر رسید:

اگر $R > F$

- تعداد خانه‌های پر $= R - F$
- تعداد خانه‌های خالی $= n - (R - F)$

اگر $F > R$

- تعداد خانه‌های خالی $= F - R$
- تعداد خانه‌های پر $= n - (F-R)$

تعداد خانه های خالی و پر

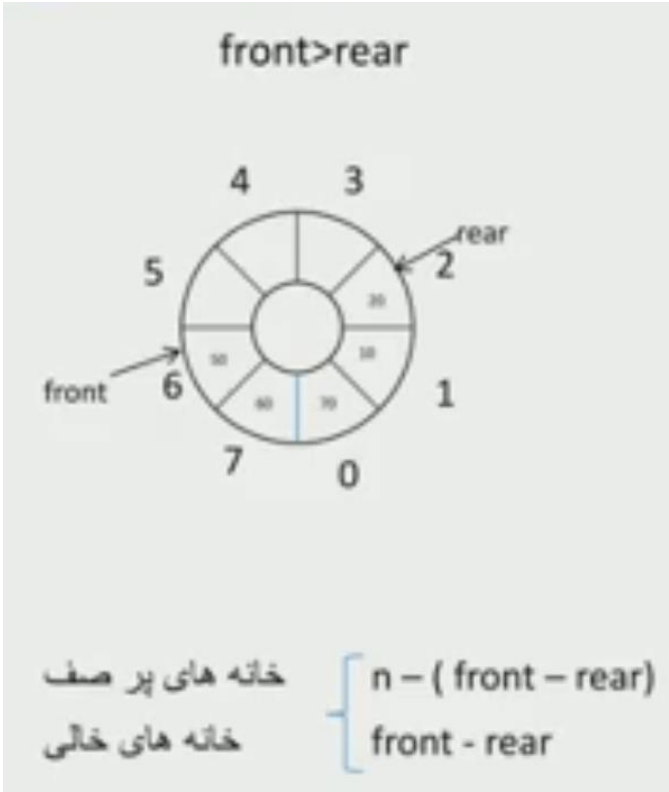
- با توجه به مثال قبل به سادگی می توان به فرمول های زیر رسید:

اگر $R > F$

- تعداد خانه‌های پر $= R - F$
- تعداد خانه‌های خالی $= n - (R - F)$

اگر $F > R$

- تعداد خانه‌های خالی $= F - R$
- تعداد خانه‌های پر $= n - (F-R)$



تمرین

۱) اگر اعداد 3,2,1 به ترتیب در پشته قرار داشته باشند (3 بالاترین) و عملیات Pop(x) بـه معنـای قـرار گرفتن بالاترین عنصر پشته در متغیر x و عملیات Push(x) به معنای ذخیره x در پشته باشد پس از انجـام عملیات زیر (از راست به چپ) وضعیت پشته چگونه است؟

Pop(x), Push(z), Push(x), Push(y), Pop(z), Push(x), Pop(y), Pop(x)