ساده ترین داده ساختارها

فرشته دهقانی

فهرست مطالب

- داده ساختارها
- صف
- پشته
- صف دو طرف
- لیست پیوندی

داده ساختارها

- ساختارهای مورد استفاده برای ذخیره‌ی ساختارمند داده ها
- به نحوی که در پاسخ دهی به سوالات ما کارا و سریع
- اگر چیزی که این داده ساختار مدلی از آن است در حال تغییر سریع باشد، باید در تطابق دادن خود با آن نیز چابک باشند.

- وابسته به سوالاتی که ما از یک سری داده ها داریم یا میزان تغییر در داده ها (مثلا یا حذف شوند یا اضافه شوند و یا آیا اصولا حذف و اضافه ای رخ میدهد)، میزان تغییر از چه نوعی باشد (مثلا آیا تنها کوچکترین عنصر حذف میشود یا هر عنصری ممکن است حذف شود)

مثال

- یک جعبه از پیچ ها با اندازه های مختلف داریم. هر بار مهره‌ای به ما میدهند تا ما چک کنیم آیا پیچ اندازه آن هست یا خیر.
- اگر برای ذخیره کردن اندازه ی این پیچ ها از یک آرایه استفاده شود، به دو روش میتوان انجام داد:
- نگهداری اندازه ها را بدون هیچ گونه ترتیب خاصی در آرایه
  - **مزیت**: اضافه کردن اندازه یک پیچ جدید به راحتی در آخر آرایه
  - **عیب**: برای پیدا کردن پیچ متناسب با مهره، چک کردن تمامی آرایه
- نگهداری اندازه ی پیچ ها را به صورت مرتب شده در آرایه
  - **مزیت**: پیدا کردن پیچ متناسب با مهره به راحتی و با استفاده از جست و جوی دودویی بسیار سریعتر و کاراتر از قبل
  - **عیب**: اضافه کردن یک پیچ جدید در آرایه بدون بهم خوردن ساختارمندی آن (مرتب بودن اعضا)،دشوار و زمان بر

ادامه

- چگونه یک سری داده را که توالی در آن ها مهم است را ذخیره کنیم؟
- در نظر بگیرید که داده ها یک به یک در حال وارد شدن به یا خارج شدن از برنامه هستند و باید آن ها را در داده ساختاری ذخیره کنیم که توالی آن ها را حفظ کند.
- بررسی داده ساختارهایی که یک توالی از عناصر را ذخیره می کنند و هر کدام برای نوع خاصی از اضافه شدن عناصر به خودشان یا حذف شدن عناصر بهینه شده اند.
  - صف
  - پشته
  - لیست پیوندی

صف
QUEUE

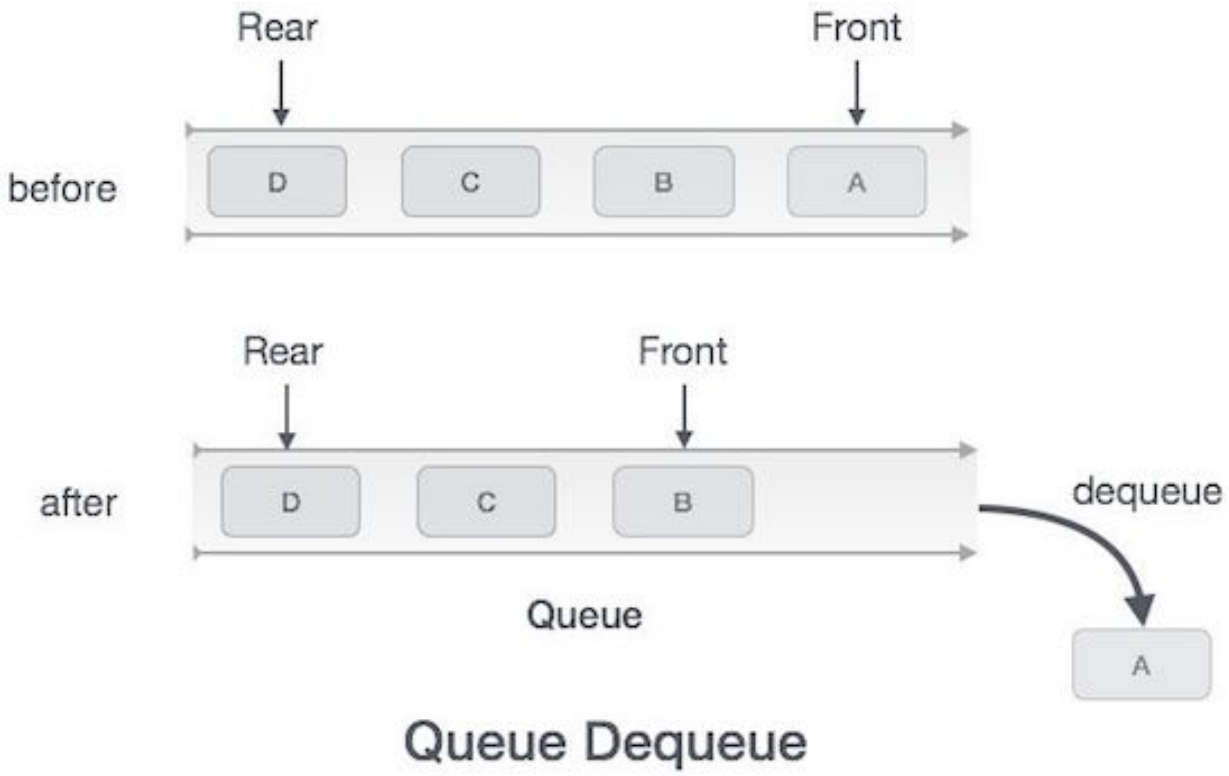
- داده ساختاری که عنصر جدید فقط در انتهای آن اضافه می شود و حذف عنصر تنها از اول آن اتفاق می افتد. (FIFO)(First In First Out)
- مثال: نانوایی

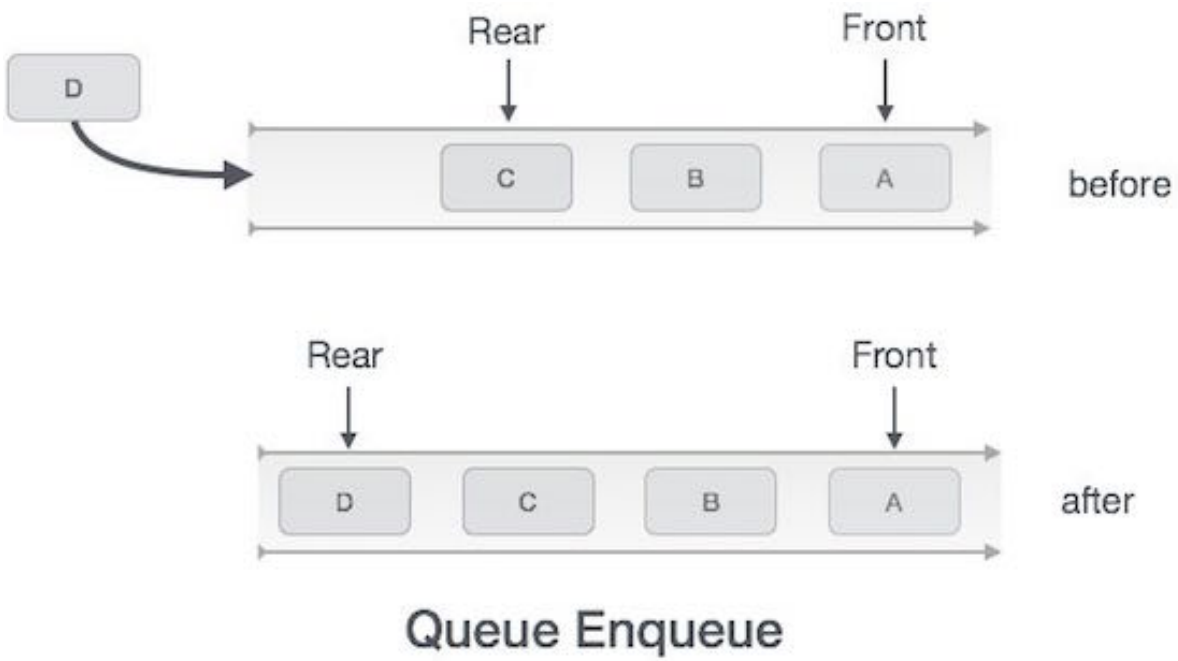
صف

❖ **پشتیبانی صف از دستورات زیر (همگی با O(1) انجام می شوند) (پیاده سازی به وسیله آرایه یا لیست پیوندی)**

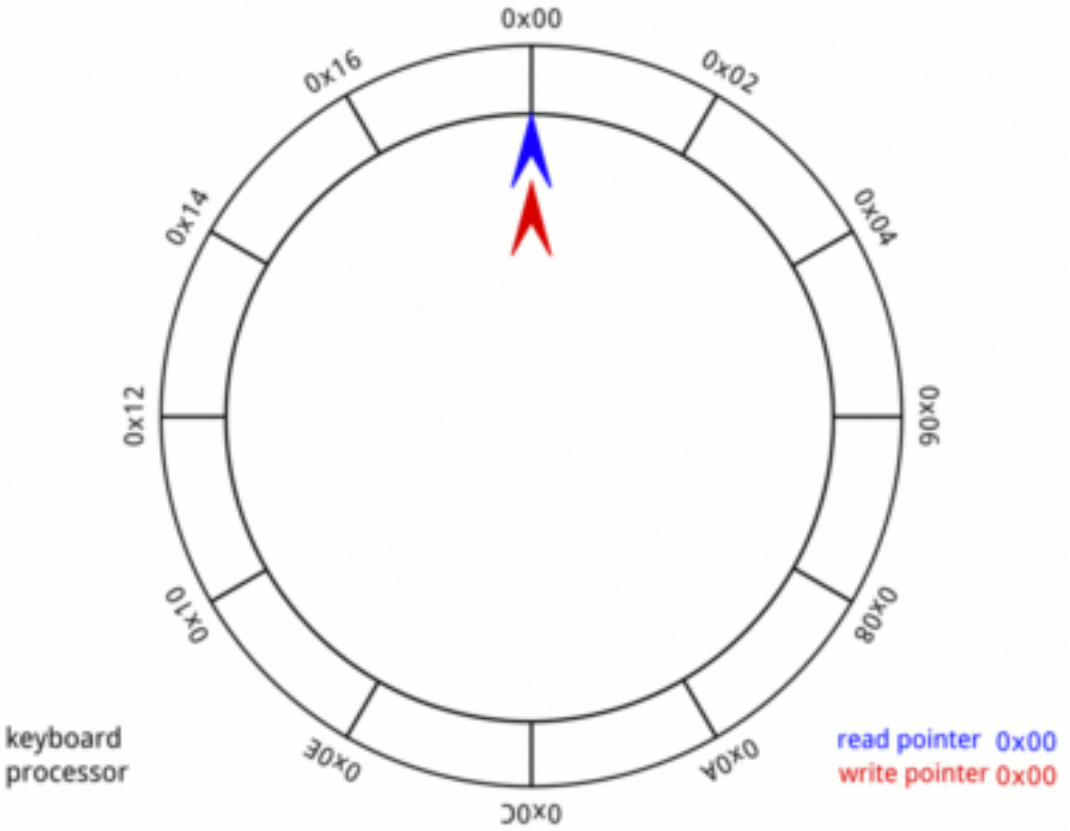
- $enqueue(x)$: عنصر x را در انتهای صف درج می کند.
- $dequeue()$:اولین عنصر را حذف می کند و آن را باز می گرداند.
- $front()$: اولین عنصر صف را باز می گرداند.
- $size()$: تعداد عناصر موجود در صف را باز می گرداند.
- $is\_empty()$: خالی بودن یا نبودن صف را مشخص می کند.
- $is\_full()$: پر بودن یا نبودن صف را مشخص می کند.

پیاده سازی صف با آرایه





CIRCULAR BUFFER



تمرین

- پیاده سازی بالا از صف این قابلیت را اضافه کنید که که عنصر iام صف را نمایش دهد.

پشته
STACK

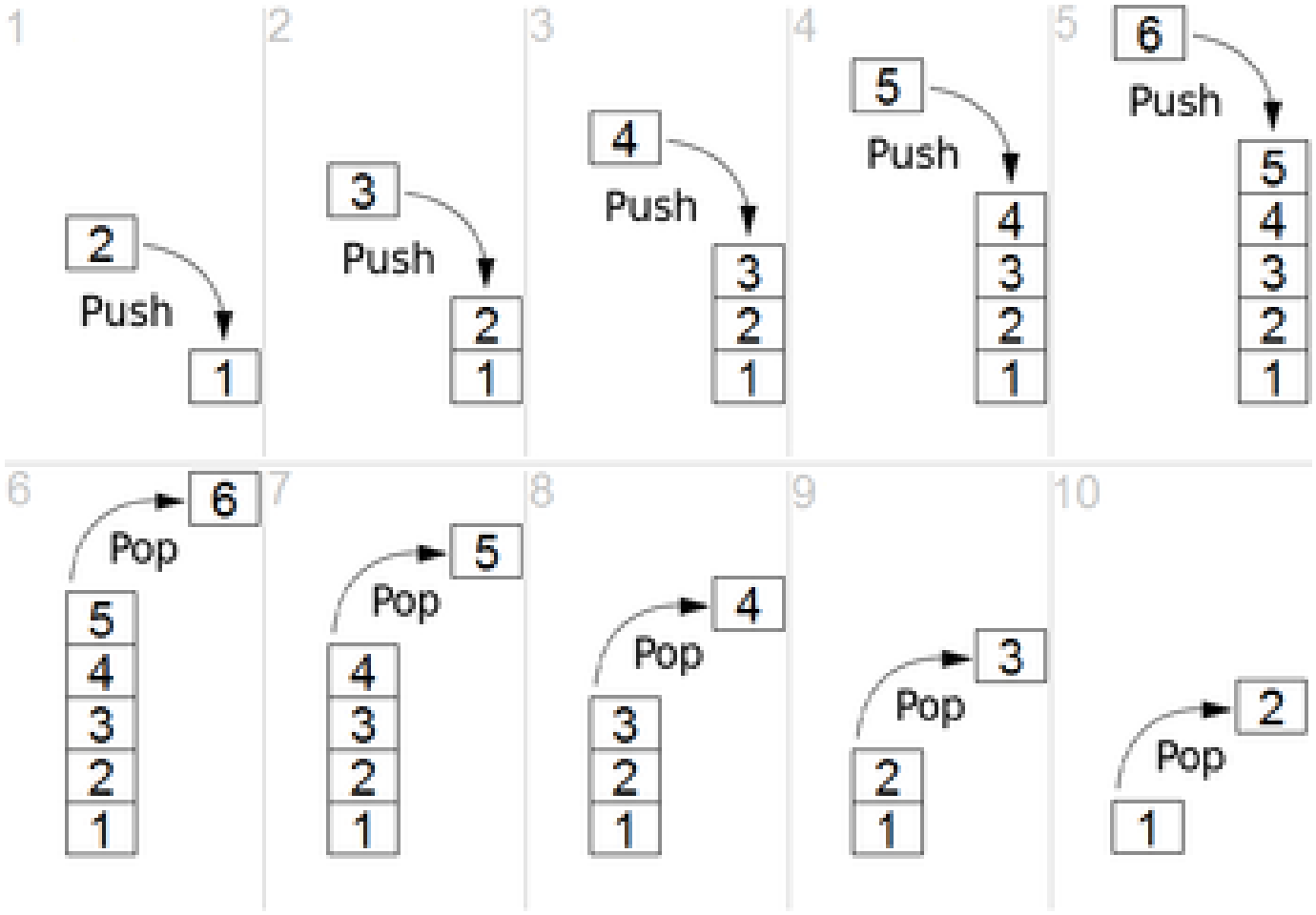
- در بعضی موارد عناصر تنها از یک طرف اضافه می شوند و از همان طرف نیز خارج می شوند.
  - مثال: ظرف های کثیفی را در نظر بگیرید که بر روی هم قرار دارند و قصد شستن آن ها را دارید در هر مرحله اگر بخواهید ظرفی را بردارید بالاترین آن ها را انتخاب می کنید و آن را می شویید و اگر ظرف کثیفی را بخواهید به آن ها اضافه کنید در بالای تمام آن ها قرار می دهید.
  - می‌توانید کارت‌ها را روی دسته کارت‌ها بگذارید یا از روی آن بردارید.
- دو کاربرد رایج از پشته‌ها در رایانه: ارزیابی عبارات و فراخوانی توابع
- در واقع در این موارد عنصری که دیر تر از همه اضافه شده است زودتر از همه خارج میگردد.(Last In First Out) (LIFO)
- به داده ساختاری که این گونه موارد را مدل می کند، پشته می گویند.

عملیات در پشته با O(1)

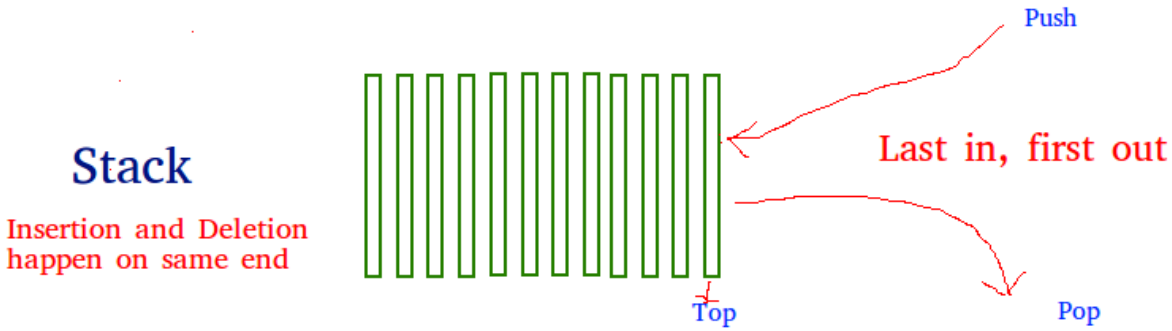
❖ **پیاده سازی پشته نیز به کمک آرایه یا لیست پیوندی انجام میشود**

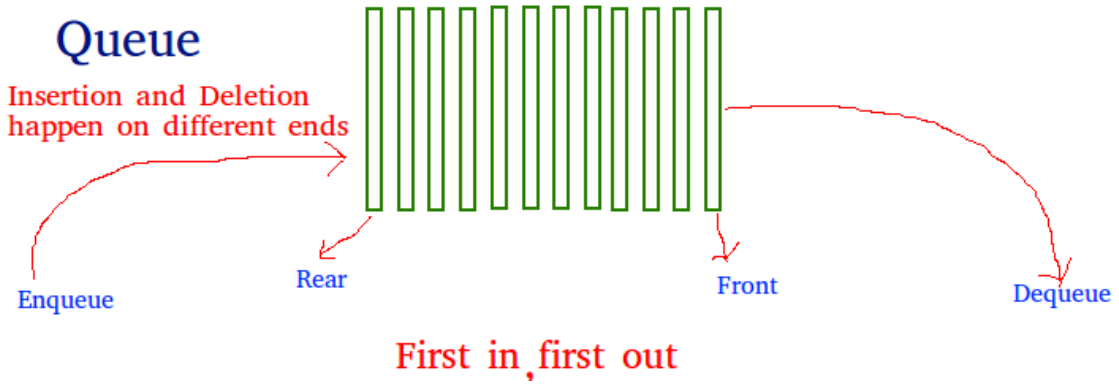
- $push(x)$: x را به بالای پشته اضافه می کند.
- $pop()$: عنصر بالای پشته را حذف می کند و آن را بازمی گرداند.
- $top()$: عنصر بالای پشته را باز می گرداند.
- $size()$: تعداد عناصر موجود در پشته را باز می گرداند.
- $is\_empty()$: خالی بودن پشته را مشخص می کند.
- $is\_full()$: پر بودن پشته را مشخص می کند.

مثال پشته


1
2
Push
1
2
3
Push
2
1
3
4
Push
3
2
1
4
5
Push
4
3
2
1
5
6
Push
5
4
3
2
1
6
6
Pop
5
4
3
2
1
7
5
Pop
4
3
2
1
8
4
Pop
3
2
1
9
3
Pop
2
1
10
2
Pop
1

مقایسه پشته و صف





صف دوطرفه
DOUBLE ENDED QUEUE

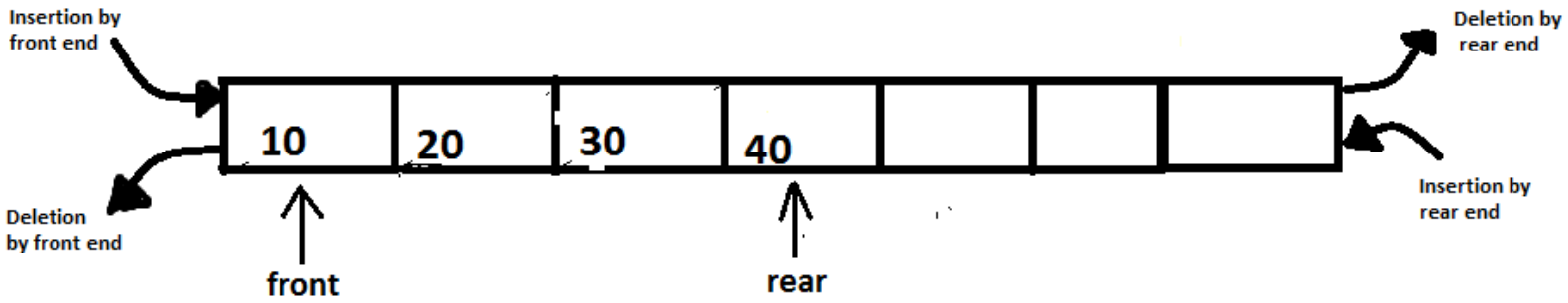
- صف عادی محدودیت هایی برای مدلسازی دارد.
  - برای مثال فرض کنید که داریم صف نانوایی را مدل میکنیم اما نانوا آشناهایش را در ابتدای صف قرار می دهد! برای همین لازم است که گاهی به سر صف هم عنصر اضافه شود
  - یا حتی گاهی لازم است که از ته صف یک عنصر حذف شود(برای مثال یک نفر که حوصله ی صبر کردن ندارد چند نفر از آخر صف را منصرف می کند و سپس خودش یواشکی! به آخر صف اضافه می شود).
- در واقع صف دو طرفه قابلیت های پشته و صف را به صورت همزمان پشتیبانی می کند.

عملیات در صف دو طرفه با O(1)

پیاده سازی صف دو طرفه نیز به کمک آرایه یا لیست پیوندی انجام میشود

- $push\_front(x)$: عنصر x را در ابتدای صف درج می کند.
- $push\_back(x)$: عنصر x را در انتهای صف درج می کند.
- $pop\_front(x)$: عنصر ابتدای صف را حذف می کند و ان را باز می گرداند.
- $pop\_back(x)$: عنصر انتهای صف را حذف و آن را باز می گرداند.
- $front()$: عنصر ابتدای صف را باز می گرداند.
- $back()$: عنصر انتهای صف را باز می گرداند.
- $size()$: تعداد عناصر موجود در صف را باز می گرداند.
- $is\_empty()$: خالی بودن یا نبودن صف را مشخص می کند.
- $is\_full()$: پر بودن یا نبودن صف را مشخص می کند.

صف دو طرفه

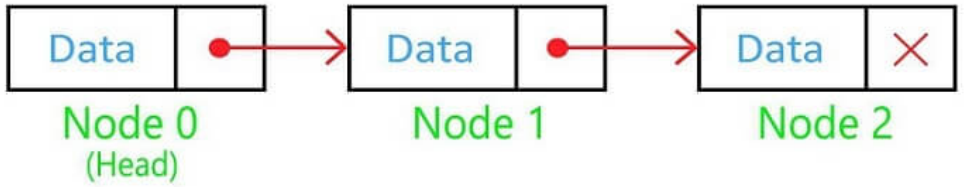


Doubly ended queue

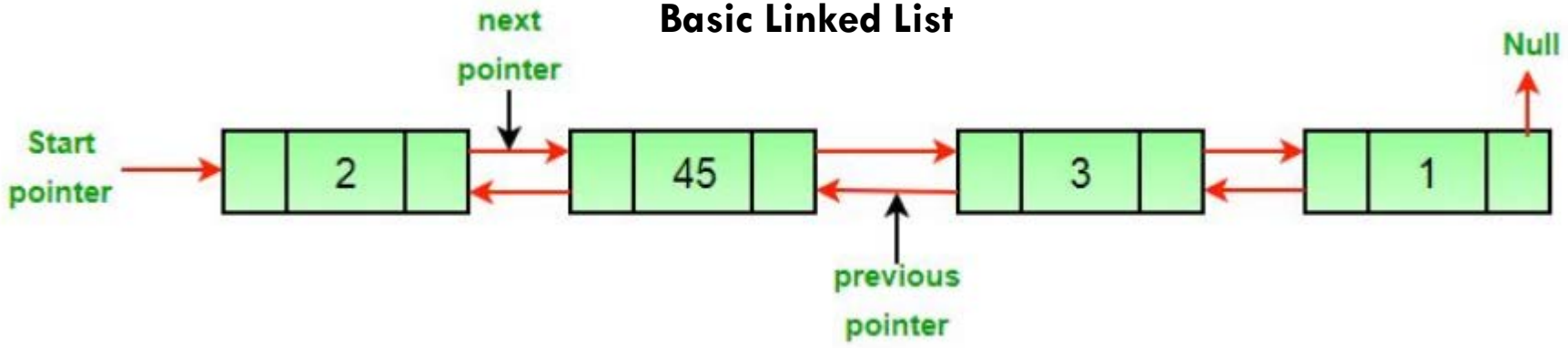
لیست پیوندی
LINKED LIST

- داده ساختاری بسیار پایه ای برای نگهداری توالی
- در لیست پیوندی هر عنصر تنها از عنصر بعدی خود مطلع است. (لیست پیوندی دوطرفه، هر عنصر علاوه بر عنصر قبلی خود از عنصر بعدی خود نیز مطلع است)
- در لیست پیوندی ساده، عنصر قبل از عنصر اول، و عنصر بعد از عنصر آخر نداریم
- در لیست پیوندی حلقوی، عنصر بعد از عنصر آخر عنصر اول است و عنصر قبل از عنصر اول عنصر آخر است.

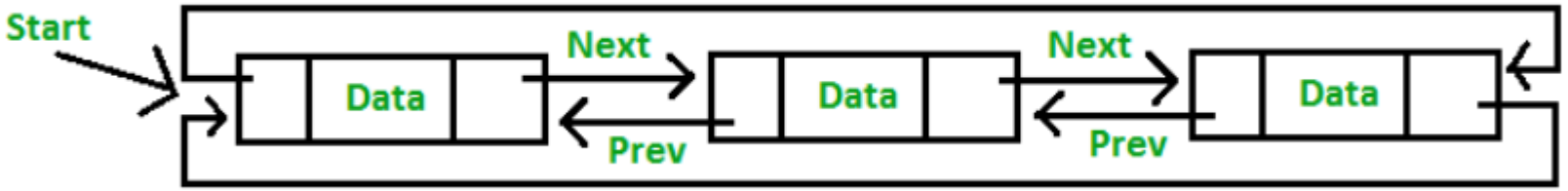
انواع متداول لیست پیوندی



Basic Linked List



Doubly Linked List

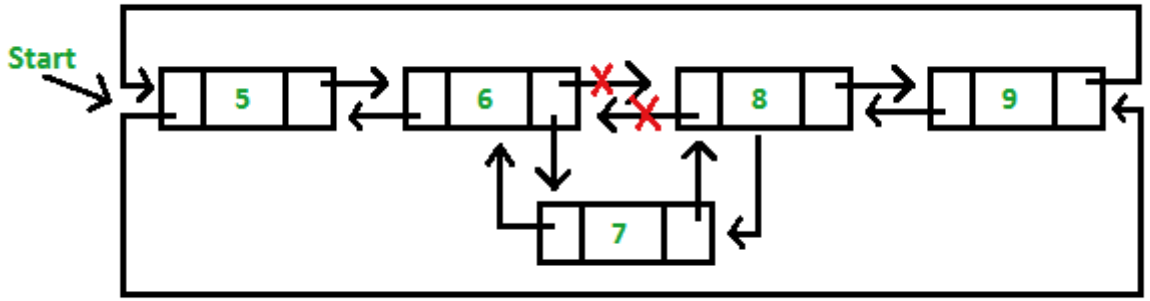


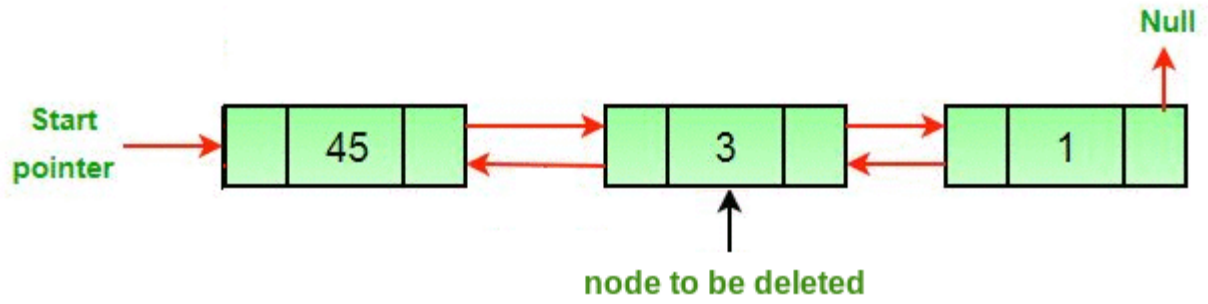
Doubly Circular Linked List

عملیات در لیست پیوندی

- $insert\_after(data, x)$:عنصر $data$ را پس از عنصر x درج می کند.
- $delete(x)$:عنصر x را حذف و آن را بازمیگرداند.
- $find(val)$: اولین عنصر با $data$ مساوی با val را باز می گرداند.
- $get(ind)$: عنصر ind ام لیست را باز میگرداند.
- $size()$: تعداد عناصر موجود در لیست را باز می گرداند.
- $is\_empty()$: خالی بودن یا نبودن لیست را مشخص می کند.

نحوه درج و حذف در لیست دوطرفه حلقوی





مقایسه آرایه و لیست پیوندی

- درج در یک مکان آرایه: O(n)
  - زیرا عناصر قبل از مکانی که عنصر جدید میخواهد درج شود باید به عقب و عناصری که بعد از مکانی که عنصر جدید میخواهد در آنجا درج شود باید به جلو تغییر مکان دهند و تعداد ان ها از O(n) است.
- درج در یک مکان مشخص از لیست پیوندی: O(1)
  - کافیست که تنها تعدادی از پیوندهای دو عنصری که درج در بین آن ها انجام میشود و پیوند های خود عنصر درج شده تغییر یابند.
- روند حذف مشابه درج
- دسترسی به عنصر iام:
- در آرایه به عنصر iام در آرایه دسترسی مستقیم : O(1)
- در لیست پیوندی باید از عنصر اول i بار جلو برویم: O(n)

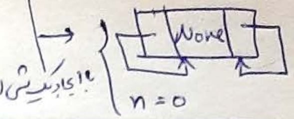
مثال لیست پیوندی

```
list = List()
```

Node { data, Next, Prev

List { head → از جنس کلاس Node
n → تعداد عناصر لیست

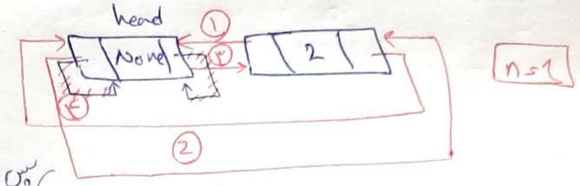
با ایجاد یک شی از جنس کلاس List، عنصر head برای آن شی ساخته می‌شود.




```
insert_after (x, data)
    y = Node(data)
    Self. n += 1        (1)
    y. prev = x         (2)
    y. next = x. next   (3)
    x. next = y         (4)
    y. next. prev = y
```

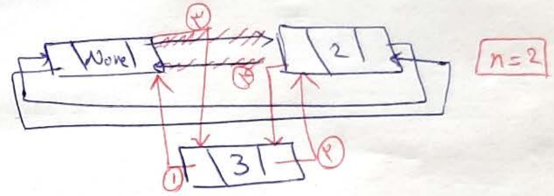
مثال: درج

insert-after (head, 2)



سپس

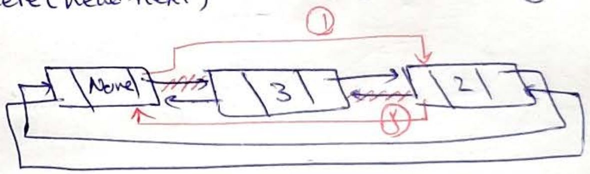
insert-after (head, 3)




```
delete (x)
    self. n -= 1
    x. prev. next = x. next   (1)
    x. next. prev = x. prev   (2)
```

مثال حذف: (از لیست بالا) ←

delete (head. next)



حالا اگه بخواهیم عنصر سوم رو به عنوان مثال حذف کنیم ←

delete (get(2))

(با در نظر گرفتن این که از صفر شماره گذاری شده)

یا اگه بخواهیم عنصری با مقدار 'zahra' رو حذف کنیم

delete (find('zahra'))

* برای درج هم مشابه حذف خواهد بود:

```
insert_after( find('zahra'), 'ahmad')
insert_after ( get(2) , 'ahmad')
```

تمرین

- تغییر تابع get به نحوی که اندیس گذاری از یک شروع شود