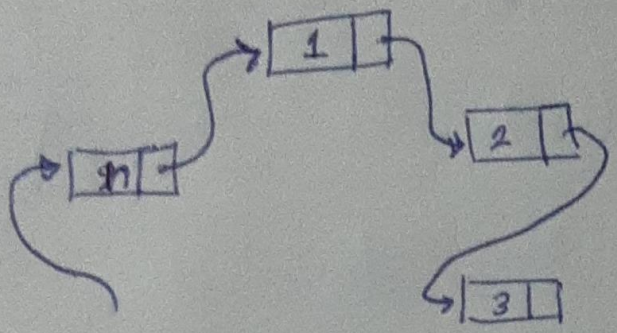
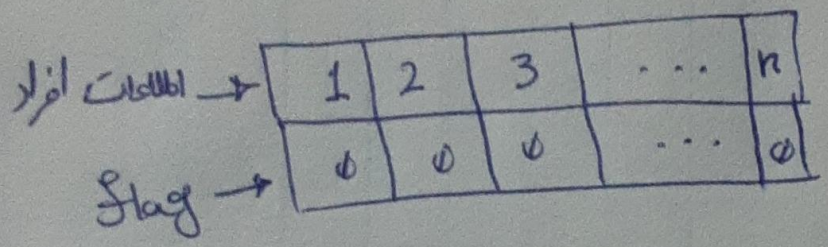
* حل به کمک لیست پیوندی به چه صورت است؟



لیست پیوندی را به کار یک فرض:

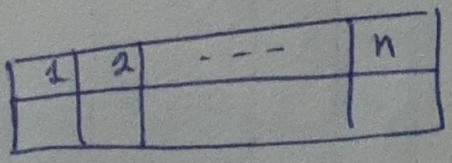
پیچیدگی زمانی $O(n)$

حذف $O(1)$، پیمایش $O(n)$



لیست:

در شروع همه عناصر صفر هستند.



تمامی خانه‌ها صفر بوده‌اند.

$n/2$

$n/4$

$\vdots$

$n/2^k \Rightarrow k = \log n$

پیچیدگی زمانی $O(n \log n)$ $\Leftarrow$

شهود پیاده‌سازی لیست پیوندی با استفاده از آرایه، (مدیریت حافظه در این نوع پیاده‌سازی با ماست)

یادآوری: هر نود لیست پیوندی دوطرفه در قالب زیر است:

prev	element	next

* پیاده‌سازی با ← یک آرایه
  ← چند آرایه

یک آرایه:

	1	2	3	4	5	6	
	$element_1$	$prev_1$	$next_1$	$element_2$	$prev_2$	$next_2$	...
	نود ۱			نود ۲			

* خالی: null یا مقدار ۰

* $prev_i$ و $next_i$ حاوی اندیس‌های مناسبی از آرایه است

سه آرایه:

	1	2	3	
prev				...
element				...
next				...

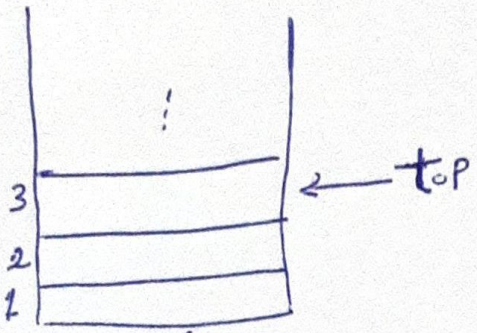
چالش‌ها: برای درج / حذف / جستجو را بررسی کنید

- درج ← نیاز به پیدا کردن خانه خالی برای درج
- حذف ← نیاز به بروزرسانی خانه‌ها برای خالی بودن
- جستجو ← نیاز به نشانه اشاره به نود اول

داده ساختار پشته (Stack):

یک داده ساختار برای نگهداری مجموعه‌های پویاست که عناصر آن در یک ترتیب خطی قرار گرفته‌اند و برای حذف عناصر در آن از سیاست "آخرین ورودی، اولین خروجی" استفاده می‌شود.

LIFO = Last In First Out



نمایش گرافیکی پشته

نکته: اگر top=0 باشد یعنی پشته خالی است.

عملیات روی پشته:

Stack-Push(S,x): یک پیمان بروزرسانی است که عنصر x را به بالای پشته S اضافه می‌کند.

```
Stack-Push(S,x)
1. S.top = S.top + 1
2. S[S.top] = x
```

پیچیدگی زمانی الگوریتم فوق O(1) است.

Stack-Pop(S): یک پیمان بروزرسانی است که عنصر بالای پشته را حذف و برمی‌گرداند. لازم به ذکر است اگر پشته تهی باشد، پیام "Under Flow" برگردانده می‌شود.

```
Stack-Pop(S)
1. if (S.top==0) then
2.         Return "underflow"
3. else
4.         S.top = S.top - 1
5.         Return S[S.top+1]
```

پیچیدگی زمانی الگوریتم فوق، $O(1)$ است.

Stack-Empty(S): یک دستور بازیابی است که تعیین می‌کند آیا پشته خالی است یا نه.

```
Stack-Empty(S)
1. if (S.top==0) then
2.         Return True
3. else
4.         Return False
```

پیچیدگی زمانی الگوریتم فوق، $O(1)$ است.

Stack-Top(S): یک دستور بازیابی است که عنصر بالای پشته را بدون تغییر در پشته برمی‌گرداند. لازم به ذکر است که اگر پشته تهی باشد پیام underflow برگردانده می‌شود.

```
Stack-Top(S)
1. if (S.top == 0) then
2.     Return "underflow"
3.   else
4.     Return S[S.top]
```

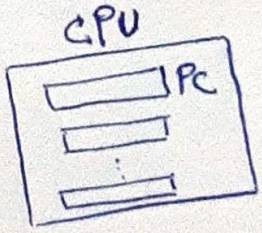
پیچیدگی زمانی الگوریتم فوق $O(1)$ است.

کاربردهای پشته:

۱) فراخوانی توابع: یکی از مهم‌ترین کاربردهای پشته است که در بخش‌های قبلی (الگوریتم‌های بازگشتی) تشریح کردیم.

یادآوری:

```
{Variables}
Function A
{
  ...
}
Function Main
{
  ...
  A()
  ...
}
```



PC: Program counter