

STACK	Queue
PUSH and POP	DeQueue and EnQueue

Good morning! Here's your coding interview problem for today.

This problem was asked by Google.

Given a stack of  $N$  elements, interleave the first half of the stack with the second half reversed using only one other queue. This should be done in-place.

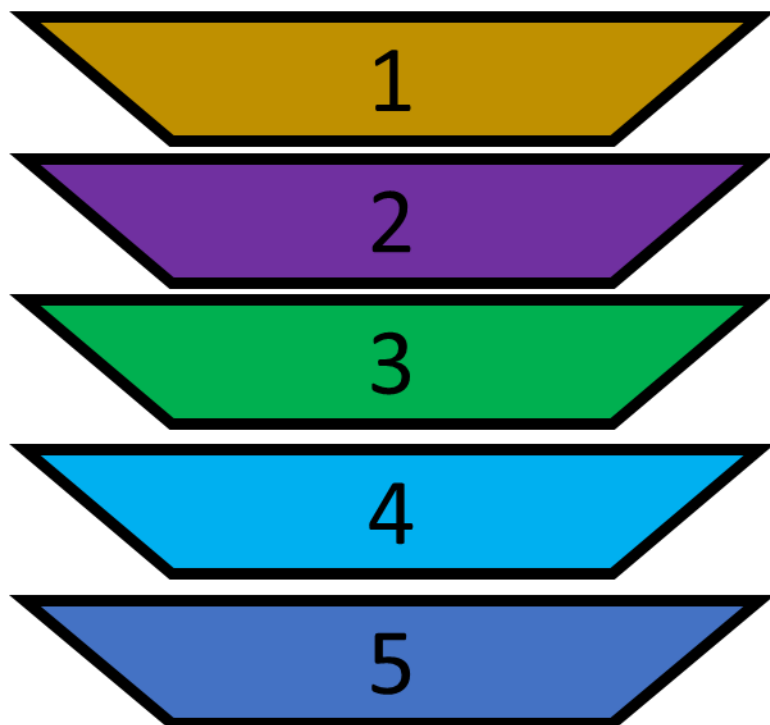
Recall that you can only push or pop from a stack, and enqueue or dequeue from a queue.

For example, if the stack is  $[1, 2, 3, 4, 5]$ , it should become  $[1, 5, 2, 4, 3]$ . If the stack is  $[1, 2, 3, 4]$ , it should become  $[1, 4, 2, 3]$ .

Hint: Try working backwards from the end state.

~~~~~  
 عقیده ی من آنست که دانشجویانم باید بتوانند مساله ی فوق را به درستی حل کنند. درست است من هنوز مفهوم صف و پشته را درس ندادم و اینکار را بعد عید میکنم. اما دانشجویان محترم صف نانوایی را دیده اند و کارکرد آن را میدانند و یا پشته ای از بشقاب ها را دیده اند و به خوبی میدانند که چگونه با آن ها کار بکنند.

در سوال بالا گفته شده یک پشته با  $n$  عنصر داریم، میکه نصف بالای پشته را با نصف پایین پشته یک در میان درهم چین کن و البته نیمه دوم رو برعکس وارد کن. عبارتی پشته ای از بشقاب های زیر را در نظر بگیرید که روی هم قرار داریم و فقط از روی پشته میتوانیم بشقاب ها را برداریم.



خب حالا می‌گه نیمه اول پشته یعنی بشقاب های ۱ و ۲ و ۳ رو فاصله بنداز و نیمه دوم رو بینشون برعکس درج کن.

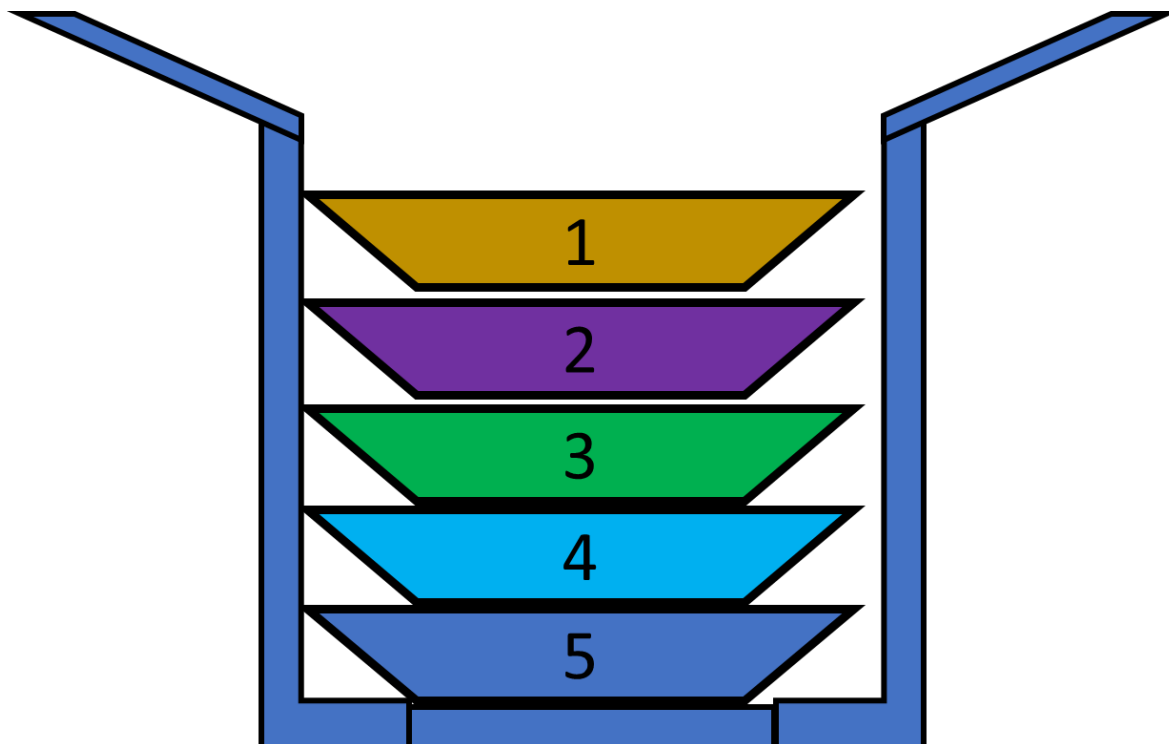
- نیمه اول فاصله بنداز ...



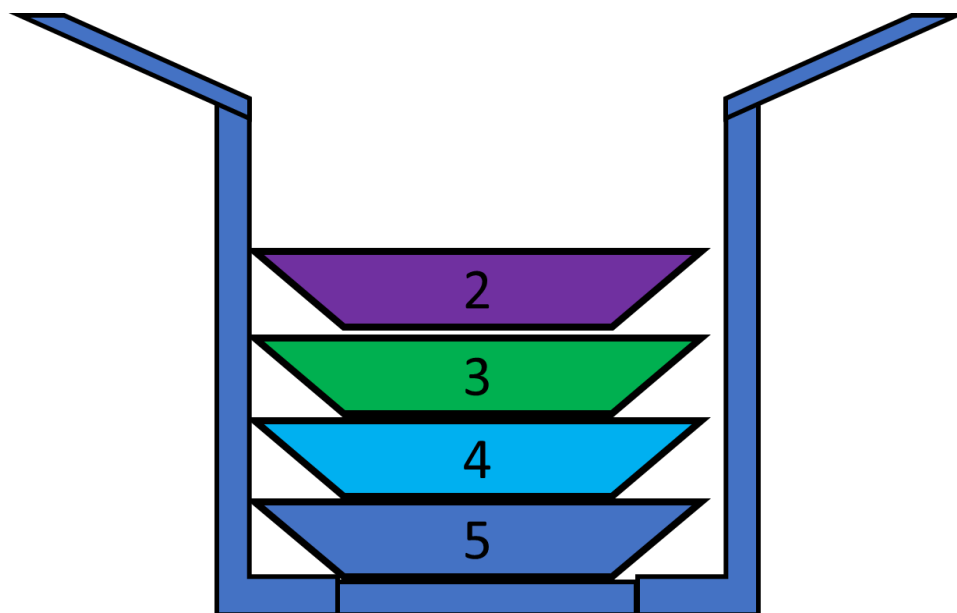
• نیمه دوم را برعکس درج کن یعنی بشقاب ۴ و ۵ اما ۵ و ۴ باید درج بشه ۵ بالا و ۴ پایین.



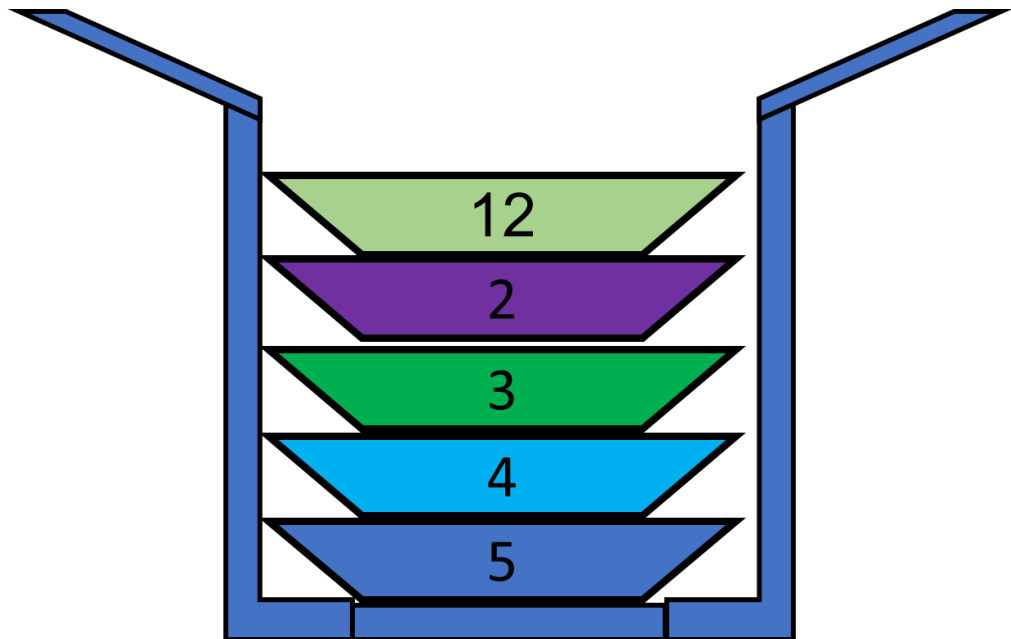
فرض کنید کد پشته را نمیدانید. اما یک درسنامه برای کل مبحث پشته کافیت. پشته شبیه یک جعبه‌ی کادو هست که فقط از یک طرفش ذی‌تا میتونیم برداریم یا بزاریم و به بقیش دسترسی نداریم. یعنی مثلاً برای پشته بشقاب فوق. انگار کل پشته بشقاب تو یک جعبه باشه. و فقط بتونیم به بشقاب یک دسترسی داشته باشیم. اگر بشقاب از جعبه برداریم میگیم Pop و اگر بشقابی را به جعبه بزاریم میگیم push، همین.



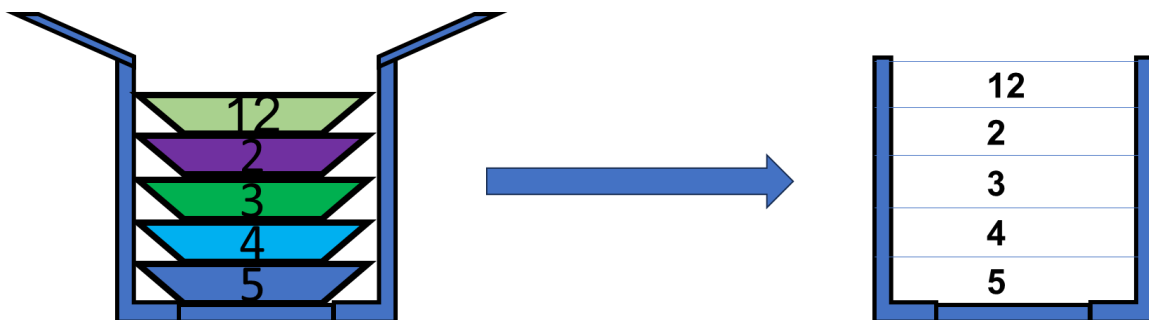
از جعبه‌آبی بالا pop کن! یعنی بالاترین عنصر را خارج کن. یعنی در اینجا بشقاب ۱ را بردار.



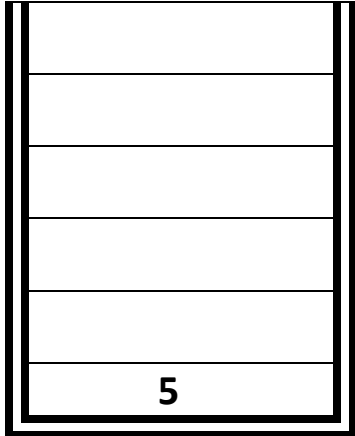
حالا به جعبه بشقاب ۱۲ رو push کن. بشقاب دوازده حالا باید بروی بقیه‌ی بشقاب ها قرار بگیره. یعنی ...



نکته اینکه حالا این جعبه رو در علم رایانه با آرایه مدل می کنن و خوب به شکل زیر نمایش می دهند. یک آرایه که به شکل عمود روی پاهای خود ایستاده است.



خوب حالا قبل از حل مساله اصلی که در بالا شد. با آموخته های خود تا بدین جا؛ توالی از pop و push ها رو روی پشته ی زیر انجام بدید و گام به گام با اعمال توالی آرایه را ترسیم کنید.

|                                                                                   |                                                                                                                                                                             |
|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <ol style="list-style-type: none"> <li>1. S.push(12)</li> <li>2. S.push(4)</li> <li>3. S.push(15)</li> <li>4. S.pop()</li> <li>5. S.push(12)</li> <li>6. S.pop()</li> </ol> |
|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

7. نکته آنکه pop آرگومان نمیگیرد. و push یک آرگومان بعنوان ورودی دریافت میکند.
8. نکته دیگر آنکه هر پشته اندازه ای دارد و تا بی نهایت مقدار نمیپذیرد. مثلا اگر پشته ای سه تا فضا داشته باشد تا سه مقدار فضا میگیرد و اگر push بیشتری روی پشته پر صورت گیرد پشته به اصطلاح overflow میکند یا سرریز میکند یا دچار خطا میشود.
9. از طرفی این اتفاق برای pop از آرایه ی خالی هم میفتد. و به اصطلاح پشته دچار خطای underflow میشود.

صف هم به همین شکل هست :

درپشته اونیکه اول در جعبه قرار میگیرد آخر سر از جعبه بیرون میآد، بهش میگن First in Last Out و صف از قانون First in First Out استفاده میکنه و چون ما با صف زیاد سرو کار داریم First in First Out زیاد شنیدیم که به اختصار میشه FIFO .

یک صف هم مثل یک تونل یا یک جعبه که دو طرف اون باز هست. از یک طرف در جعبه درج میکنیم از طرف دیگر بر میداریم. همین. به گذاشتن عنصر در صف درج در صف یا EnQueue میگن و به حذف از صف DeQueue میگن.

صف همونجوری افقی نشون میدن و از عمودی کردن ساختار آرایه جلوگیری میکنن چون اینطوری از نظر بصری تحلیلش ساده تره.

صف یک کارواش ماشین را در نظر بگیرید ...



به ابتدای صف Head صف یا سر صف و به انتهای آن دم صف یا tail می گویند.

اگر خودرویی از ابتدای صف حذف شود و وارد کارواش شود و شروع به گرفتن سرویس کند در واقع این اتفاق از ابتدای صف میفتد. و به آن DnQueue یا حذف از صف می گویند.

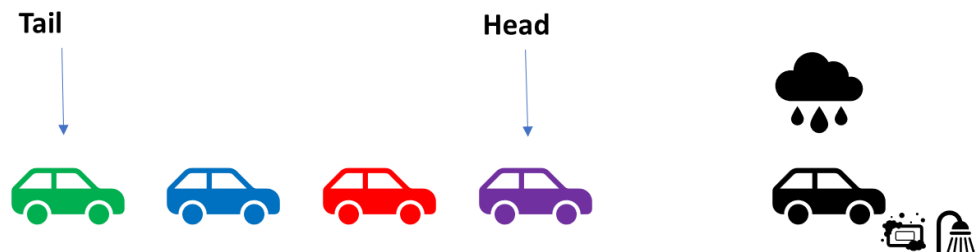
اگر خودرویی به صف بخواهد اضافه شود به انتهای صف اضافه خواهد شد. عملیات درج در صف EnQueue گفته میشود و به انتهای صف رخ میدهد.

پس همواره از یک سر صف داده ها وارد و از طرف دیگر خارج میشوند.

فرض کنید در شکل فوق خودرویی دیگر به صف اضافه شود. نام عملیات درج به صف چه بود ؟ DeQueue یا EnQueue ؟

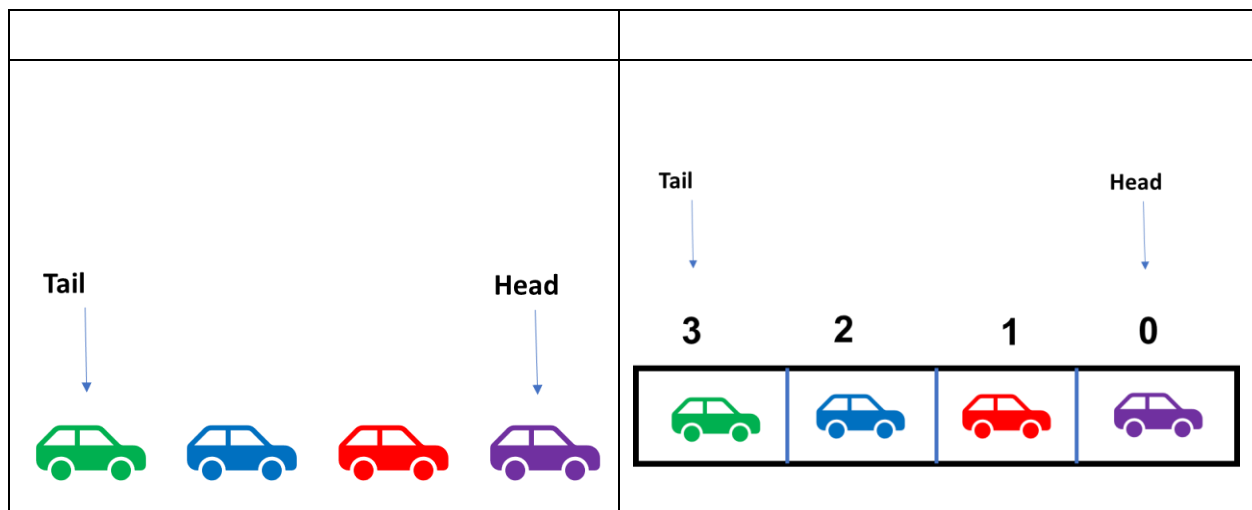


خوب حال کارواش خالی شده و یک ماشین از صف خارج و وارد صف میشود.



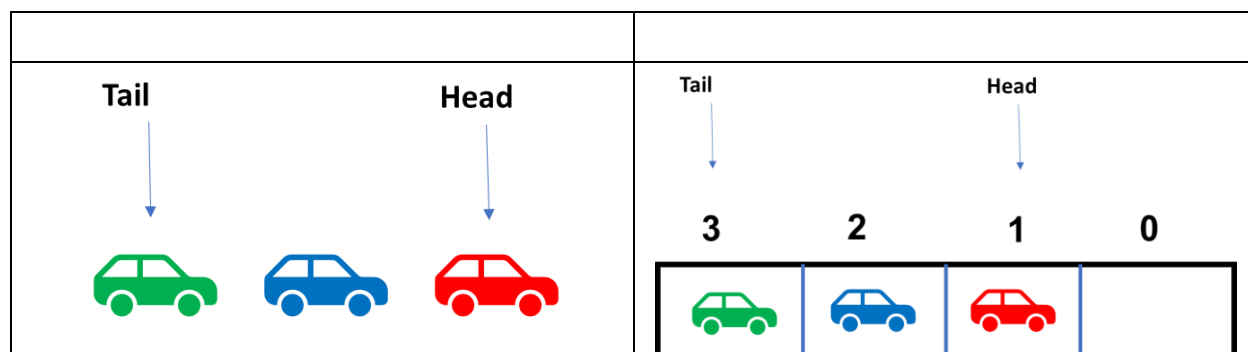
حدس شما درست است خودرویی که در ابتدای صف هست بایستی از صف خارج شده چون آنکه اول آمده است اول باید وارد کارواش شود این سیاست را FIFO میخوانیم.

اما در رایانه چگونه صف را مدل میکنیم ؟ دوباره با آرایه میتوانیم مدل سازی را انجام دهیم.

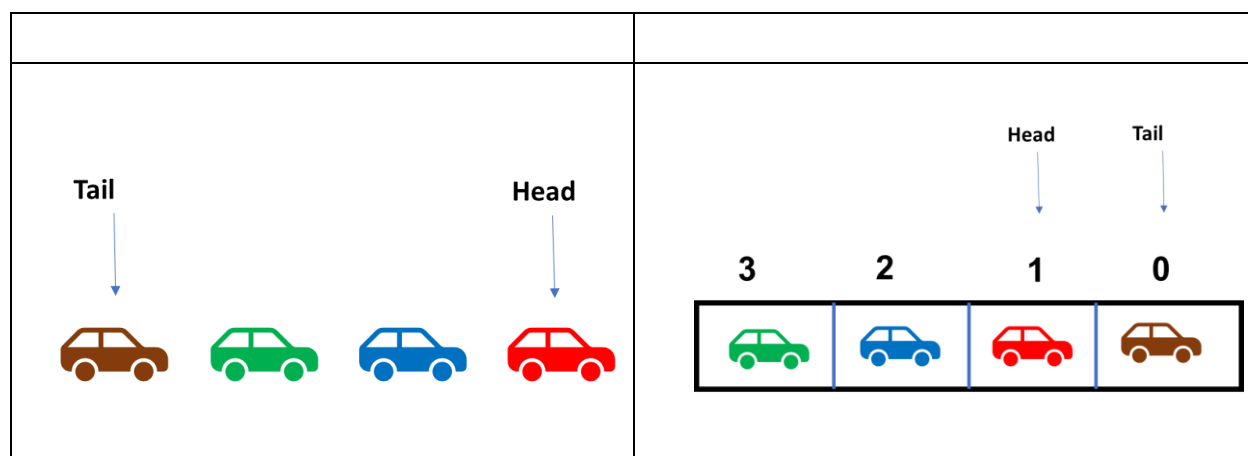




اما یک نکته در آرایه به جای اینکه پس از آنکه یک ماشین از صف خارج شود، همه‌ی ماشین‌ها یک قدم جلو بروند، اینجا زرنگی می‌کنیم و head رو جابه‌جا می‌کنیم به طوریکه به خانه‌ی بعدی اشاره می‌کند.



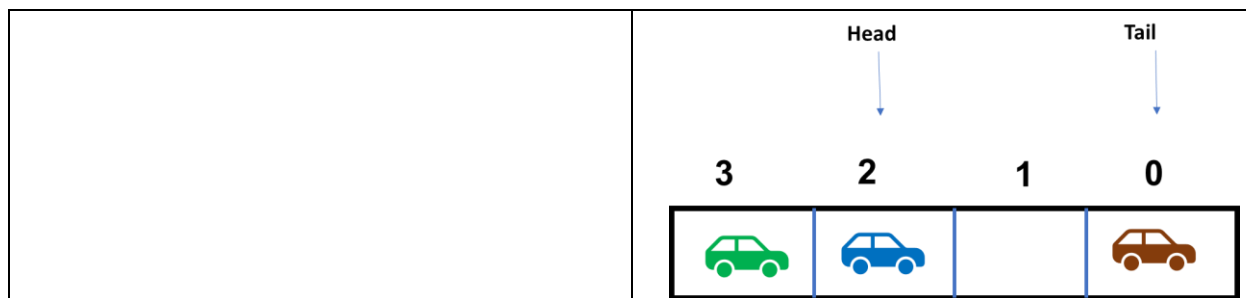
خوب حالا اگر یک ماشین جدید دیگه بیاد چی؟ هیچی باز هم tail رو جابه‌جا می‌کنیم تا به آن اشاره کند یعنی:



اگر خودروی جدید دیگری بخواهد وارد صف شود چه میشود؟ ظرفیت صف تکمیل است و خودروی جدید نمی‌پذیرد. به اصطلاح صف دچار خطای overflow میشود.

حال فرض کنید خودروی دیگر باید از صف خارج و وارد کارواش شود چه خواهد شد؟ حذف از ابتدای صف رخ میدهد یعنی جایی که head دارد بدانجا اشاره می‌کند.

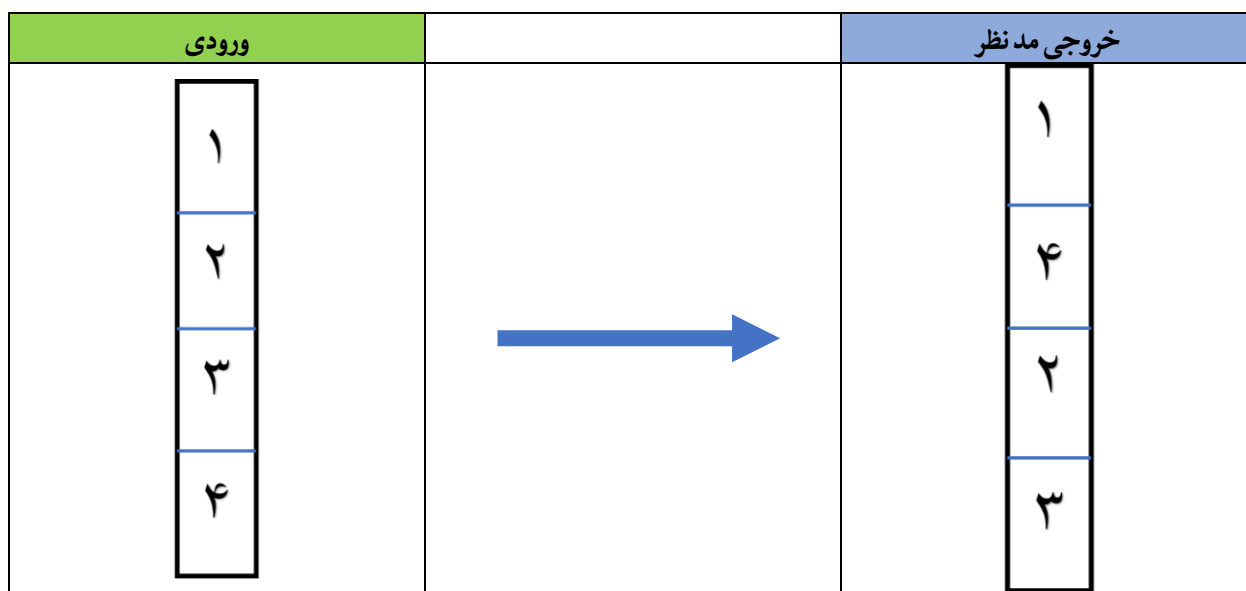
|  |  |
|--|--|
|  |  |
|--|--|



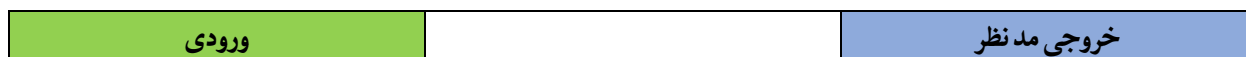
حال صف یکجای خالی ایجاد شده است. اصطلاح دیگری هم داریم... اگر صف خالی باشد و بخواهیم از صف EnQueue کنیم با خطای Under flow روبرو میشویم.

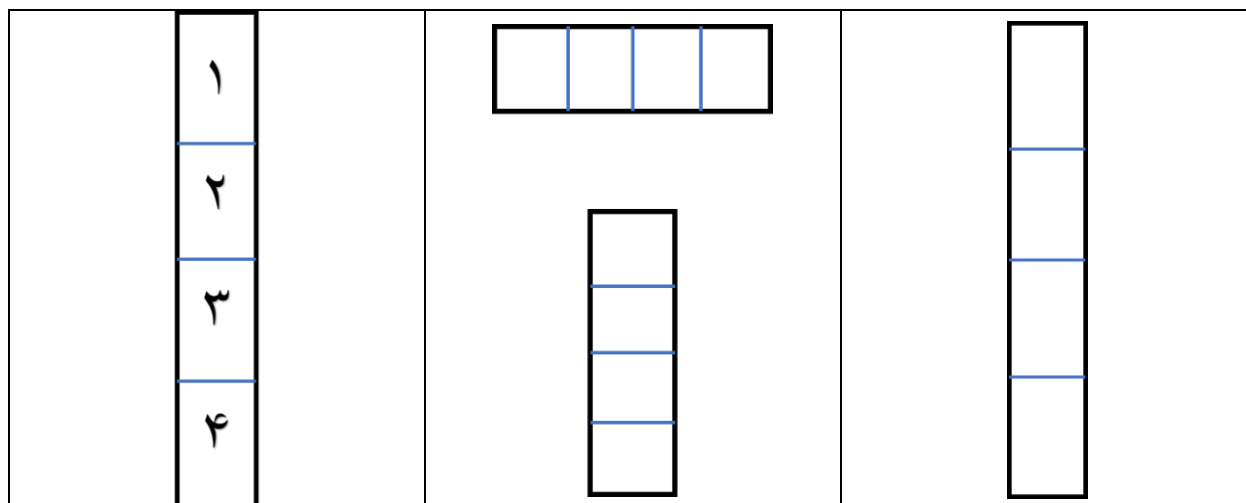
خب سبرگردیم به سوال حال سوال را چطور حل کنیم؟ یک راه حل اینه که همه دیتا ها رو pop کنیم اونایی که باید غیر معکوس در پشت نهایی قرار بگیرن را در پشتی دیگری بریزیم و اونهایی که باید معکوس قرار بگیرن رو در یک صف بریزیم. و بعد یک در میان آن ها را به پشتی اول انتقال دهیم.

برای مثال اول پشتی ای با مقادیر ۱ و ۲ و ۳ و ۴ را در نظر بگیرید. ورودی و خروجی مدنظر در زیر آمده است.

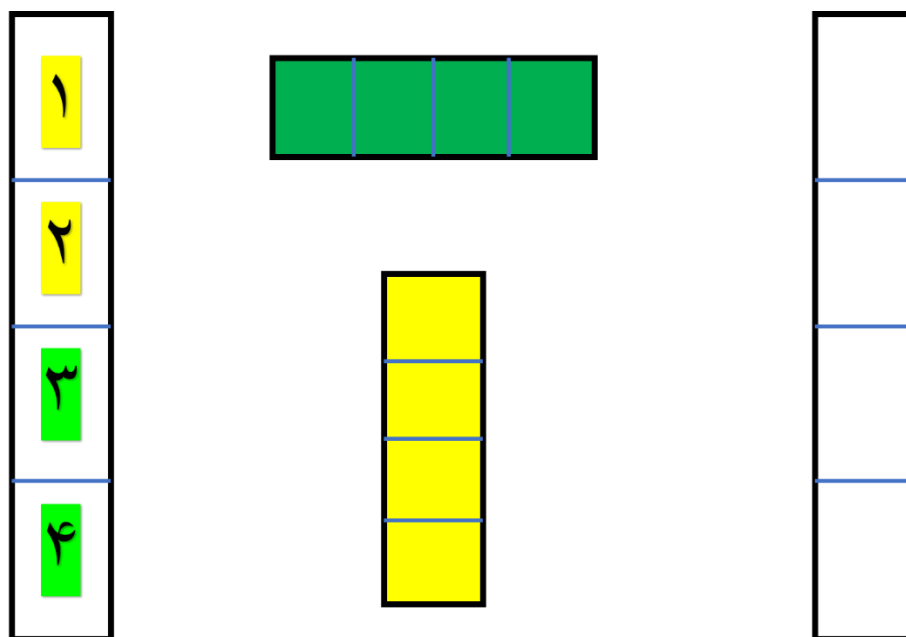


در این روش از یک پشتی و یک صف کمکی استفاده شده است.

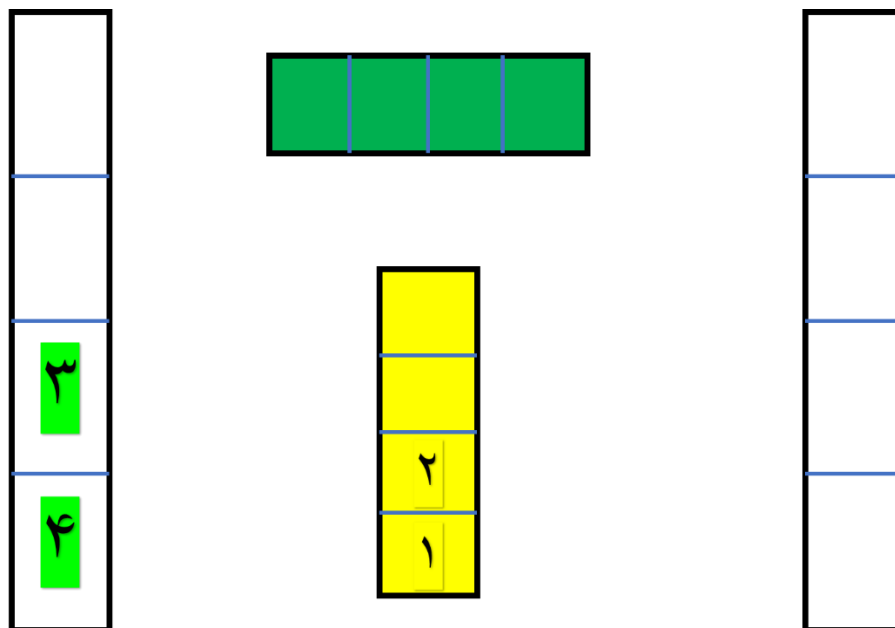




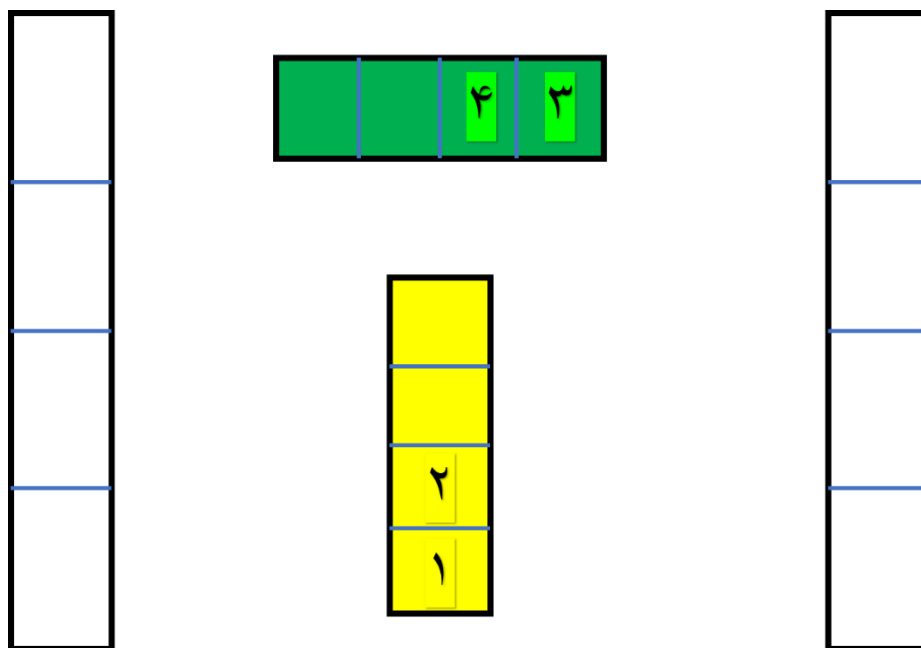
خب به توالی زیر توجه بفرمایید. پشته کمکی برای نیمه‌ی بالا stack ما به کار میرود پس از stack خود نصف عناصر را pop میکنیم و در پشته کمکی میریزیم.



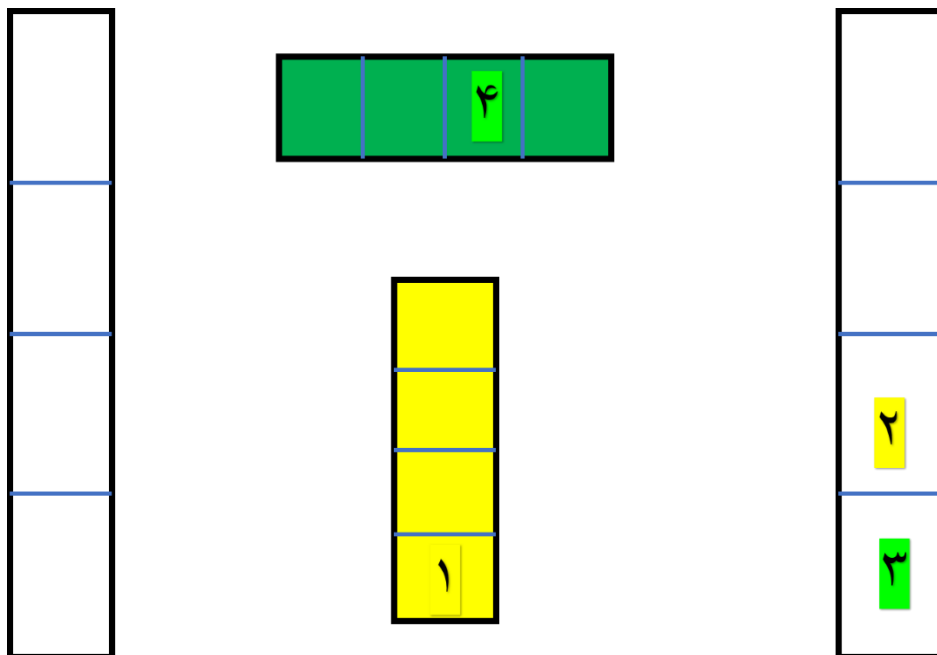
گام. حال صف کمکی نیز برای نیمه‌ی دوم بکار گرفته شده است. پس نیمه دوم stack را از پشته خارج و به صف انتقال میدهیم.



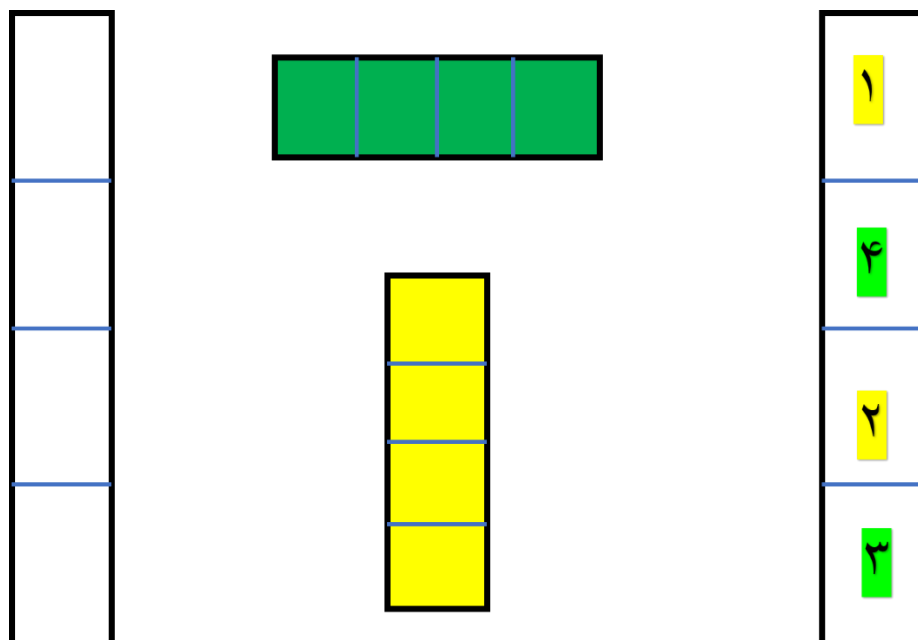
گام سوم. حال یک در میان از صف کمکی وپشته کمکی دیتا را خارج و به پشته push میکنیم.



گام چهارم. یک cycle کار در زیر آمده است.



گام: در cycle دوم همه دیتا ها همانطور که خواستیم به پشته نهاییمان رسید.



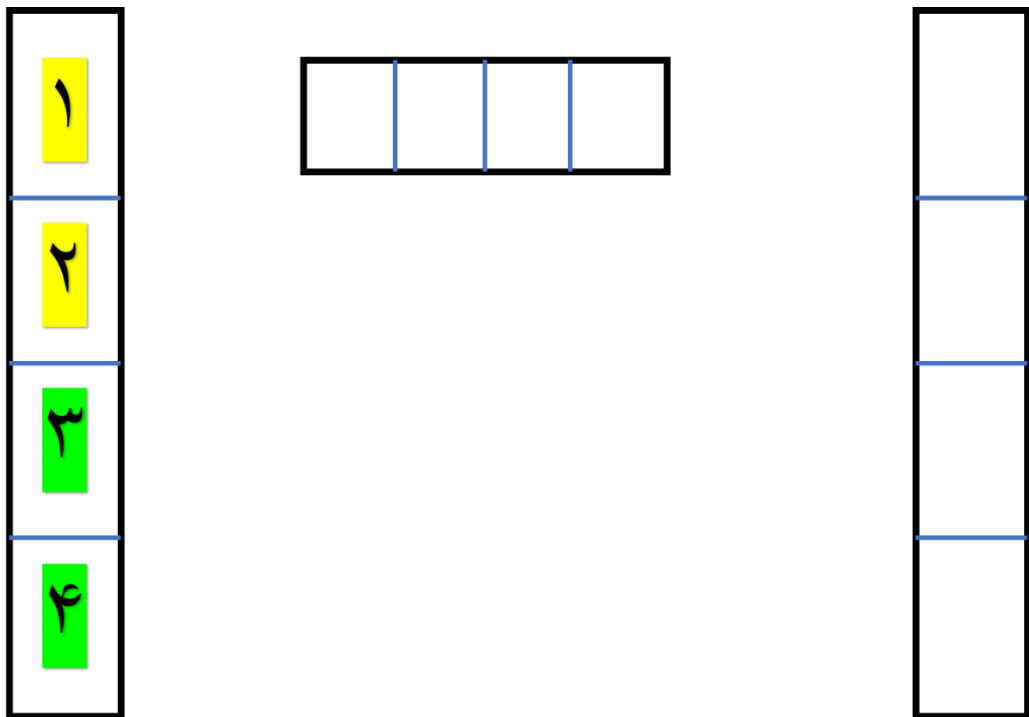
اما نکته فهمیدید چه بود. روشی که مساله را حل کردیم `not in place` یا `out of place` بود چون فضاهای کمکی نسبت به ورودی رشد کردند. اما اینجا باید یک روش `inplace` را داشته باشیم یعنی فضای کمکیان به اندازه‌ی ورودی وابسته نباشد. کمی فکر کنید.

البته دو نکته:

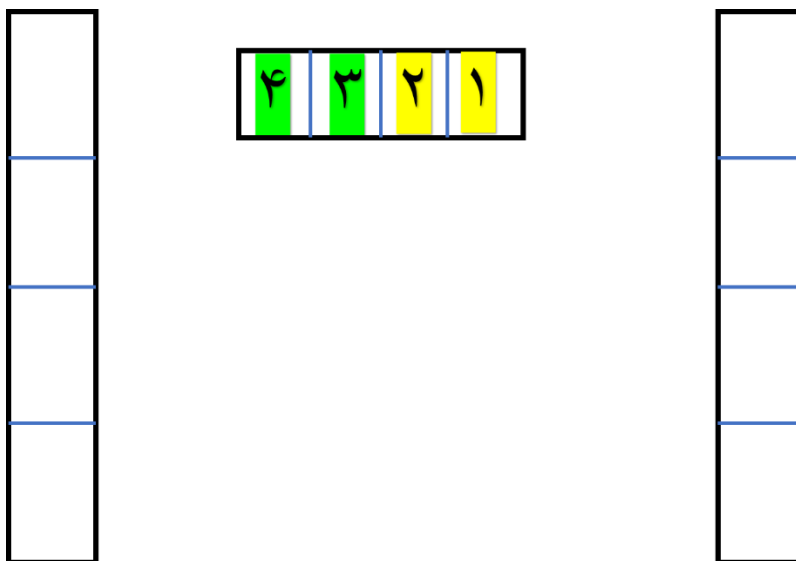
- خود صورت سوال گفته که یک صف مجازه! و البته تنها یک صف مجازه.
- یعنی بیخیال پشته شو.

منظور از اینکه صف مجازه و `inplace` حل کن یعنی صف نسبت به ورودی اضافه بشه اشکال نداره اما اگه خودت حافظه کمکی میخوای استفاده کنی `inplace` باشه یعنی نسبت به ورودی رشد نکنه. به هر حال روش حل ساده است و بایک صف میتوان اینکار را کرد. چون ساختار صف دوسر دارد. ( البته این روشی که داریم صحبت میکنم خیلی `tight` نیست. در واقع از ساختار صف که ما ایندکس `head` و `tail` را داریم استفاده میکنیم. اما بگذارید بگوییم چون جالب است.

با مثال من همراه شوید تا روشی که به ذهنم رسیده را بیاموزید.



اول همه را وارد صف کمکی میکنیم.



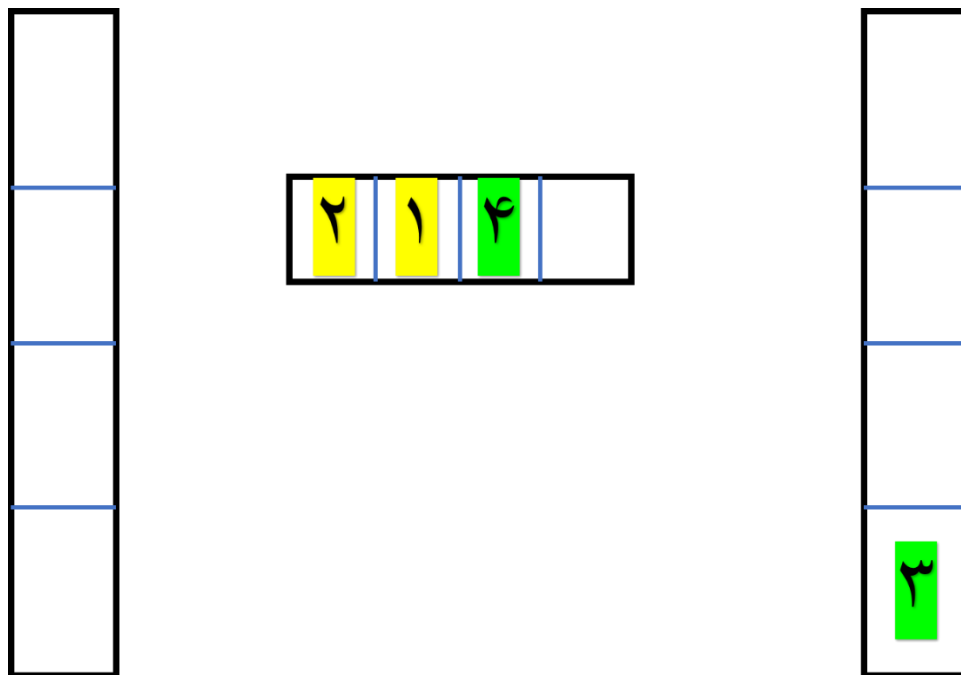
سپس حالت یک چرخه روی صف کمکی ایجاد میکنیم از ابتدای صف خارج و به انتهای صف کمکی اضافه میکنیم گویی میخواهیم جای رنگ سبز و زرد رو دسته ای عوض کنیم یعنی زرد ها بیان آخر صف و سبزها بیان اول صف. یعنی:

| گام اول |  | گام دوم |  |
|---------|--|---------|--|
|         |  |         |  |
| گام سوم |  |         |  |
|         |  |         |  |

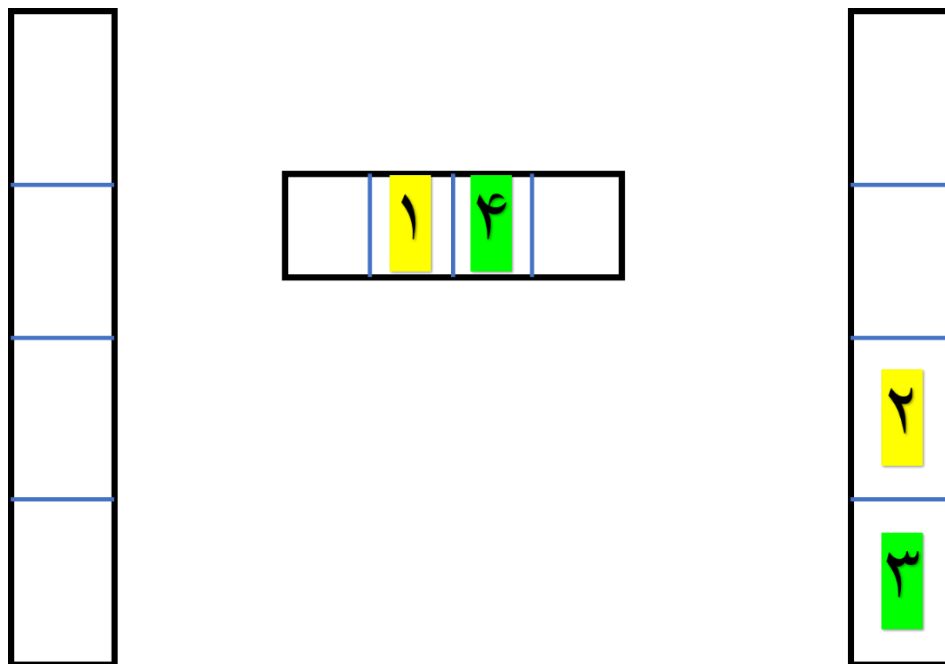
حال یک عنصر از انتهای صف خارج میکنیم وارد پشته میکنیم و یک عنصر از ابتدای صف خارج و وارد پشته میکنیم.

گویی بخش زرد رنگ در پشته ذخیره شده باشد و گویی رنگ سبز در یک صف ذخیره شده باشد.

پس خواهیم داشت:



گام





گام

|   |
|---|
|   |
| ۶ |
| ۲ |
| ۳ |

|  |   |  |  |
|--|---|--|--|
|  | ۱ |  |  |
|--|---|--|--|

|  |
|--|
|  |
|  |
|  |
|  |

گام

|   |
|---|
| ۱ |
| ۶ |
| ۲ |
| ۳ |

|  |  |  |  |
|--|--|--|--|
|  |  |  |  |
|--|--|--|--|

|  |
|--|
|  |
|  |
|  |
|  |

خب وبه جواب رسیدیم.

حال شما سوال بجایی میتوانید پرسید که استاد شما ساختار صف را به هم زدید و من میگویم حق با شماست. از آنجایی که صف قانون های نوشته شده ی بسیار سفت و سخت دارد. اگر از طرفی درج شود دیگر از همان طرف قابل حذف نیست. کاری که ما در بالا انجام دادیم اما این دلیل نمیشود ما نتوانیم خلاقیت در حل را از خود نشان دهیم.

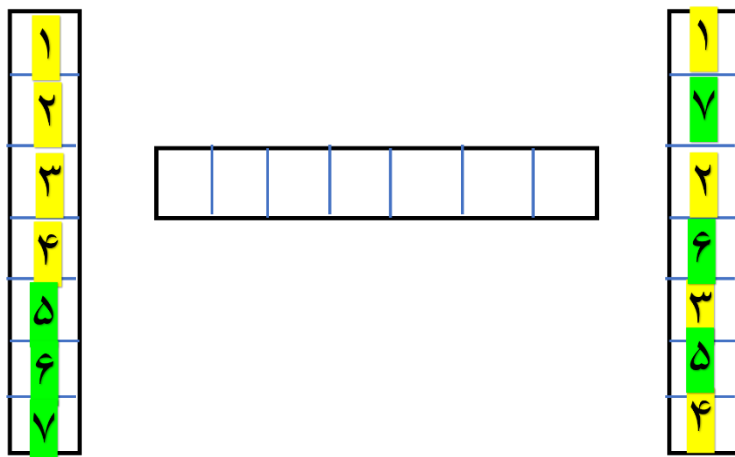
آیا روشی بهتر میشناسید که بدون دست زدن به ساختار صف کمکی بتوانید به خروجی زیر برسید؟ بله

ما باید شبیه مهره های گردنبند که یک در میان مهره های سبز و زرد دارند اعداد را در صف کمکی قرار دهیم و سپس همه را یک جا وارد پشته نهایی کنیم. البته بخاطر ذات انتقال صف به پشته باید یادمان باشد این دو صف اولویت قوانینشان برعکس هم هست. پس اونیکه میخوانیم روی پشته باشد باید ته صف باشه و اونیکه پایین پشته جلوی صف.

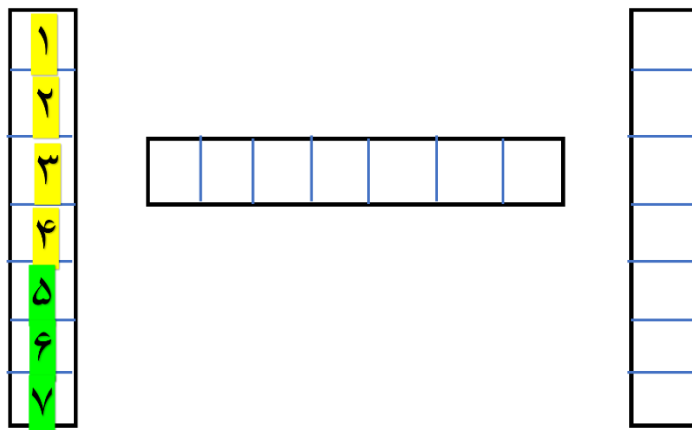
- همرویک بار وارد صف میکنیم
- بخش جلوی صف یا  $n/2$  زرد را وارد پشته میکنیم
- اینکار باعث میشود نریب نیمه اول معکوس شود و ترتیب نیمه دوم در صف حفظ شود
- حال مهره چینی میکنیم
- یکی از پشته کم میکنیم و به صف اضافه میکنیم
- صف را یکدور در خودش میچرخانیم
- یکی از پشته کم میکنیم به صف اضافه میکنیم
- صف را یکدور در خودش میچرخانیم
- اینکار را انقدر تکرار میکنیم تا همه مهره ها یک در میان کنار هم بشینند.

یکبار در شکل زیر ببینیم و با هم کار را دنبال کنیم ؛

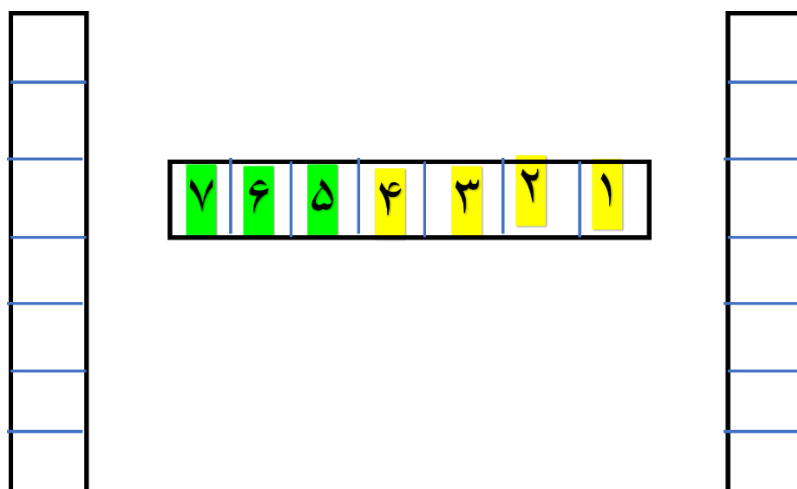
گام صفرم : ورودی در سمت راست و خروجی مدنظر در چپ آمده است.



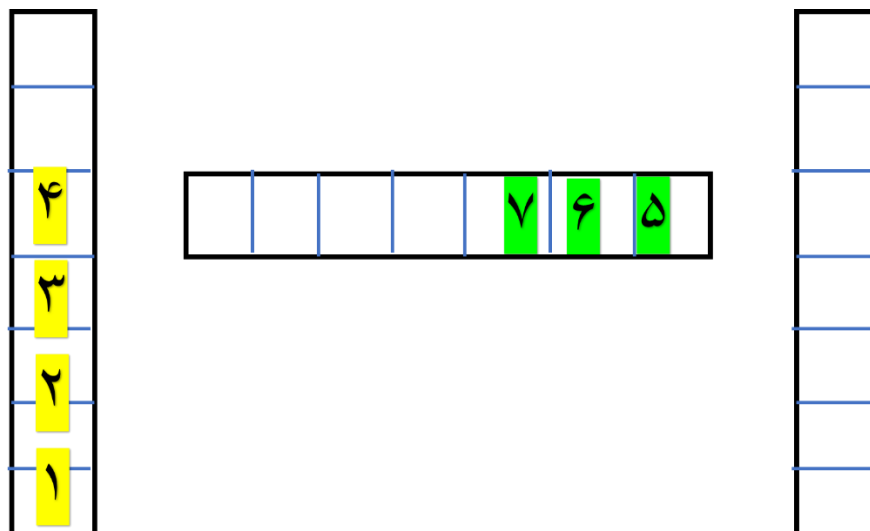
گام: شروع.



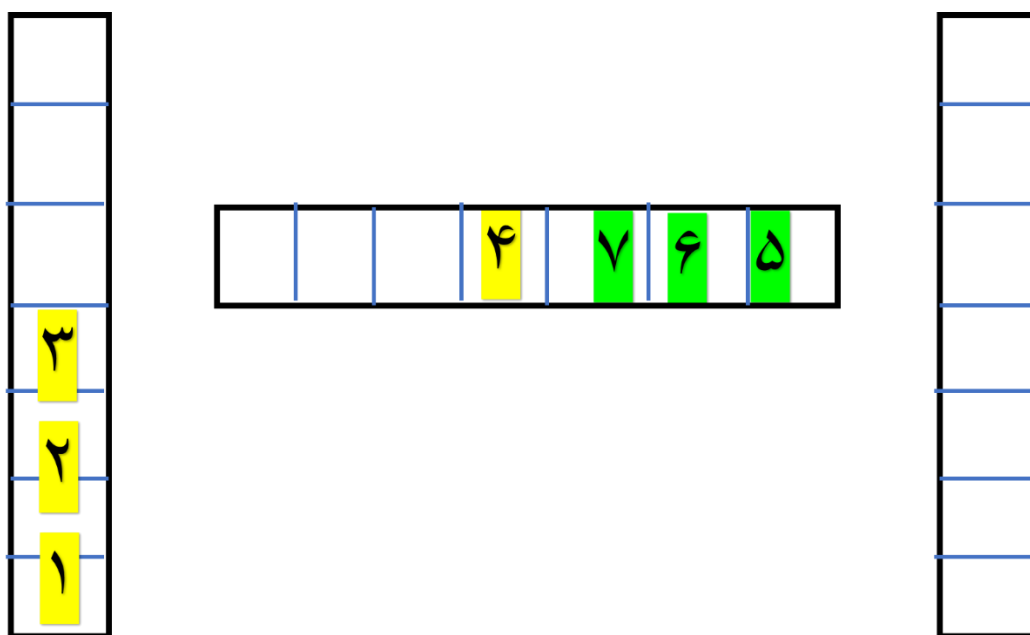
گام: همه را از پشته وارد صف میکنیم.



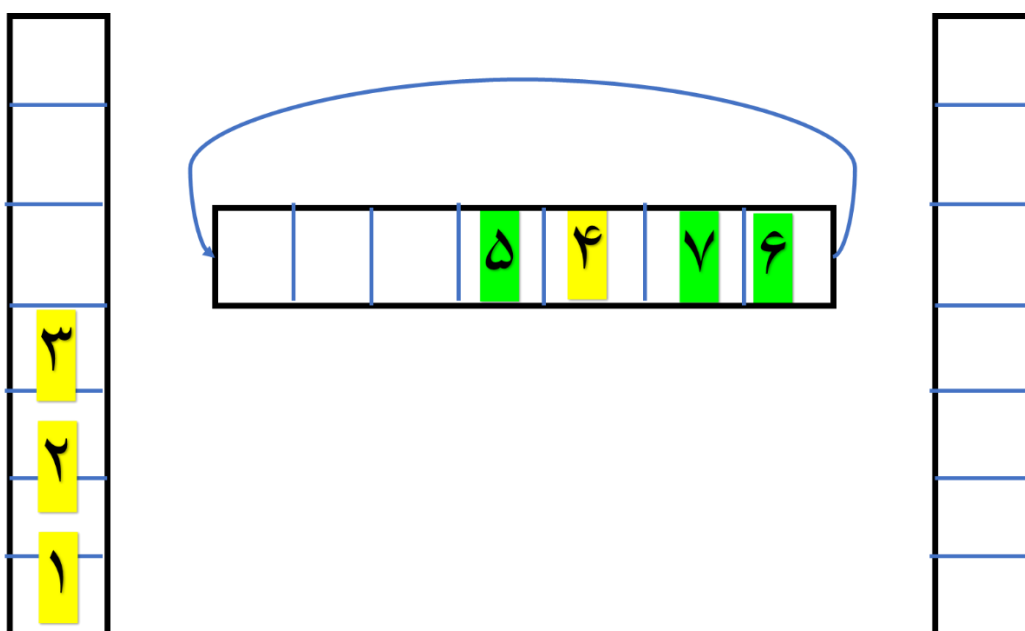
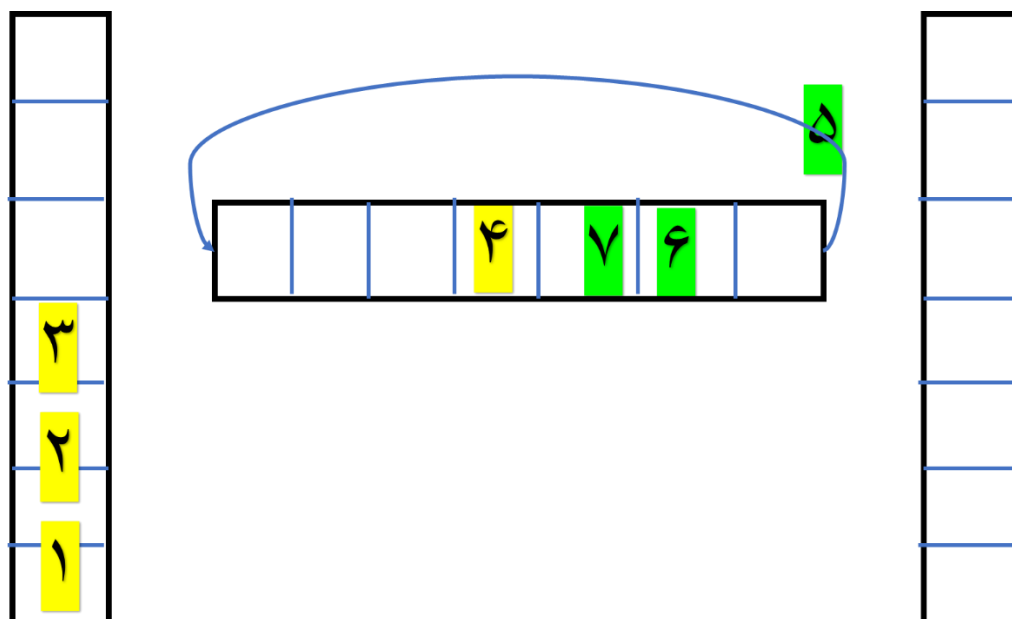
گام: حال بخش زرد را به پشته برمیگردانیم



گام. حال مهره چینی میکنم. یکی از این یکی از آن. از پشته اول یکی بر میدارم.

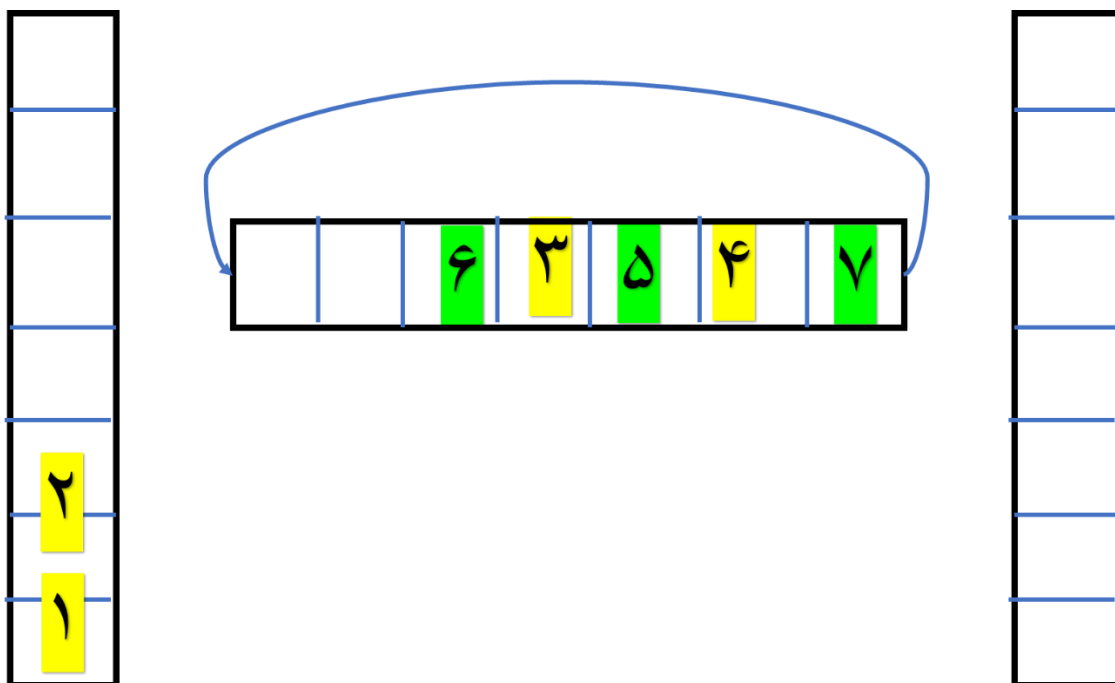


گام. حال صف را یکدور میچرخانیم یعنی پنج را از ابتدای صف خارج و به انتهای همان صف اضافه میکنیم.

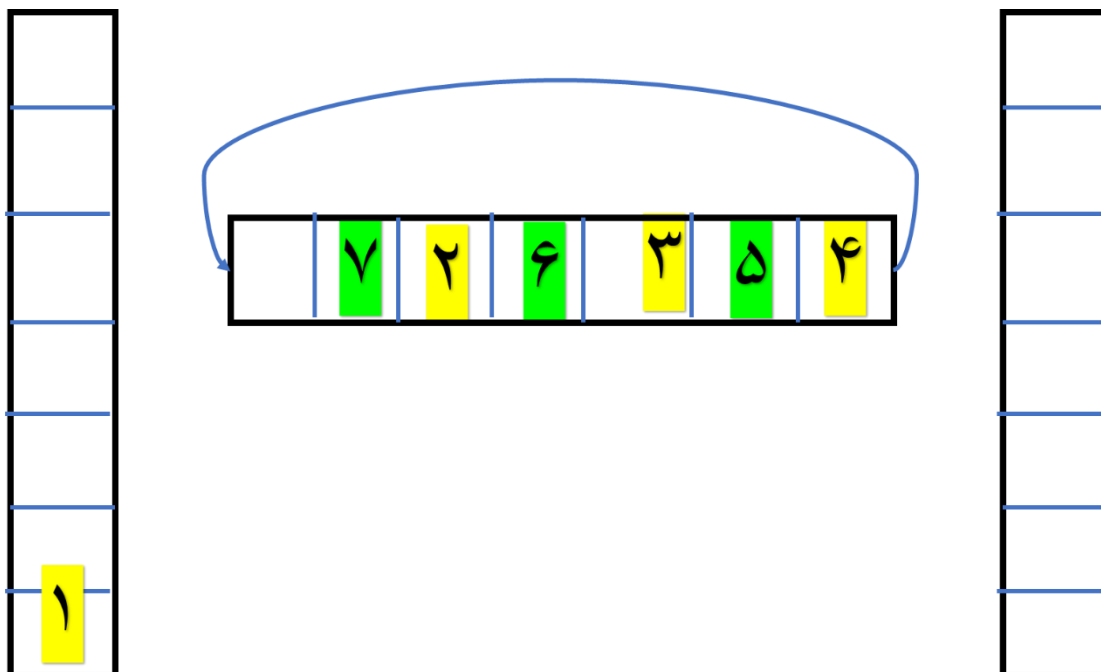


گام. دوباره یک عنصر دیگر از پشته یعنی ۳ را وارد صف میکنیم و دوباره صف را میچرخانیم یعنی پس از اینکه ۳ وارد صف شد. ۶ را از ابتدای صف خارج میکنیم و به انتهای صف اضافه میکنیم.

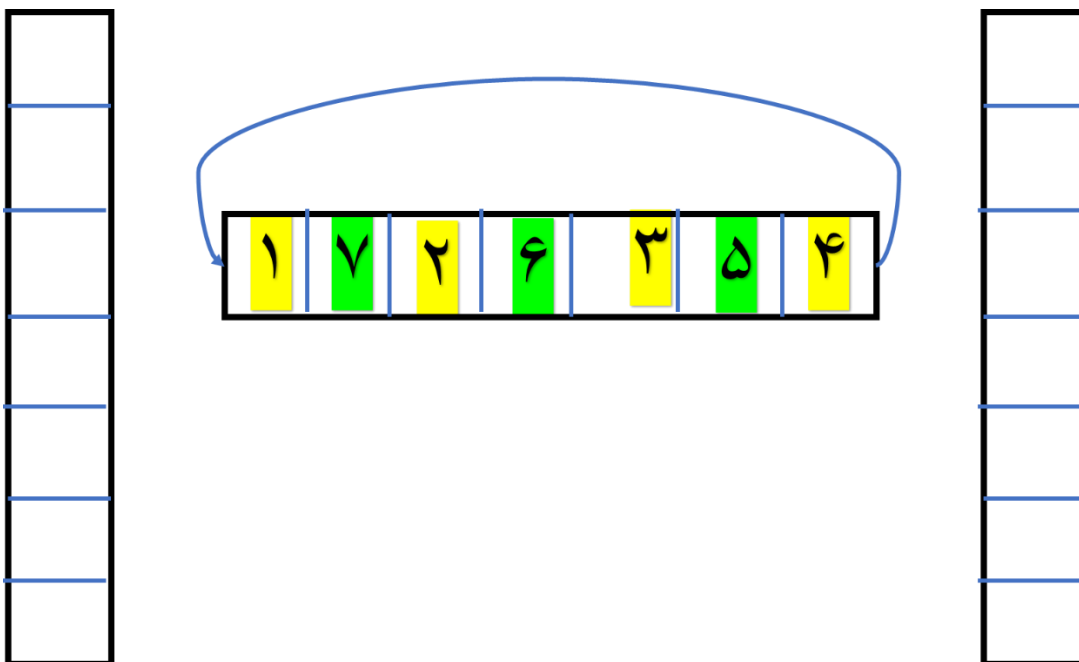
پس



گام. یک بار دیگر اینبار با ۲ و ۷.



در نهایت یک روهم وارد میکنیم. و پاسخ نهایی شد.



گام پایانی. حال از صف خارج و به پشته وارد میکنیم.

