پاسخ سوالات تمرین دوم

با استفاده از دقیقا دو پشته (Stack)، ساختمان داده صف (Queue) را پیاده سازی کنید و زمان اجرای عملیات صف را تحلیل کنید.

استک ها را S1,S2 مینامیم . فرض کنید از استک S1 برای ذخیره ی داده ها و از S2 به عنوان فضای کمکی استفاده میکنیم.

روش اول:

- برای Enqueue کردن عنصر x آن را مستقیم در push ، S1 می کنیم.
- برای Dequeue کردن، تمامی عناصر موجود در S1 را pop کرده و در S2 ،push می کنیم. آخرین عنصری که pop کرده ایم همان عنصری است که باید Dequeue شود. آن را می کنیم. حال دوباره تمامی عناصر را به S1 برمیگردانیم.

Enqueue -> O(1)

Dequque -> O(n)

روش دوم:

- برای Enqueue کردن عنصر x ، ابتدا تمام اعضای S1 را pop می کنیم و در S2 ،push می کنیم. حال x را در S1 ،push میکنیم و سپس تمامی اعضایی که در S2 ریخته شده بودند را pop کرده و دوباره در S1 ،push میکنیم.
- برای Dequeue کردن صرفا از S1 ،pop می کنیم.

Enqueue -> O(n)

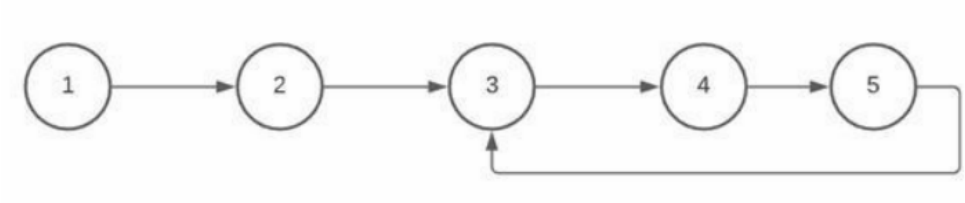
Dequeue -> O(1)

روش سوم:

- عملیات Enqueue همانند راه اول می باشد و عنصر x آن را مستقیم در S1 ،push می کنیم.
- ابتدا چک میکنیم که S2 خالی است یا خیر. اگر خالی نبود مستقیم از آن pop می کنیم. اگر خالی بود تمامی عناصر S1 را همانند راه دوم به S2 منتقل می کنیم و آخرین عضو را pop می کنیم. در این راه حل نیازی نیست دوباره عناصر را از S2 به S1 برگردانیم. علت آن نیز این است که هر سری که میخواهیم Dequeue کنیم، S2 را چک می کنیم. در واقع این راه سوم بهینه شده ی راه اول است و پیچیدگی زمانی مشابه آن دارد.

می‌گوییم یک لیست پیوندی دارای حلقه است، هرگاه گره‌ای وجود داشته باشد که در حین پیمایش لیست پیوندی بیش از یکبار بازدید شود.

به عنوان مثال لیست پیوندی‌ای که head به لیستی از گره های $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 3$ اشاره می‌کند، دارای حلقه است که شکل آن مطابق زیر است.



در مقابل برای نمونه لیست پیوندی‌ای که head به لیستی از گره های $1 \rightarrow 2 \rightarrow 3 \rightarrow NULL$ اشاره می‌کند، داری حلقه نیست. همچنین توجه کنید که اعداد گفته شده در مثال های فوق اعداد گره‌ها و مشخص کننده آن‌ها هستند، نه مقادیر گره‌ها.

حال از شما خواسته شده الگوریتمی طراحی کنید که با گرفتن یک پوینتر که به head یک لیست پیوندی اشاره می‌کند، مشخص کند که آیا آن لیست پیوندی دارای حلقه است یا خیر؟

```
slow = head

fast = head

while (slow!=null and fast!=null and fast.next!=null)

{

    slow = slow.next

    fast = fast.next.next

    if (slow == fast)

        return true

}

return false
```

از دو پوینتر استفاده می‌کنیم. یک پوینتر slow که هر مرحله یک واحد در لیست پیوندی حرکت می‌کند و یک پوینتر fast که هر مرحله دو واحد در لیست پیوندی حرکت می‌کند. اگر پوینتر fast به انتهای لیست پیوندی برسد حلقه‌ای در لیست پیوندی وجود نداشته اما اگر دو پوینتر در مرحله‌ای به هم برسند در لیست پیوندی حلقه داریم زیرا که اگر لیست پیوندی حلقه داشته باشد در مرحله‌ای هر دو پوینتر وارد بخش حلقوی لیست پیوندی شده‌اند. از طرفی می‌دانیم که پوینتر fast هر مرحله دو واحد و پوینتر slow هر مرحله یک واحد به جلو حرکت می‌کند پس در واقع در حرکت نسبی می‌توان پوینتر slow را ثابت فرض کرد در حالی که پوینتر fast در هر مرحله یک خانه حلقه جلو می‌رود پس در نهایت به پوینتر slow می‌رسد.