محمدحسن تهلری - ۹۹۳۱۰۶۱

① برای enqueue عناصر را در استک دوم (B) push کرده تا زمانی که پر شود، سپس در استک اول (A) push می‌کنیم. شکل ←


```
Enqueue (A, B, x) {
    if (B.full != true){
        B.Push(x) ; }
    else
        A.Push(x);
    if (A.full == true) {
        print "Queue is full" }
```

برای dequeue عناصر را ابتدا در استک دوم push می‌کنیم و سپس آن عناصر را از بالا به ترتیب pop کرده و در استک اول push می‌کنیم، در نهایت استک اول را pop می‌کنیم.

شکل ←

```
Dequeue (A, B) {
    while (B.empty != true) {
        A.Push (B.Pop) ;}
    A.Pop ;
```

متدهای .empty و .full برای خالی بودن استک را با true و false مشخص می‌کنند.

زمان اجرای Enqueue از مرتبه O(1) است چونکه ربطی به تعداد عناصر موجود در استک ندارد.

زمان اجرای Dequeue از مرتبه O(n) است چونکه باید به ترتیب تمام استک دوم را Pop و در استک اول Push کند پس به تعداد n جستجو دارد.

۲) از همان الگوریتم Search در لینک لیست استفاده می‌کنیم منتها آن را درون حلقه for می‌گذاریم تا اعداد i از 1 تا n را دانه دانه چک کرده و با اولین عددی که در تا پوینتر به آن اشاره کرده بیرون بیاید و در غیر این صورت NULL برمی‌گرداند.

* x.next منظور پوینتر اشاره کننده به گره بعدی است.

```
Duplicate search (L, x):
for (int i=1 ; i≤n ; i++) {
    x = L.head ;
    R = 0 ;
    while (x ≠ NULL && x.next ≠ i) {
        x = x.next ; }
    if (x ≠ NULL) {
        print ("daraye halghe ast") ;
        R = 1 ;
        break ; } }
if (R == 0)
    print ("faghede halghe ast");
```