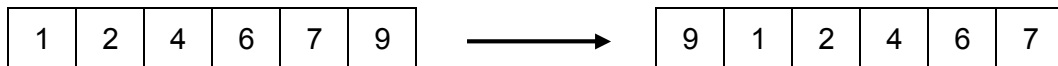




שאלה 4 (20 נקודות)

בהנתן מערך ממויין, **הסטה מעגלית** שלו תוגדר כהזזת כל איבר במערך תא אחד קדימה, כאשר האיבר האחרון במערך מועבר לתא הראשון. לדוגמה:



מערך ממויין שהופעלו עליו מספר שלם כלשהו של הסטות מעגליות (0 הסטות או יותר) ייקרא **מערך ממויין בהסטה**. לדוגמה, המערכים הבאים ממויינים בהסטה:

(הופעלו 2 הסטות)

5	6	1	2	3	4
---	---	---	---	---	---

(הופעלו 4 הסטות)

3	4	5	6	1	2
---	---	---	---	---	---

(הופעלו 0 הסטות)

1	2	3	4	5	6
---	---	---	---	---	---

עליכם לממש את הפונקציה `findcirc()` שמקבלת מערך ממויין בהסטה (שימו לב שגודל ההסטה אינו ידוע), את גודלו `n` וערך `x`, ומחפשת בו את הערך `x`. הפונקציה מחזירה את האינדקס במערך שבו `x` נמצא, או (-1) אם `x` אינו נמצא במערך. ניתן להניח כי כל אברי המערך שונים זה מזה. **שימו לב** כי קיים פתרון הפועל בסיבוכיות זמן $O(\log n)$; פתרון בסיבוכיות זמן גבוהה מזו יזכה לכל היותר בניקוד חלקי.

```
int findcirc(int a[], int n, int x) {
```

```
    int i1, i2, k=breakpoint(a,n);
```

```
    i1 = binsearch(a,k+1,x);
```

```
    i2 = binsearch(a+k+1,n-k-1,x);
```

```
    if (i1!=-1) return i1;
```

```
    if (i2!=-1) return i2+k+1;
```

```
    return -1;
```

```
}
```



```
int breakpoint(int a[], int n) {  
  
    int low=0, high=n-1, mid;  
    while (low < high) {  
        mid = (low+high)/2;  
        if (a[mid] > a[mid+1]) return mid;  
        else if ((a[mid] < a[low]) && (a[mid] < a[high])) high = mid;  
        else if ((a[mid] > a[low]) && (a[mid] > a[high])) low = mid;  
        else return high;  
    }  
    return low;  
}
```

```
int binsearch(int a[], int n, int x) {  
  
    if (n <= 0) return -1;  
    if (a[n/2] == x) return n/2;  
    if (a[n/2] > x)  
        return binsearch(a, n/2, x);  
    else {  
        int pos = binsearch(a+n/2+1, n-n/2-1, x);  
        return (pos==-1) ? -1 : pos+n/2+1;  
    }  
}
```



- אפשרות נוספת לפתרון השאלה:

```
int findcirc(int a[], int n, int x) {  
  
    int low=0, high=n-1, mid;  
  
    while (high >= low) {  
        mid = (low+high)/2;  
        if (a[mid] == x) return mid;  
  
        if (a[high] > a[low]) {  
            if (a[mid]>x) high=mid-1;  
            else low=mid+1;  
        }  
        else if (a[mid] > a[high]) {  
            if (x>=a[low] && x<=a[mid]) high=mid-1;  
            else low=mid+1;  
        }  
        else {  
            if (x>=a[mid] && x<=a[high]) low=mid+1;  
            else high=mid-1;  
        }  
    }  
  
    return -1;  
}
```