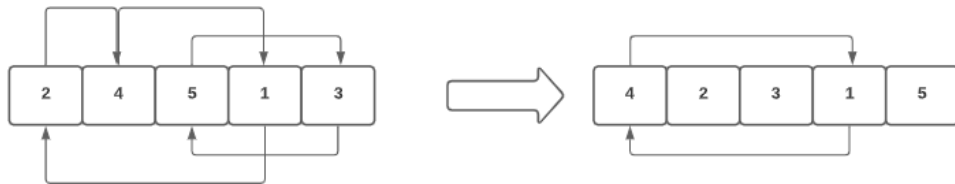


## Cyclic-Sort



Sorted Array

\* Cyclic Sort (Sort the elements from 1 to n)

arr = { 3, 4, 6, 2, 1, 5 }

0 1 2 3 4 5

{ 1, 2, 3, 4, 5, 6 }

1 == 1

→ 0 1 2 3 4 5

3 4 6 2 1 5

→ 6 4 3 2 1 5

→ 5 4 3 2 1 6

→ 1 4 3 2 5 6

→ 1 2 3 4 5 6

1, 1, 1, 1, 1

$T(n) = (n-1) + O(n)$

$S(n) = O(n)$

$\frac{3-1}{2}$

i=0

i=0

```
while (i < a.length) {
    if (a[i] != i+1) {
        int ind = a[i] - 1;
        int x = a[i];
        a[i] = a[ind];
        a[ind] = x;
    } else {
        i++;
    }
}
```



\* Cyclic Sort (Sort the elements from 1 to n)

arr = { 3, 4, 6, 2, 1, 5 }

11 to 16

0 1 2 3 4 5  
3, 4, 6, 2, 1, 5

0 1 2 3 4 5  
{ 1, 2, 3, 4, 5, 6 }

1, 1, 1, 1

1 == 1

T.C.  $O(n-1) + O(n)$

S.C.  $O(1)$   $O(n)$   $\frac{3-1}{2}$

0 1 2 3 4 5  
12, 14, 16, 12, 11, 15  
16, 14, 13, 12, 11, 15  
15, 14, 13, 12, 11, 16  
11, 14, 13, 12, 11, 16  
11, 12, 13, 14, 15, 16

```

while (i < a.length) {
    if (a[i] != i+1) {
        int index = a[i] - 1;
        int x = a[i];
        a[i] = a[index];
        a[index] = x;
    } else {
        i++;
    }
}

```

```

public static void cyclicSort(int[] arr) {

```

```

    if(arr.length <= 1) {

```

```

        return;
    }

```

```

    int i = 0;

```

```

    while(i < arr.length) {

```

```

        if(arr[i] - 1 != i) {

```

```

            int otherIndex = arr[i] - 1;

```

```

            int x = arr[i];

```

```

            arr[i] = arr[otherIndex];

```

```

            arr[otherIndex] = x;

```

```

        } else {

```

```

            i++;

```

```

        }

```

```

    }

```

```

}

```

[LeetCode 268 - Missing Number \[easy\]](#)

[LeetCode 442 - Find All Duplicates in an Array \[medium\]](#)

[LeetCode 448 - Find All Numbers Disappeared in an Array \[easy\]](#)

[LeetCode 645 - Set Mismatch \[easy\]](#)