LIST EXERCISES

1. Given a list, only rotation operation is allowed on list. We can rotate the list as many times as we want. Return the maximum possible of summation of i*arr[i].

Examples :

```
Input: arr[] = {1, 20, 2, 10}
Output: 72
We can 72 by rotating array twice.
{2, 10, 1, 20}
20*3 + 1*2 + 10*1 + 2*0 = 72

Input: arr[] = {10, 1, 2, 3, 4, 5, 6, 7, 8, 9};
Output: 330
We can 330 by rotating array 9 times.
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
0*1 + 1*2 + 2*3 ... 9*10 = 330
```

2. Given a list of elements of length N, ranging from 0 to N – 1. All elements may not be present in the list. If element is not present then there will be -1 present in the list. Rearrange the list such that A[i] = i and if i is not present, display -1 at that place.

Examples:

```
Input : arr = {-1, -1, 6, 1, 9, 3, 2, -1, 4, -1}
Output : [-1, 1, 2, 3, 4, -1, 6, -1, -1, 9]

Input : arr = {19, 7, 0, 3, 18, 15, 12, 6, 1, 8,
               11, 10, 9, 5, 13, 16, 2, 14, 17, 4}
Output : [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
          11, 12, 13, 14, 15, 16, 17, 18, 19]
```

3. Given a sorted list of positive integers, rearrange the list alternately i.e. first element should be maximum value, second-minimum value, third -second max, fourth-second min and so on.

Input: *arr[] = {1, 2, 3, 4, 5, 6, 7}*
Output: *arr[] = {7, 1, 6, 2, 5, 3, 4}*

Input: *arr[] = {1, 2, 3, 4, 5, 6}*
Output: *arr[] = {6, 1, 5, 2, 4, 3}*

LIST EXERCISES

4. Given a list, rearrange the list such that :

   If index i is even, arr[i] <= arr[i+1]

   If index i is odd, arr[i] >= arr[i+1]

   Note: There can be multiple answers.

```
Input  : arr[] = {2, 3, 4, 5}
Output : arr[] = {2, 4, 3, 5}
Explanation : Elements at even indexes are
smaller and elements at odd indexes are greater
than their next elements

Note : Another valid answer
is arr[] = {3, 4, 2, 5}

Input  :arr[] = {6, 4, 2, 1, 8, 3}
Output :arr[] = {4, 6, 1, 8, 2, 3}
```

5. Given a list and a number k where k is smaller than size of list, we need to find the k'th smallest element in the given list. It is given that the list elements are distinct.

Input: *arr[] = {7, 10, 4, 3, 20, 15}*
k = 3
Output: *7*

Input: *arr[] = {7, 10, 4, 3, 20, 15}*
k = 4
Output: *10*

6. Given n size unsorted list, find its mean and median.

```
Mean of an array = (sum of all elements) /
                   (number of elements)

Median of a sorted array of size n is defined
as below :
It is middle element when n is odd and average
of middle two elements when n is even.

Since the array is not sorted here, we sort
the array first, then apply above formula.
```

LIST EXERCISES

Examples:

```
Input  : a[] = {1, 3, 4, 2, 6, 5, 8, 7}
Output : Mean = 4.5
         Median = 4.5
Sum of the elements is 1 + 3 + 4 + 2 + 6 +
5 + 8 + 7 = 36
Mean = 36/8 = 4.5
Since number of elements are even, median
is average of 4th and 5th largest elements.
which means (4 + 5)/2 = 4.5

Input  : a[] = {4, 4, 4, 4, 4}
Output : Mean = 4
         Median = 4
```

7. Given a list of positive numbers, find the maximum sum of a subsequence with the constraint that no 2 numbers in the sequence should be adjacent in the list. So 3 2 7 10 should return 13 (sum of 3 and 10) or 3 2 5 10 7 should return 15 (sum of 3, 5 and 7).

Examples :

```
Input : arr[] = {5, 5, 10, 100, 10, 5}
Output : 110

Input : arr[] = {1, 2, 3}
Output : 4

Input : arr[] = {1, 20, 3}
Output : 20
```

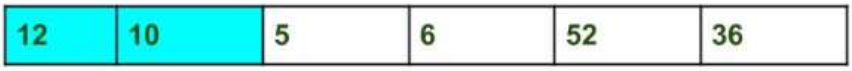
8. Given two integer lists arr1 [] and arr2 [] sorted in ascending order and an integer k. Find k pairs with smallest sums such that one element of a pair belongs to arr1 [] and other element belongs to arr2 [].

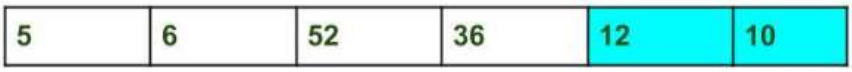
Examples:

```
Input :  arr1[] = {1, 7, 11}
         arr2[] = {2, 4, 6}
         k = 3
Output : [1, 2],
         [1, 4],
         [1, 6]
Explanation: The first 3 pairs are returned
from the sequence [1, 2], [1, 4], [1, 6],
[7, 2], [7, 4], [11, 2], [7, 6], [11, 4],
[11, 6]
```

LIST EXERCISES

9. There is a given a list and split it from a specified position, and move the first part of list add to the end.





Examples:

```
Input : arr[] = {12, 10, 5, 6, 52, 36}
            k = 2
Output : arr[] = {5, 6, 52, 36, 12, 10}
Explanation : Split from index 2 and first
part {12, 10} add to the end .

Input : arr[] = {3, 1, 2}
           k = 1
Output : arr[] = {1, 2, 3}
Explanation : Split from index 1 and first
part add to the end.
```

10. Given an unsorted list arr[] of size n, the task is to find the minimum difference between any pair in the given list.

Input: *arr[] = {1, 2, 3, 4}*
Output: *1*
The possible absolute differences are:
{1, 2, 3, 1, 2, 1}

Input: *arr[] = {10, 2, 5, 4}*
Output: *1*

11. Given a list of integers, sort the list according to frequency of elements. For example, if the input list is {2, 3, 2, 4, 5, 12, 2, 3, 3, 3, 12}, then modify the list to {3, 3, 3, 3, 2, 2, 2, 12, 12, 4, 5}.

12. Given an list sequence [A1, A2 …An], the task is to find the maximum possible sum of increasing subsequence S of length k such that S1<=S2<=S3……<=Sk.

LIST EXERCISES

Input :
n = 8 k = 3
A=[8 5 9 10 5 6 21 8]
Output : *40*
Possible Increasing subsequence of Length 3 with maximum possible sum is 9 10 21

Input :
n = 9 k = 4
A=[2 5 3 9 15 33 6 18 20]
Output : *62*
Possible Increasing subsequence of Length 4 with maximum possible sum is 9 15 18 20

13. Given two lists of N and M integers. The task is to find the number of unordered pairs formed of elements from both lists in such a way that their sum is an odd number.
Note: An element can only be one pair.

Input: *a[] = {9, 14, 6, 2, 11}, b[] = {8, 4, 7, 20}*
Output: *3*
{9, 20}, {14, 7} and {11, 8}

Input: *a[] = {2, 4, 6}, b[] = {8, 10, 12}*
Output: *0*

14. You are given a list of 0s and 1s in random order. Segregate 0s on left side and 1s on right side of the list. Traverse list only once.
Input list = [0, 1, 0, 1, 0, 0, 1, 1, 1, 0]
Output list = [0, 0, 0, 0, 0, 1, 1, 1, 1, 1]

15. Given n list that contains both positive and negative integers, find the product of the maximum product sub list.

```
Input: arr[] = {6, -3, -10, 0, 2}
Output:   180  // The subarray is {6, -3, -10}

Input: arr[] = {-1, -3, -10, 0, 60}
Output:   60  // The subarray is {60}

Input: arr[] = {-2, -3, 0, -2, -40}
Output:   80  // The subarray is {-2, -40}
```