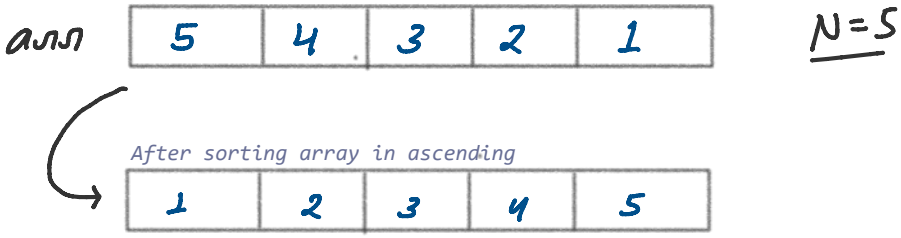
INSERTION SORT

24 September 2023 09:22

23/09/2023

INSERTION SORT

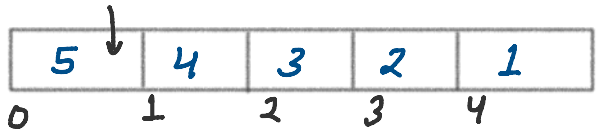
✅ *SORTINGS 03: INSERTION SORT*
- Move element of arr[0,i-1] that are greater than key to one position ahead of there current position



DRY RUN

Example:01

Always ignore the first element



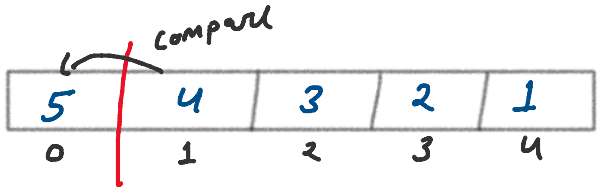
i=1 j= i-1

initially store the second element as a key value to Move element of arr[0,i-1] that are greater than key to one position ahead of there current position

key = arr[i] 4

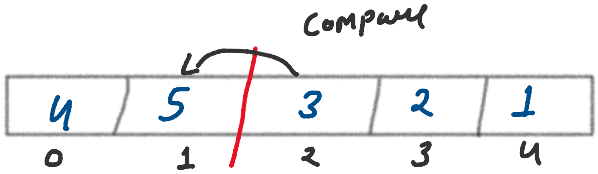
1st Iteration

key = 4
i=1 , j=0



5 > 4 ✓ → 4 5 3 2 1

2nd Iteration



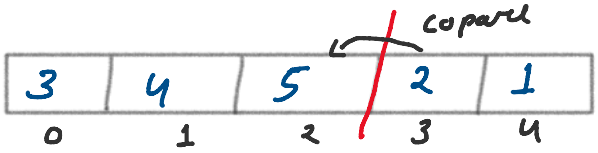
key = 3

2nd Iteration

key = 3
i = 2 , j = 1
i = 2 , j = 0

5 7 3 → 4 3 5 2 1
4 7 3 → 3 4 5 2 1

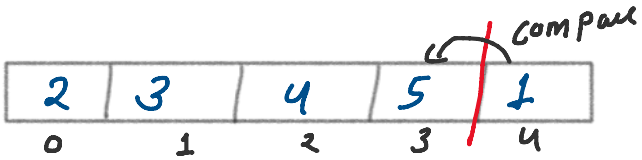
3rd Iteration



key = 2
i = 3 , j = 2
i = 3 , j = 1
i = 3 , j = 0

5 7 2 → 3 4 2 5 1
4 7 2 → 3 2 4 5 1
3 7 2 → 2 3 4 5 1

4rt Iteration



key = 1
i = 4 , j = 3
i = 4 , j = 2
i = 4 , j = 1
i = 4 , j = 0

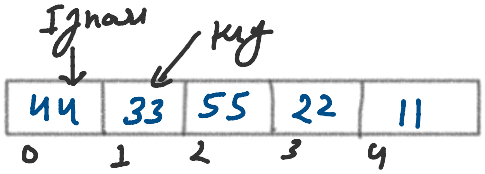
5 7 1 → 2 3 4 1 5
4 7 1 → 2 3 1 4 5
3 7 1 → 2 1 3 4 5
2 7 1 → 1 2 3 4 5

i = 5 { RUN jaaooo..... (i < 5) END }

Final output

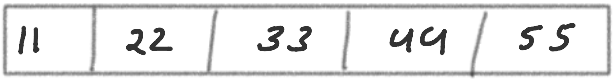
DRY RUN

Example:02 ans



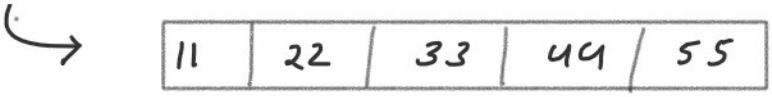
N = 5

i = 1 , j = i - 1



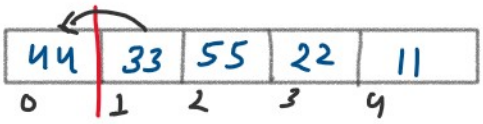
i=1 , j= i-1

key = arr[i]



SORTED ARRAY

Iteration:01

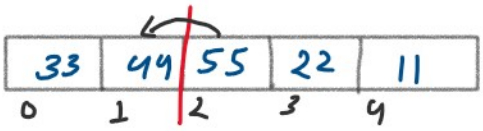


key = 33

i = 1 , j = 0

44 > 33 ✓ → 33 44 55 22 11

Iteration:02



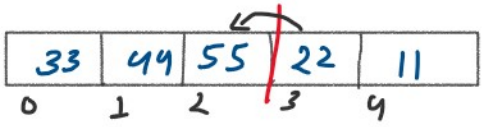
key = 55

i = 2 , j = 1

44 ? 55 ✗ → SAME ⇒ 33 44 55 22 11

BREAK inner loop

Iteration:03



key = 22

i = 3 , j = 2

i = 3 , j = 1

i = 3 , j = 0

55 ? 22 ✓ →

44 ? 22 ✓ →

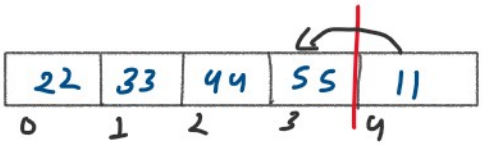
33 > 22 ✓ →

33 44 22 55 11

33 22 44 55 11

22 33 44 55 11

Iteration:04



key = 11

i = 4 , j = 3

i = 4 , j = 2

i = 4 , j = 1

i = 4 , j = 0

55 ? 11 ✓ →

44 ? 11 ✓ →

33 ? 11 ✓ →

22 ? 11 ✓ →

22 33 44 11 55

22 33 11 44 55

22 11 33 44 55

11 22 33 44 55

j = -1 ✓ END inner loop

i=4, j=0 22711

J=-1 X END Inner loop

i=5 { RUN j000.... (i<5)

↳ Final output

11	22	33	44	55

$i \in [1, N]$ Iteration

$j \in [i-1, 0]$ Comparison

Outer loop	Ist	1
	IInd	2
	IIIrd	3
	IVth	4

Inner loop ↳

GENERALIZE

N=5

Iteration		Comparison
1	→	1
2	→	2
0		0
0		0
0		0
n-1	→	n-1
n	→	n

$S_n = 1+2+3+\dots+n$

$= O\left(\frac{n^2}{2} - \frac{n}{2}\right) \Rightarrow O(n^2)$

Toc = $O(N^2)$

Soc = $O(1)$

```cpp
// ✅SORTINGS 03: INSERTION SORT

#include<iostream>
#include<vector>
using namespace std;

// INSERTION SORT Function
void insertionSort(vector<int> &arr){
    int N=arr.size();
    for(int i=1; i<N; i++){
        // Key value =  key value se pahle ki all values ko hum compare karte hai key value se hi
        //taki key value ko hum uske right position par rakh paye
        int key = arr[i];
        int j = i-1;

        // Move element of arr[0,i-1] that are greater than key to one position ahead of there current position
        while(j>=0 && arr[j]>key){
            arr[j+1]=arr[j];
            j--;
        }
        // Insert key at right index/position (Insertion)
        arr[j+1]=key;
    }
}

int main(){
    vector<int> arr{5,4,3,1,2};

    insertionSort(arr);
    for(auto value: arr){
        cout<<value<<" ";
    }
    return 0;
}

/*
INPUT: {5,4,3,1,2}
OUTPUT: {1,2,3,4,5}
TIME COMPLEXITY: O(N^2)
SPACE COMPLEXITY: O(1)
*/
```