3/21/2021

Sorting- O(n^2)

Bubble, Selection, Insertion Sort

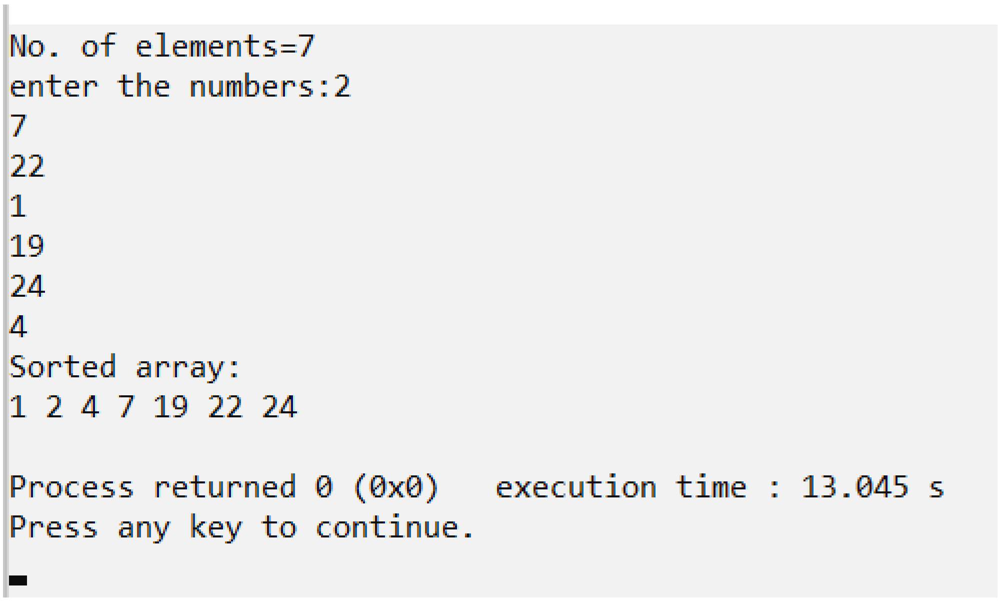
Munyatam Bhardwaj
42416401518

1. BUBBLE SORT

CODE:

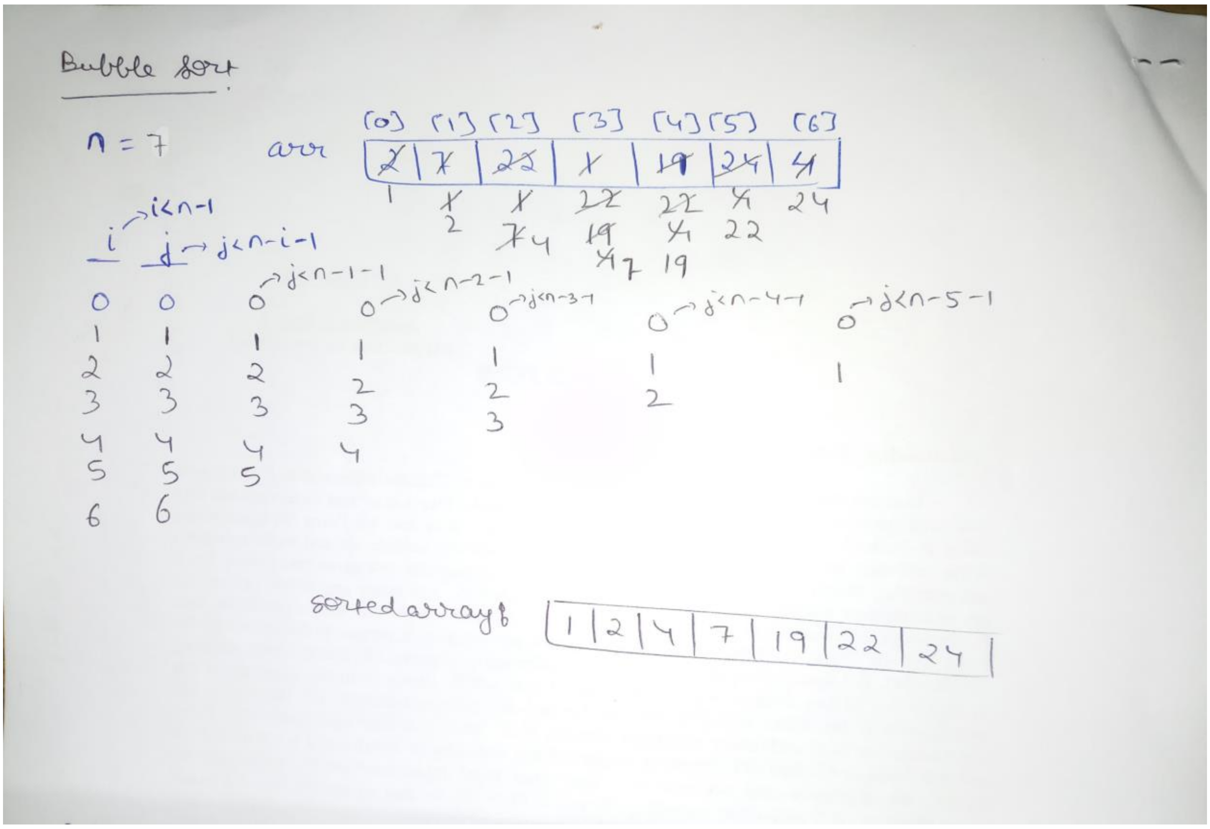
```
// C++ program for implementation of Bubble sort
#include <bits/stdc++.h>
using namespace std;

void swap(int *xp, int *yp)
{
    int temp = *xp;
    *xp = *yp;
    *yp = temp;
}

void bubbleSort(int arr[], int n)
{
    int i, j;
    for (i = 0; i < n-1; i++)

        for (j = 0; j < n-i-1; j++)
            if (arr[j] > arr[j+1])
                swap(&arr[j], &arr[j+1]);
}

void printArray(int arr[], int size)
{
    int i;
    for (i = 0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}


int main()
{
    int arr[100];
    int n;
    cout<<"No. of elements=";
    cin>>n;
    cout<<"enter the numbers:";
    for(int i=0;i<n;i++)
        cin>>arr[i];

    bubbleSort(arr, n);
    cout<<"Sorted array: \n";
    printArray(arr, n);
    return 0;
}
```

OUTPUT- Bubble sort:



```
No. of elements=7
enter the numbers:2
7
22
1
19
24
4
Sorted array:
1 2 4 7 19 22 24

Process returned 0 (0x0)   execution time : 13.045 s
Press any key to continue.
```

DRY RUN: Bubble sort:



Bubble sort

n = 7

arr

[0]	[1]	[2]	[3]	[4]	[5]	[6]
~~2~~	~~7~~	~~22~~	~~1~~	~~19~~	~~24~~	~~4~~
1	~~1~~	~~1~~	~~22~~	~~22~~	~~4~~	24
	2	~~7~~ 4	~~19~~	~~4~~	22	
			~~4~~ 7	19		

i → i<n-1

j → j<n-i-1

i	j	j<n-1-1	j<n-2-1	j<n-3-1	j<n-4-1	j<n-5-1
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	2	2	2	2	2	
3	3	3	3	3		
4	4	4	4			
5	5	5				
6	6					

sortedarray:

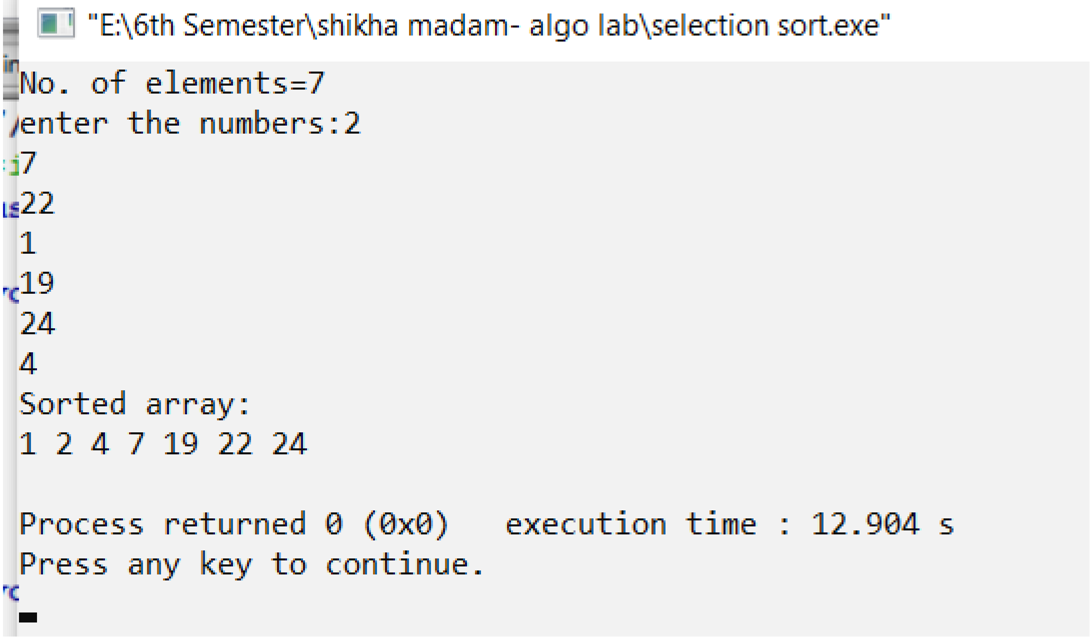
1	2	4	7	19	22	24

2. SELECTION SORT

CODE:

```
// C++ program for implementation of selection sort
#include <bits/stdc++.h>
using namespace std;

void swap(int *xp, int *yp)
{
    int temp = *xp;
    *xp = *yp;
    *yp = temp;
}

void selectionSort(int arr[], int n)
{
    int i, j, min_idx;

    for (i = 0; i < n-1; i++)
    {
        min_idx = i;
        for (j = i+1; j < n; j++)
            if (arr[j] < arr[min_idx])
                min_idx = j;

        swap(&arr[min_idx], &arr[i]);
    }
}

void printArray(int arr[], int size)
{
    int i;
    for (i=0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main()
{
  int arr[100];
    int n;
    cout<<"No. of elements=";
    cin>>n;
    cout<<"enter the numbers:";
    for(int i=0;i<n;i++)
        cin>>arr[i];

    selectionSort(arr, n);
    cout << "Sorted array: \n";
    printArray(arr, n);
    return 0;
}
```

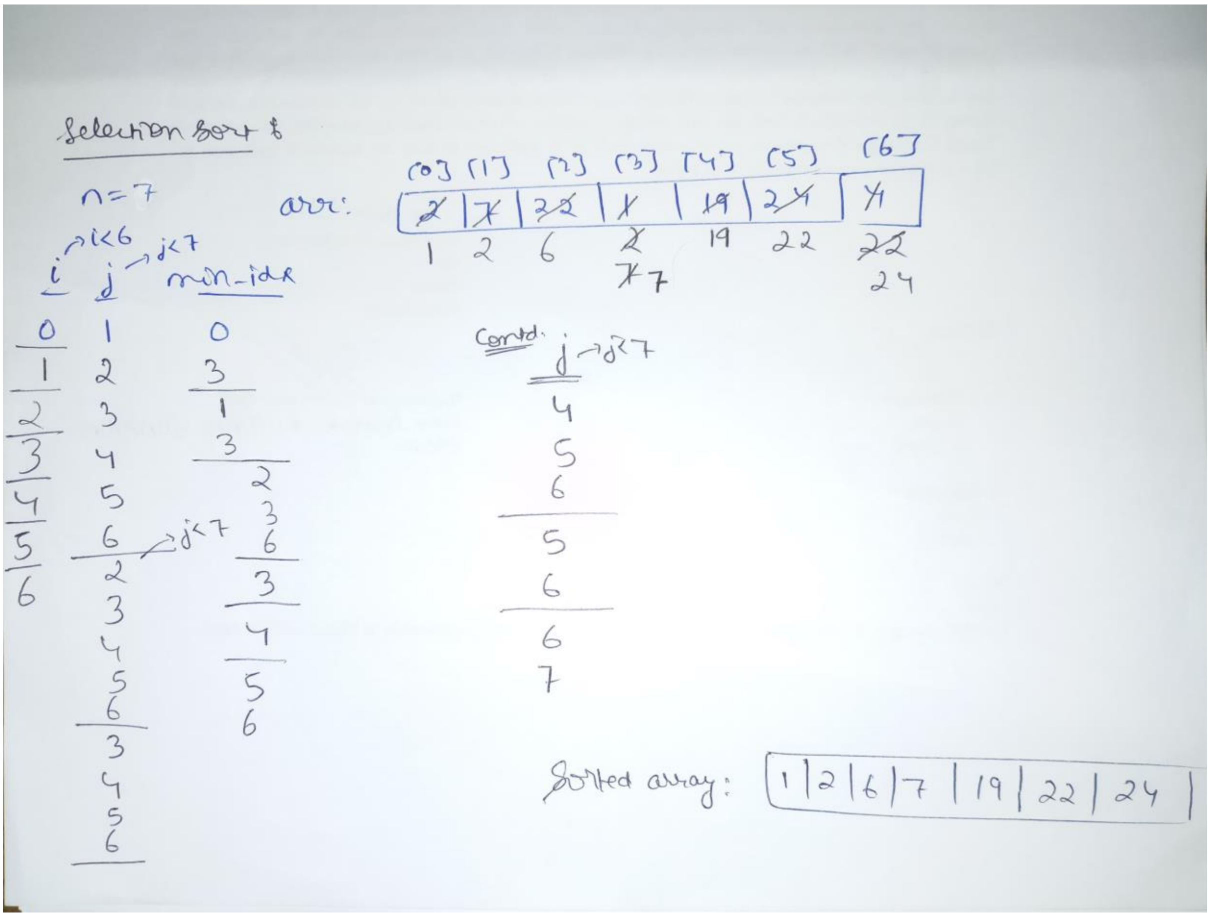
OUTPUT- Selection sort:



"E:\6th Semester\shikha madam- algo lab\selection sort.exe"

```
No. of elements=7
enter the numbers:2
7
22
1
19
24
4
Sorted array:
1 2 4 7 19 22 24

Process returned 0 (0x0)   execution time : 12.904 s
Press any key to continue.
```

DRY RUN- Selection sort:



Selection sort :

n=7

arr:

[0]	[1]	[2]	[3]	[4]	[5]	[6]
~~2~~	~~7~~	~~22~~	~~1~~	~~19~~	~~24~~	~~41~~
1	2	6	~~2~~	19	22	~~22~~
			~~7~~ 7			24

→i<6 →j<7

i	j	min_idx
0	1	0
1	2	3
2	3	1
3	4	3
4	5	2
5	6	3
6	2	6
	3	3
	4	4
	5	5
	6	6
	3	
	4	
	5	
	6	

→j<7

Contd.

j →j<7

4
5
6

5
6

6
7

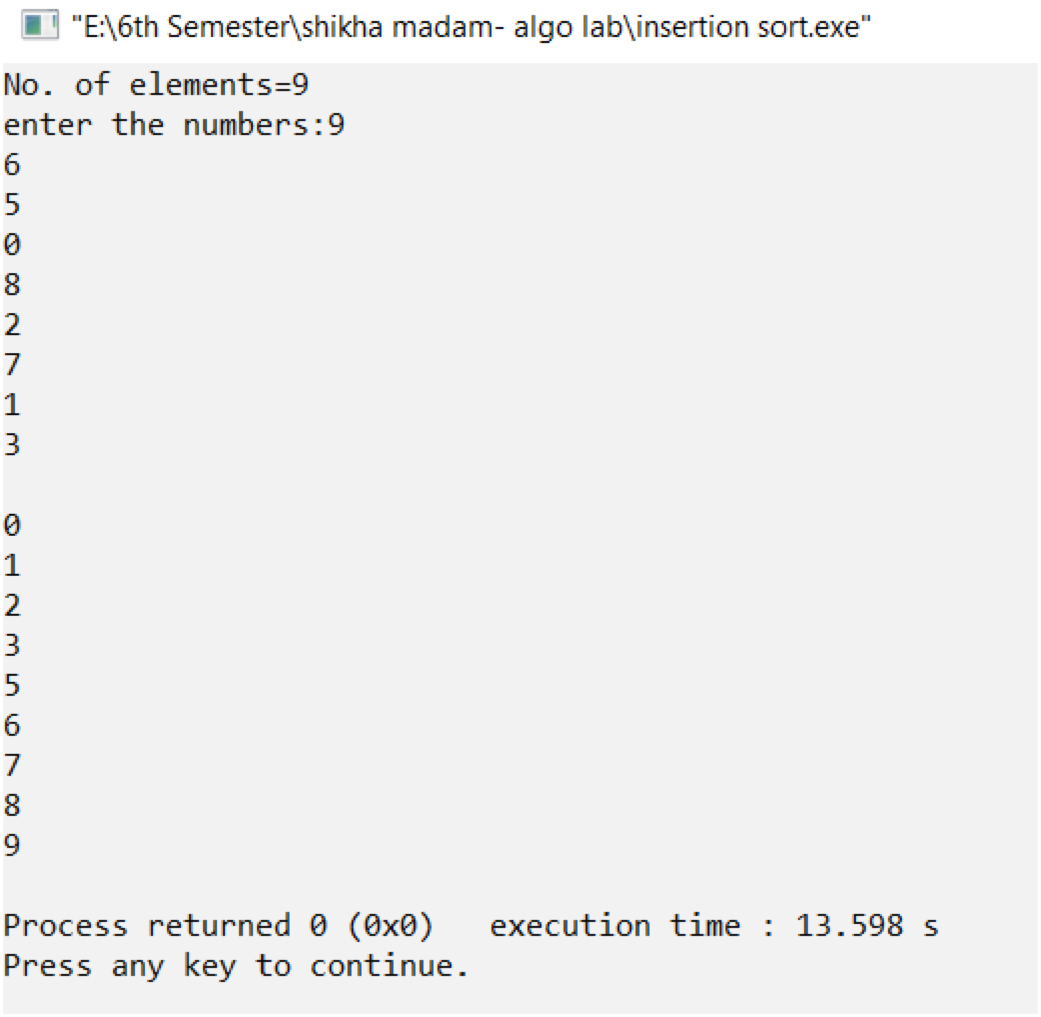
Sorted array: | 1 | 2 | 6 | 7 | 19 | 22 | 24 |

3. INSERTION SORT

CODE:

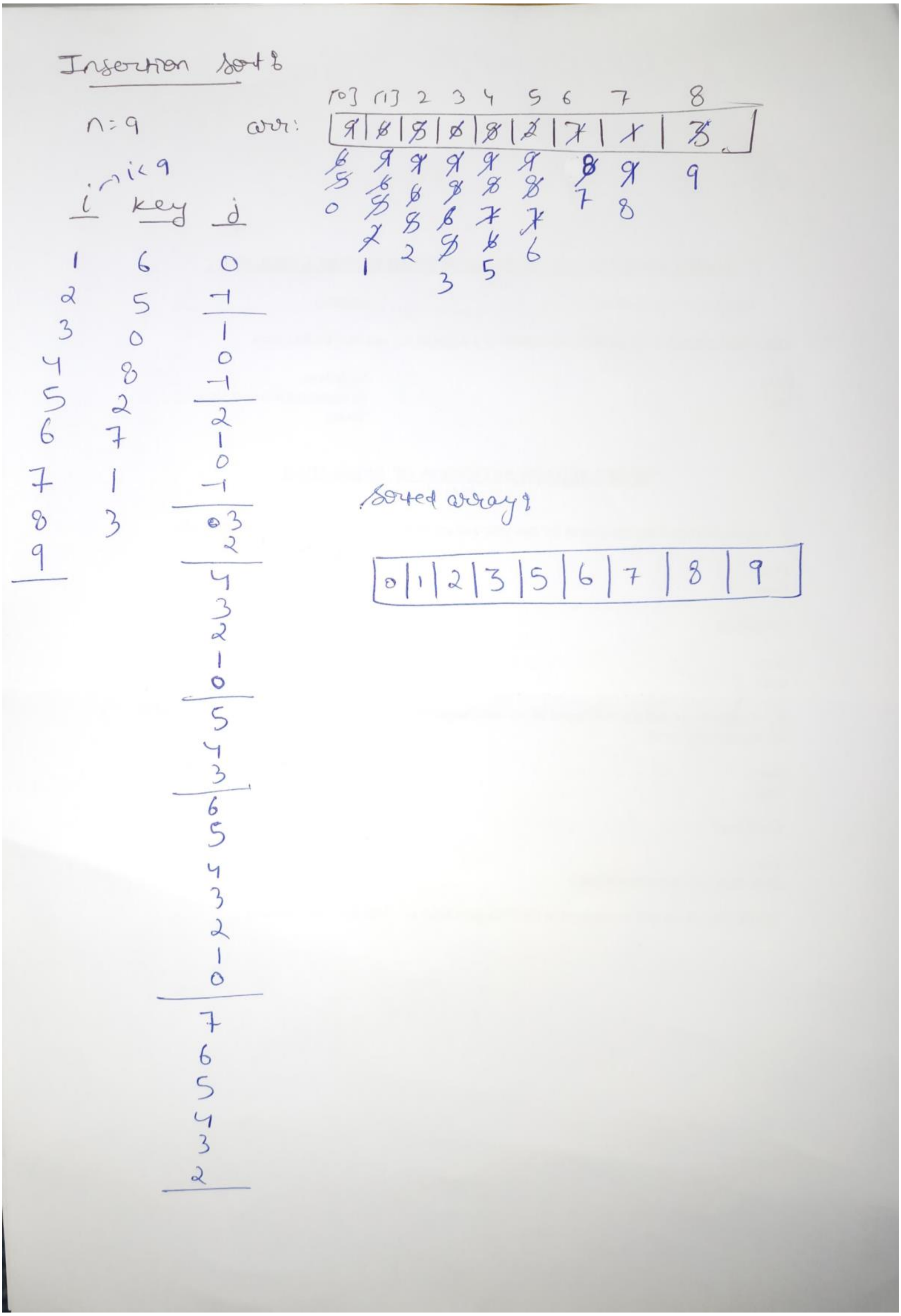
```
#include <bits/stdc++.h>
using namespace std;

/* Function to sort an array using insertion sort*/
void insertionSort(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;

        while (j >= 0 && arr[j] > key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}

void printArray(int arr[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        cout << "\n"<< arr[i] ;
    cout << endl;
}

int main()
{
    int arr[100];
    int n;
    cout<<"No. of elements=";
    cin>>n;
    cout<<"enter the numbers:";
    for(int i=0;i<n;i++)
        cin>>arr[i];

    insertionSort(arr, n);
    printArray(arr, n);

    return 0;
}
```

OUTPUT- Insertion sort:



```
No. of elements=9
enter the numbers:9
6
5
0
8
2
7
1
3

0
1
2
3
5
6
7
8
9

Process returned 0 (0x0)   execution time : 13.598 s
Press any key to continue.
```

DRY RUN- insertion sort:



Insertion sort:

n=9

i<9

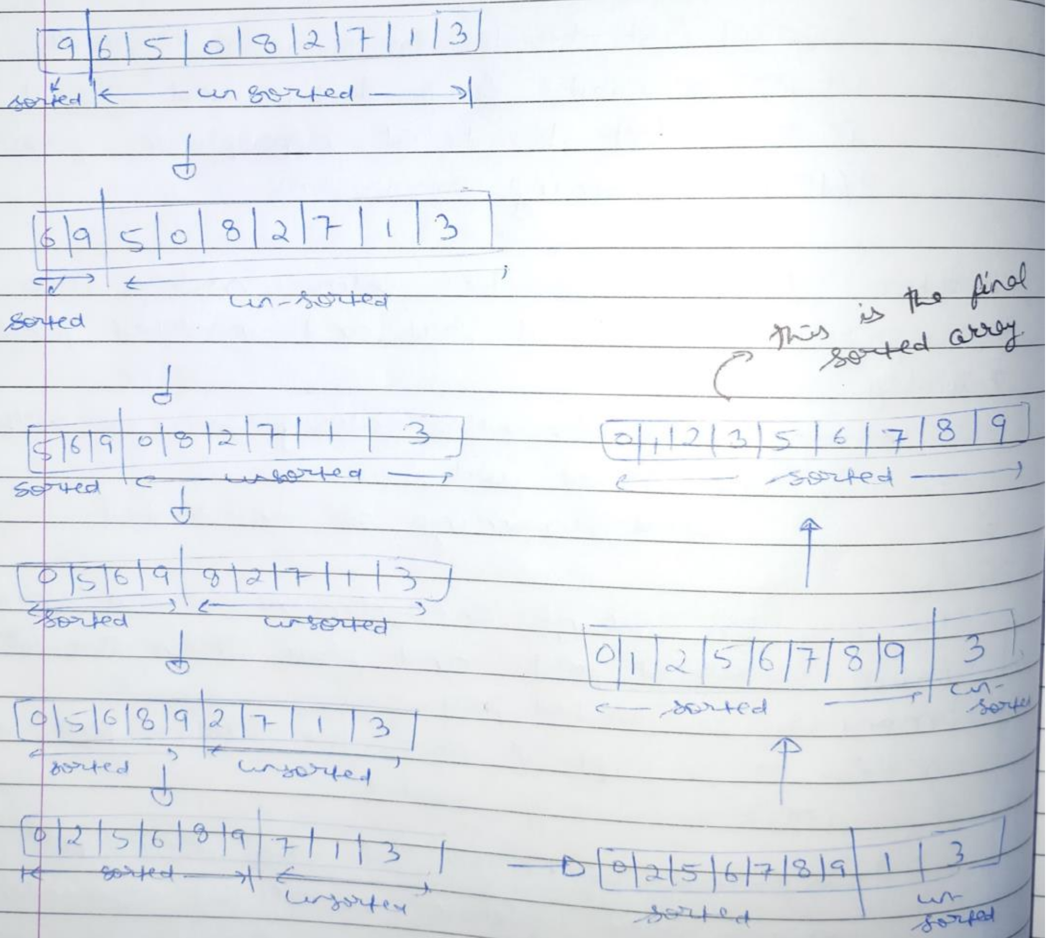
[0]	[1]	2	3	4	5	6	7	8
~~9~~	~~6~~	~~5~~	~~0~~	~~8~~	~~2~~	~~7~~	~~1~~	~~3~~
~~6~~	~~9~~	~~9~~	~~9~~	~~9~~	~~9~~	~~8~~	~~9~~	9
~~5~~	~~6~~	~~6~~	~~8~~	~~8~~	~~8~~	7	8	
0	~~5~~	~~5~~	~~6~~	~~7~~	~~7~~			
	~~2~~	2	~~5~~	~~6~~	6			
	1		3	5				

i	key	j
1	6	0 -1
2	5	1 0 -1
3	0	2 1 0 -1
4	8	3 2
5	2	4 3 2 1 0
6	7	5 4 3
7	1	6 5 4 3 2 1 0
8	3	7 6 5 4 3 2
9		

Sorted array:

0	1	2	3	5	6	7	8	9

by insertion sort



9 | 6 | 5 | 0 | 8 | 2 | 7 | 1 | 3

6 | 9 | 5 | 0 | 8 | 2 | 7 | 1 | 3

5 | 6 | 9 | 0 | 8 | 2 | 7 | 1 | 3

0 | 5 | 6 | 9 | 8 | 2 | 7 | 1 | 3

0 | 5 | 6 | 8 | 9 | 2 | 7 | 1 | 3

0 | 2 | 5 | 6 | 8 | 9 | 7 | 1 | 3

0 | 2 | 5 | 6 | 7 | 8 | 9 | 1 | 3

0 | 1 | 2 | 5 | 6 | 7 | 8 | 9 | 3

0 | 1 | 2 | 3 | 5 | 6 | 7 | 8 | 9

This is the final sorted array