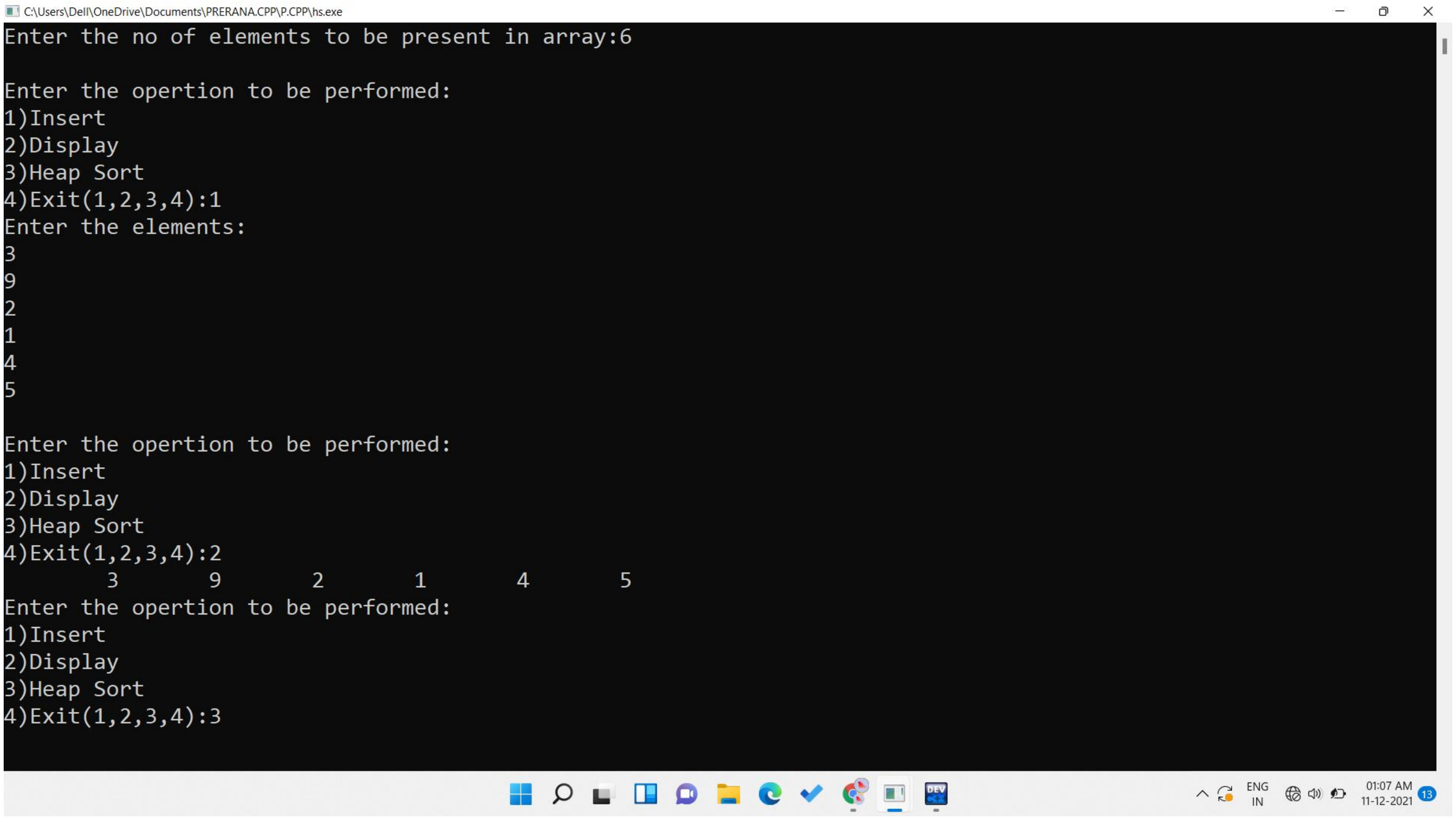
```
//
Assignment no-9
// Name- Prerana Rajesh Gajare      Class-SEIT
 RollNo-SI41
/*PROBLEM STATEMENT:-
            Implement Heap sort to sort a given
set of values using max or min-heap.
 */
//Source Code:-
#include <iostream>
using  namespace std;

//Class hs
class hs
{
    public:
    //Function declaration
    void getdata(int[],int);
    void max_heapify(int [],int,int);
    void min_heapify(int[],int,int);
    void heapsort(int[],int,int);
    void display(int [],int);
};

// TO ACCEPT DATA IN THE FORM OF ARRAY
void hs :: getdata(int heap[], int n)
{
    cout<<"Enter the elements:";
    for (int i = 0; i < n; i++)
    {
        cin >> heap[i];
    }
}

```

```
// TO PERFORM MAX HEAP OPERATION
void hs :: max_heapify(int heap[],int n,int i)
{
    int largest=i;// Initialize largest as root
    int left=2*i+1;//Position of left child
    int right=2*i+2;//Position of right child

     // If left child is greater than root ,set position of left as largest
    if (left<n && heap[left]>heap[largest])
        largest=left;
    else
        largest=i;

    // If right child is greater than root ,set position of right as largest
    if(right<n && heap[right]>heap[largest])
        largest=right;

    // If largest is not root
    if(largest!=i)
    {

        swap(heap[i],heap[largest]);

        // Recursively heapify the affected sub-tree
        max_heapify(heap,n,largest);
    }

}

//  TO PERFORM MIN HEAP OPERATION
void hs :: min_heapify(int heap[], int n,int i)
```

```
{

    int smallest = i; // Initialize smallest as root
    int left=2*i+1;//Position of left child
    int right=2*i+2;//Position of right child

    // If left child is smaller than root,set position of left as smallest
    if (left < n && heap[left]<heap[smallest])
        smallest = left;
    else
        smallest=i;

    // If right child is smaller than root ,set position of right as smallest
    if (right < n && heap[right]<heap[smallest])
        smallest = right;

    // If smallest is not root
    if (smallest != i) {
        swap(heap[i], heap[smallest]);

        // Recursively heapify the affected sub-tree
        min_heapify(heap, n, smallest);
    }

}

// TO PERFORM HEAPSORT OPERATION
void hs :: heapsort(int heap[], int n,int m)
{
    // Building heap
    for (int i=n/2-1;i>= 0;i--)
    {
        switch(m)
        {
            case 1:
                max_heapify(heap, n, i);
                break;
            case 2:
                min_heapify(heap,n,i);
                break;
        }

    }

//  To extract one by one element from heap
    for (int i=n-1; i>=0;i--)
    {
        //Swap the first and last index
        swap(heap[0], heap[i]);
        switch(m)
        {
            case 1:
                max_heapify(heap, i, 0);
                break;
            case 2:
                min_heapify(heap,i,0);
                break;
        }

    }

}

 // TO DISPLAY THE DATA IN FORM OF ARRAY
```

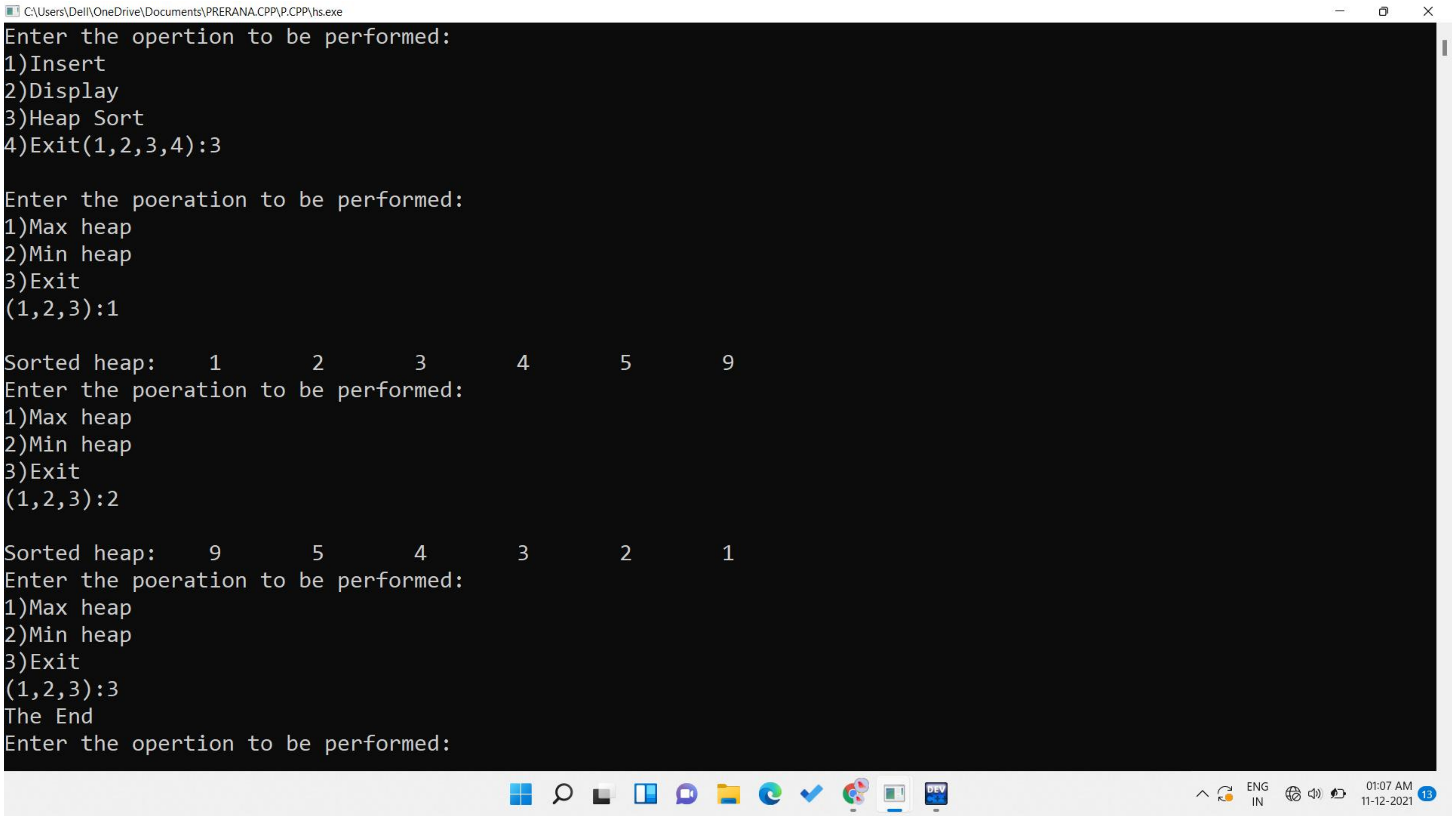
```cpp
void hs :: display(int heap[], int n)
{
    for (int i=0; i<n;i++)
    {
        cout<<"\t"<< heap[i];
    }
}


int main()
{

    int n;
    int heap[15];//Declaring an array heap of size 15
    hs h;// Creating object of class hs

    // Input the number of elements to be present in array.
    cout<<"Enter the no of elements to be present in array:";
    cin>>n;

    int l;
    do
    {
        cout<<"\nEnter the opertion to be performed:";
        cout<<"\n1)Insert";
        cout<<"\n2)Display";
        cout<<"\n3)Heap Sort";
        cout<<"\n4)Exit";
        cout<<"(1,2,3,4):";
        cin>>l;
        switch(l)
        {
            case 1:
                    h.getdata(heap,n);//Calling getdata function
                    break;
            case 2:
                    h.display(heap, n);//Calling display function
                    break;
            case 3:
                    int m;
                    do
                    {
                        cout<<"\nEnter the poeration to be performed:";
                        cout<<"\n1)Max heap";
                        cout<<"\n2)Min heap";
                        cout<<"\n3)Exit";
                        cout<<"\n(1,2,3):";
                        cin>>m;
                        switch(m)
                        {
                        case 1:

                                h.heapsort(heap,n, m);//Calling heapsort function to perform maxheap
                                cout<<"\nSorted heap:";
                                h.display(heap, n);//Display the elements of max heap
                                break;
                        case 2:

```

```cpp
                                h.heapsort(heap,n,m);//Calling heapsort function to perform minheap
                                cout<<"\nSorted heap:";
                                h.display(heap,n);//Display the elements of minheap
                                break;
                        case 3:
                                cout<<"The End";
                                break;
                        default:
                                cout<<"Wrong choice";
                    }
                }while(m!=3);
                break;
            case 4:
                    cout<<"The End";
                    break;
            default:
                    cout<<"Wrong Choice";
        }
    }while(l!=4);

}
```



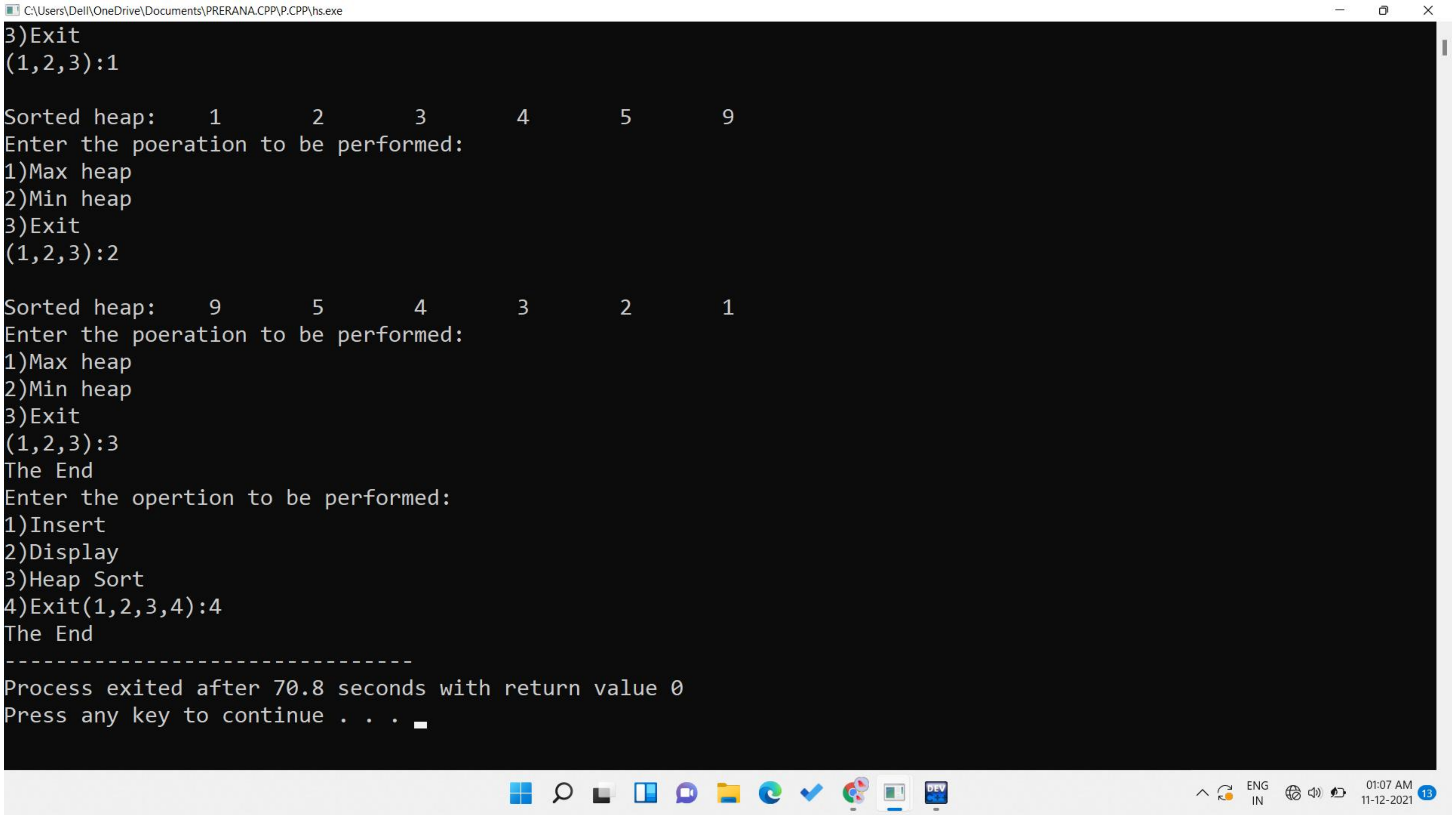
C:\Users\Dell\OneDrive\Documents\PRERANA.CPP\P.CPP\hs.exe

```
Enter the no of elements to be present in array:6

Enter the opertion to be performed:
1)Insert
2)Display
3)Heap Sort
4)Exit(1,2,3,4):1
Enter the elements:
3
9
2
1
4
5

Enter the opertion to be performed:
1)Insert
2)Display
3)Heap Sort
4)Exit(1,2,3,4):2
        3       9       2       1       4       5
Enter the opertion to be performed:
1)Insert
2)Display
3)Heap Sort
4)Exit(1,2,3,4):3
```



C:\Users\Dell\OneDrive\Documents\PRERANA.CPP\P.CPP\hs.exe

```
Enter the opertion to be performed:
1)Insert
2)Display
3)Heap Sort
4)Exit(1,2,3,4):3

Enter the poeration to be performed:
1)Max heap
2)Min heap
3)Exit
(1,2,3):1

Sorted heap:    1       2       3       4       5       9
Enter the poeration to be performed:
1)Max heap
2)Min heap
3)Exit
(1,2,3):2

Sorted heap:    9       5       4       3       2       1
Enter the poeration to be performed:
1)Max heap
2)Min heap
3)Exit
(1,2,3):3
The End
Enter the opertion to be performed:
```



```
3)Exit
(1,2,3):1

Sorted heap:    1       2       3       4       5       9
Enter the poeration to be performed:
1)Max heap
2)Min heap
3)Exit
(1,2,3):2

Sorted heap:    9       5       4       3       2       1
Enter the poeration to be performed:
1)Max heap
2)Min heap
3)Exit
(1,2,3):3
The End
Enter the opertion to be performed:
1)Insert
2)Display
3)Heap Sort
4)Exit(1,2,3,4):4
The End
--------------------------------
Process exited after 70.8 seconds with return value 0
Press any key to continue . . .
```