Heap Sort:

```
#include<iostream>
#include<conio.h>

using namespace std;

class BinaryMinHeap{
	public:
		int *data;
		int heapS;
		int arrS;

		int gtLeftChild(int indN){
			return (2 * indN) + 1;
		}

		int gtRightChild(int indN){
			return (2 * indN) + 2;
		}

		int getPind(int indN){
			return (indN - 1) / 2;
		}

		BinaryMinHeap(int size){
			data = new int[size];
			heapS = 0;
			arrS = size;
		}

		int getMinimum(){
			if(isEmpty() == true){
				cout << "\nHeap is Empty.";
			}else{
				return data[0];
			}
			return -1;
		}

		void display(){
			if(isEmpty() == true){
				cout << "\nHeap is Empty.";
			}
			else{
				cout << "\nElements are Present in Heap are ";
				for(int i = 0; i < heapS; i++)
					cout << data[i] << " ";
			}
		}

		void insert(int val){
			cout << "\nYou want to insert this value " << val << " to
Heap.";
			if(heapS == arrS){
				cout << "\nSorry to Inform you Heap is Full.";
```

```
		}
		else{
			data[heapS] = val;
			heapS++;
			reheapUp(heapS - 1);
		}
	}

	void reheapUp(int nodI){
		int parN, temp;
		if(nodI != 0){
			parN = getPind(nodI);
			if(data[parN] > data[nodI]){
				temp = data[parN];
				data[parN] = data[nodI];
				data[nodI] = temp;
				reheapUp(parN);
			}
		}
	}

	bool isEmpty(){
		if(heapS == 0){
			return true;
		}
		return false;
	}

	void remove(){
		if(isEmpty() == true){
			cout << "\nHeap is Empty.";
		}
		else{
			cout << "\nRemoved Element is" << data[0] << " from
Heap.";
			data[0] = data[heapS - 1];
			heapS--;
			reheapDown(0);
		}
	}

	void reheapDown(int nodI){
		int lc,rc,mInd,temp;
		lc = gtLeftChild(nodI);
		rc = gtRightChild(nodI);
		if(rc >= heapS){
			if(lc >= heapS){
				return;
			}else{
				mInd = lc;
			}
		}else{
			if(data[lc] < data[rc]){
				mInd = lc;
			}else{
				mInd = rc;
```

```
                        }
                }
                if(data[mInd] < data[nodI]){
                        temp = data[mInd];
                        data[mInd] = data[nodI];
                        data[nodI] = temp;
                }
        }

        void sort(){
                cout << "\nSorted Heap:";
                int oheapS = heapS;
                int sortArr[heapS], i;
                for(i = 0; i < oheapS; i++){
                        sortArr[i] = data[0];
                        cout << sortArr[i] << " ";
                        remove();
                }
        }

        void checkSpace()
        {
                if(heapS==0)
                {
                        cout << "\nHeap is Empty.";
                }
                else if(heapS == arrS)
                {
                        cout << "\nSorry to Inform you Heap is Full.";
                }
                else
                {
                        int k = arrS - heapS;
                        cout << "\nYou can add " << k << " more Elements in
the given heap with their size is " << arrS;
                }
        }
};

int main(){
        int size;
        int k;
        cout << "\nTo Create Heap Press 1:";
        cin >> k;
        if (k != 1){
                return 0;
        }
        cout << "\nEnter size of Heap: ";
   cin >> size;
   BinaryMinHeap bn(size);
        int ch, p;
         cout << "1) Insert element to Heap: " << endl;
   cout << "2) Delete element from Heap: " << endl;
   cout << "3) Display all the elements of Heap:" << endl;
   cout << "4) Display all the elements in Sorted form in Heap: " << endl;
   cout << "5) Check Available Space in the Heap: " << endl;
```

```
    cout << "6) Display Minimum Element in the Heap: " << endl;
        cout << "7) Exit" << endl;

    do {
       cout << "\nEnter your choice : " << endl;
       cin >> ch;
       switch (ch) {
       case 1:
          cout << "\nEnter Element you Want insert in the Heap : ";
          cin >> p;
          bn.insert(p);
          break;
       case 2:
          cout << "\nBefore Element removed: ";
          bn.display();
          bn.remove();
          cout << "\nAfter Element removed: ";
          bn.display();
          break;
       case 3:
          cout << "\nDisplay Elements in the Heap: ";
          bn.display();
          break;
       case 4:
          cout << "\nDisplay Sorted Heap: ";
          bn.sort();
          break;
       case 5:
          cout << "\nAvailable Space in the Heap is ";
          bn.checkSpace();
          break;
       case 6:
          cout << "\nDisplay Minimum Element in Min Heap: " << bn.getMinimum();
          break;
                  case 7:
          exit(0);
       default:
          cout << "Invalid choice" << endl;
       }
    } while (ch != 7);
        return 0;
}
```

Output:

```
To Create Heap Press 1:1

Enter size of Heap: 10
1) Insert element to Heap:
2) Delete element from Heap:
3) Display all the elements of Heap:
4) Display all the elements in Sorted form in Heap:
5) Check Available Space in the Heap:
6) Display Minimum Element in the Heap:
7) Exit

Enter your choice :
1

Enter Element you Want insert in the Heap : 22

You want to insert this value 22 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 23

You want to insert this value 23 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 34

You want to insert this value 34 to Heap.
```

```
Enter your choice :
1

Enter Element you Want insert in the Heap : 54

You want to insert this value 54 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 11

You want to insert this value 11 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 10

You want to insert this value 10 to Heap.
Enter your choice :
2

Before Element removed:
Elements are Present in Heap are 10 22 11 54 23 34
Removed Element is10 from Heap.
After Element removed:
Elements are Present in Heap are 11 22 34 54 23
Enter your choice :
3

Display Elements in the Heap:
```

```
4

Display Sorted Heap:
Sorted Heap:11
Removed Element is11 from Heap.22
Removed Element is22 from Heap.23
Removed Element is23 from Heap.34
Removed Element is34 from Heap.54
Removed Element is54 from Heap.
Enter your choice :
5

Available Space in the Heap is
Heap is Empty.
Enter your choice :
1

Enter Element you Want insert in the Heap : 11

You want to insert this value 11 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 33

You want to insert this value 33 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 21
```

```
Enter Element you Want insert in the Heap : 21

You want to insert this value 21 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 54

You want to insert this value 54 to Heap.
Enter your choice :
1

Enter Element you Want insert in the Heap : 10

You want to insert this value 10 to Heap.
Enter your choice :
3

Display Elements in the Heap:
Elements are Present in Heap are 10 11 21 54 33
Enter your choice :
5

Available Space in the Heap is
You can add 5 more Elements in the given heap with their size is 10
Enter your choice :
6

Display Minimum Element in Min Heap: 10
```