

lab_11.cpp

```
1 // <----Lab 11 - Heap----->
2
3 #include <iostream>
4 #include <vector>
5 using namespace std;
6
7 void swap(int *a, int *b)
8 {
9     int temp = *b;
10    *b = *a;
11    *a = temp;
12 }
13
14 void heapifyMax(vector<int> &hT, int i, int size)
15 {
16     int largest = i;
17     int l = 2 * i + 1;
18     int r = 2 * i + 2;
19
20     if (l < size && hT[l] > hT[largest])
21         largest = l;
22     if (r < size && hT[r] > hT[largest])
23         largest = r;
24
25     if (largest != i)
26     {
27         swap(&hT[i], &hT[largest]);
28         heapifyMax(hT, largest, size);
29     }
30 }
31
32 void insertMax(vector<int> &hT, int newNum)
33 {
34     int size = hT.size();
35     if (size == 0)
36         hT.push_back(newNum);
37     else
38     {
39         hT.push_back(newNum);
40         for (int i = size / 2 - 1; i >= 0; i--)
41             heapifyMax(hT, i, size);
42     }
43 }
44
45 void deleteNodeMax(vector<int> &hT, int num)
46 {
47     int size = hT.size();
48     int i;
49     for (i = 0; i < size; i++)
50     {
51         if (num == hT[i])
52             break;
53     }
```

```
54     swap(&hT[i], &hT[size - 1]);
55     hT.pop_back();
56     for (int i = size / 2 - 1; i >= 0; i--)
57         heapifyMax(hT, i, size - 1);
58 }
59
60 // Q1. The above code is for Max Heap. Implement a Min Heap and then insert and delete
    elements.
61
62 void heapifyMin(vector<int> &hT, int i, int size)
63 {
64     int smallest = i;
65     int l = 2 * i + 1;
66     int r = 2 * i + 2;
67
68     if (l < size && hT[l] < hT[smallest])
69         smallest = l;
70     if (r < size && hT[r] < hT[smallest])
71         smallest = r;
72
73     if (smallest != i)
74     {
75         swap(&hT[i], &hT[smallest]);
76         heapifyMin(hT, smallest, size);
77     }
78 }
79
80 void insertMin(vector<int> &hT, int newNum)
81 {
82     int size = hT.size();
83     if (size == 0)
84         hT.push_back(newNum);
85     else
86     {
87         hT.push_back(newNum);
88         int i = size;
89         while (i > 0 && hT[(i - 1) / 2] > hT[i])
90         {
91             swap(&hT[i], &hT[(i - 1) / 2]);
92             i = (i - 1) / 2;
93         }
94     }
95 }
96
97 void deleteNodeMin(vector<int> &hT, int num)
98 {
99     int size = hT.size();
100     int i;
101     for (i = 0; i < size; i++)
102     {
103         if (num == hT[i])
104             break;
105     }
106     swap(&hT[i], &hT[size - 1]);
107     hT.pop_back();
108     for (int i = size / 2 - 1; i >= 0; i--)
```

```
109         heapifyMin(hT, i, size - 1);
110     }
111
112     // Q2. Implement peek and extract operations for both max and min values.
113     int extractMin(vector<int> &hT)
114     {
115         if (hT.size() == 0)
116         {
117             cout << "Heap is empty\n";
118             return -1;
119         }
120
121         int minVal = hT[0];
122         deleteNodeMin(hT, hT[0]);
123         return minVal;
124     }
125
126     int peekMin(const vector<int> &hT)
127     {
128         if (hT.size() == 0)
129         {
130             cout << "Heap is empty\n";
131             return -1;
132         }
133         return hT[0];
134     }
135
136     int peekMax(const vector<int> &hT)
137     {
138         if (hT.size() == 0)
139         {
140             cout << "Heap is empty\n";
141             return -1;
142         }
143         return hT[0];
144     }
145     int extractMax(vector<int> &hT)
146     {
147         if (hT.size() == 0)
148         {
149             cout << "Heap is empty\n";
150             return -1;
151         }
152
153         int maxVal = hT[0];
154         deleteNodeMax(hT, hT[0]);
155         return maxVal;
156     }
157
158     void printArray(const vector<int> &hT)
159     {
160         for (int i = 0; i < hT.size(); ++i)
161             cout << hT[i] << " ";
162         cout << "\n";
163     }
164
```

```
165 int main()
166 {
167     vector<int> maxHeap;
168     insertMax(maxHeap, 3);
169     insertMax(maxHeap, 4);
170     insertMax(maxHeap, 9);
171     insertMax(maxHeap, 5);
172     insertMax(maxHeap, 2);
173
174     cout << "Max-Heap array: ";
175     printArray(maxHeap);
176
177     deleteNodeMax(maxHeap, 4);
178     cout << "After deleting an element in Max-Heap: ";
179     printArray(maxHeap);
180
181     cout << "Max-Heap peek: " << peekMax(maxHeap) << endl;
182     cout << "Extract Max value: " << extractMax(maxHeap) << endl;
183     cout << "Max-Heap array after extraction: ";
184     printArray(maxHeap);
185
186     vector<int> minHeap;
187     insertMin(minHeap, 3);
188     insertMin(minHeap, 4);
189     insertMin(minHeap, 9);
190     insertMin(minHeap, 5);
191     insertMin(minHeap, 2);
192
193     cout << "\nMin-Heap array: ";
194     printArray(minHeap);
195
196     deleteNodeMin(minHeap, 4);
197     cout << "After deleting an element in Min-Heap: ";
198     printArray(minHeap);
199
200     cout << "Min-Heap peek: " << peekMin(minHeap) << endl;
201     cout << "Extract Min value: " << extractMin(minHeap) << endl;
202     cout << "Min-Heap array after extraction: ";
203     printArray(minHeap);
204
205     return 0;
206 }
207
```