

```

1: #include<iostream>
2: #include<cmath>
3: using namespace std;
4:
5: void swap(int arr[], int n, int i)
6: {
7:     int temp = arr[n];
8:     arr[n] = arr[i];
9:     arr[i] = temp;
10: }
11:
12: void heapify(int arr[], int n, int i)
13: {
14:     int l_c = 2 * i + 1;
15:     int r_c = 2 * i + 2;
16:     int large = i;
17:
18:     if(l_c < n && arr[l_c] > arr[large])
19:     {
20:         large = l_c;
21:     }
22:     if(r_c < n && arr[r_c] > arr[large])
23:     {
24:         large = r_c;
25:     }
26:
27:     if(large != i)
28:     {
29:         swap(arr, i, large);
30:         heapify(arr, n, large);
31:     }
32: }
33:
34: void build_heap(int arr[], int n)
35: {
36:     for(int i = (n - 1) / 2; i >= 0; i--)
37:     {
38:         heapify(arr, n, i);
39:     }
40: }
41:
42: int height(int arr[], int n)
43: {
44:     int i = 0;
45:     int height = 0;
46:

```

```

47:     while(i < n)
48:     {
49:         height++;
50:         i = 2 * i + 1;
51:     }
52:     return height;
53: }
54:
55: void heapTree(int arr[], int n)
56: {
57:     int h = height(arr, n);
58:     int blocks = pow(2, h);
59:     int temp_blocks = blocks;
60:     int index = 0;
61:     int k = 3;
62:
63:     for(int i = 0; i < h; i++)
64:     {
65:         for(int j = 0; j < blocks; j++)
66:         {
67:             if(j == temp_blocks / 2 && index < n)
68:             {
69:                 cout << arr[index];
70:                 index++;
71:             }
72:             else
73:             {
74:                 if(j == (temp_blocks * k) / 2 && index < n)
75:                 {
76:                     cout << arr[index];
77:                     index++;
78:                     k += 2;
79:                 }
80:                 else
81:                 {
82:                     cout << " ";
83:                 }
84:             }
85:         }
86:         cout << "\n";
87:         temp_blocks = temp_blocks / 2;
88:         k = 3;
89:     }
90: }
91:
92: int main()

```

```

93: {
94:     int n;
95:     cout << "Enter the length:\n";
96:     cin >> n;
97:
98:     int arr[n];
99:     cout << "Enter elements: \n";
100:    for(int i = 0; i < n; i++)
101:    {
102:        cin >> arr[i];
103:    }
104:
105:    build_heap(arr, n);
106:    heapTree(arr, n);
107:
108:    /*
109:    int h = height(arr, n);
110:    int blocks = pow(2, h);
111:    int temp_blocks = blocks;
112:    int index = 0;
113:    int printed_index;
114:    int k = 3;
115:
116:    for(int i = 0; i < h; i++)
117:    {
118:        for(int j = 0; j < blocks; j++)
119:        {
120:            if(j == temp_blocks / 2 && index < n)
121:            {
122:                cout << arr[index];
123:                index++;
124:                printed_index = j;
125:            }
126:            else
127:            {
128:                if(j == (temp_blocks * k) / 2 && index < n)
129:                {
130:                    cout << arr[index];
131:                    index++;
132:                    k += 2;
133:                }
134:                else
135:                {
136:                    cout << " ";
137:                }
138:            }

```

```
139:     }
140:     cout << "\n";
141:     printed_index = 0;
142:     temp_blocks = temp_blocks / 2;
143:     k = 3;
144: }*/
145:
146: return 0;
147: }
```