```
class MinHeap {
private:
    vector<int> heap;

    void heapifyUp(int i) {
        while (i > 0 && heap[i] < heap[(i - 1) / 2]) {
            swap(heap[i], heap[(i - 1) / 2]);
            i = (i - 1) / 2;
        }
    }

    void heapifyDown(int i) {
        int n = heap.size();
        while (2 * i + 1 < n) {
            int smallest = i;
            int left = 2 * i + 1, right = 2 * i + 2;
            if (left < n && heap[left] < heap[smallest]) smallest = left;
            if (right < n && heap[right] < heap[smallest]) smallest =
right;
            if (smallest == i) break;
            swap(heap[i], heap[smallest]);
            i = smallest;
        }
    }

public:
    void insert(int val) {
        heap.push_back(val);
        heapifyUp(heap.size() - 1);
    }

    void removeMin() {
        if (heap.empty()) return;
        heap[0] = heap.back();
        heap.pop_back();
        heapifyDown(0);
    }

    int getMin() {
        return heap.empty() ? -1 : heap[0];
    }

    void print() {
        for (int x : heap)
            cout << x << " ";
        cout << endl;
    }
};

int main() {
    MinHeap h;
    h.insert(5);
    h.insert(3);
    h.insert(8);
    h.insert(1);

    h.print(); // 1 3 8 5 (có thể khác tùy cách xây)
    h.removeMin();
    h.print(); // 3 5 8

    return 0;
}
```
