```
#include <iostream>
#include <vector>
using namespace std;


void swap(int *a, int *b) {
  int temp = *b;
  *b = *a;
  *a = temp;
}


void heapify(vector<int> &hT, int i) {
  int size = hT.size();


  int largest = i;
  int l = 2 * i + 1;
  int r = 2 * i + 2;
  if (l < size && hT[l] > hT[largest])
    largest = l;
  if (r < size && hT[r] > hT[largest])
    largest = r;


  if (largest != i) {
    swap(&hT[i], &hT[largest]);
    heapify(hT, largest);
  }
}

void insert(vector<int> &hT, int newNum) {
  int size = hT.size();
  if (size == 0) {
    hT.push_back(newNum);
  } else {
    hT.push_back(newNum);
    for (int i = size / 2 - 1; i >= 0; i--) {
      heapify(hT, i);
    }
  }
}


void deleteNode(vector<int> &hT, int num) {
  int size = hT.size();
  int i;
  for (i = 0; i < size; i++) {
    if (num == hT[i])
      break;
  }
  swap(&hT[i], &hT[size - 1]);

  hT.pop_back();
  for (int i = size / 2 - 1; i >= 0; i--) {
    heapify(hT, i);
  }
}


void printArray(vector<int> &hT) {
  for (int i = 0; i < hT.size(); ++i)
    cout << hT[i] << " ";
  cout << "\n";
}


int main() {
  vector<int> heapTree;

  insert(heapTree, 3);
  insert(heapTree, 4);
  insert(heapTree, 9);
  insert(heapTree, 5);
  insert(heapTree, 2);

  cout << "Max-Heap array: ";
  printArray(heapTree);

  deleteNode(heapTree, 4);

  cout << "After deleting an element: ";

  printArray(heapTree);
}
```