堆的实现【C++】

```cpp
template<typename T>
class Heap {
public:
    Heap() {}
    template<typename FuncT>
    Heap(FuncT cmpFunc) {
        data = vector<T>();
        cmp = cmpFunc;
    }
    template<typename FuncT>
    Heap(vector<T> arr, FuncT cmpFunc) {
        data = arr;
        cmp = cmpFunc;
        heapify();
    }
    void push(T val) {
        data.push_back(val);
        siftUp(data.size() - 1);
    }
    T top() { return data[0]; }
    T pop() {
        T val = top();
        data[0] = data[data.size() - 1];
        data.pop_back();
        siftDown(0);
        return val;
    }
    bool empty() { return data.empty(); }
    int size() { return data.size(); }
private:
    // 将 data[idx] 向上调整
    void siftUp(int idx) {
        while (idx > 0 && cmp(data[idx], data[(idx - 1) / 2])) {
            // father = (idx - 1) / 2
            swap(data[(idx - 1) / 2], data[idx]);
            idx = (idx - 1) / 2;
        }
    }
    void siftDown(int idx) {
        while (idx * 2 + 1 < data.size()) {
            int lc = idx * 2 + 1;
            int rc = idx * 2 + 2;
            int tmpidx = idx;
            if (lc < data.size() && cmp(data[lc], data[tmpidx])) tmpidx = lc;
            if (rc < data.size() && cmp(data[rc], data[tmpidx])) tmpidx = rc;
            if (tmpidx == idx) break;
            swap(data[tmpidx], data[idx]);
            idx = tmpidx;
        }
    }
    void heapify() {
        int idx = (data.size() - 1) / 2;
        for (int i = idx; i >= 0; i--) {
            siftDown(i);
        }
    }
private:
    vector<T> data;
    function<bool(T, T)> cmp;
};
```

剑指 Offer 40. 最小的k个数

https://leetcode-cn.com/problems/zui-xiao-de-kge-shu-lcof/

```
class Solution {
public:
    vector<int> getLeastNumbers(vector<int>& arr, int k) {
        // Plan A: 最小堆 弹出k次
        vector<int> ans;
        Heap<int> h{arr, less<int>()};
        for (int i = 0; i < k; i++) {
            ans.push_back(h.pop());
        }
        return ans;
        // Plan B: 最大堆 始终维护 k 个数
        vector<int> ans;
        Heap<int> h{greater<int>()};
        for (int i = 0; i < arr.size(); i++) {
            h.push(arr[i]);
            if (h.size() > k) h.pop();
        }
        while (!h.empty()) ans.push_back(h.pop());
        return ans;
    }
};
```

1046. 最后一块石头的重量

https://leetcode-cn.com/problems/last-stone-weight/

```
class Solution {
public:
    int lastStoneWeight(vector<int>& stones) {
        Heap<int> h{stones, greater<int>()};
        while (h.size() > 1) {
            int x = h.pop();
            int y = h.pop();
            if (x != y) h.push(x - y);
        }
        return h.size() == 1 ? h.top() : 0;
    }
};
```

215. 数组中的第K个最大元素

https://leetcode-cn.com/problems/kth-largest-element-in-an-array/

```
class Solution {
public:
    int findKthLargest(vector<int>& nums, int k) {
        Heap<int> h{less<int>()};
        for (int i = 0; i < nums.size(); i++) {
            h.push(nums[i]);
            if (h.size() > k) h.pop();
        }
        return h.top();
    }
};
```

703. 数据流中的第 K 大元素

https://leetcode-cn.com/problems/kth-largest-element-in-a-stream/

```
class KthLargest {
public:
    KthLargest(int k, vector<int>& nums) {
        h = Heap<int>{less<int>()};
        this->k = k;
        for (int i = 0; i < nums.size(); i++) {
            add(nums[i]);
        }
    }
    int add(int val) {
        h.push(val);
        if (h.size() > k) h.pop();
        return h.top();
    }
private:
    Heap<int> h;
    int k;
};
```

692. 前K个高频单词

https://leetcode-cn.com/problems/top-k-frequent-words/

```
struct Node {
    string s;
    int freq;
};

bool cmp(Node a, Node b) {
    if (a.freq == b.freq) {
        return a.s < b.s;
    }
    return a.freq > b.freq;
}

class Solution {
public:
    vector<string> topKFrequent(vector<string>& words, int k) {
        map<string, int> cnt;
        for (int i = 0; i < words.size(); i++) {
            cnt[words[i]]++;
        }
        Heap<Node> h{cmp};
        map<string, int> :: iterator iter;
        for (iter = cnt.begin(); iter != cnt.end(); iter++) {
            h.push(Node{iter->first, iter->second});
        }
        vector<string> ans;
        for (int i = 0; i < k; i++) ans.push_back(h.pop().s);
        return ans;
    }
};
```

295. 数据流的中位数

https://leetcode-cn.com/problems/find-median-from-data-stream/

```
class MedianFinder {
public:
    MedianFinder() {
        // smallNum 大顶堆 && largeNum 小顶堆
        smallNum = Heap<int>{greater<int>()};
        largeNum = Heap<int>{less<int>()};
    }

    void addNum(int num) {
        if (smallNum.empty() || num <= smallNum.top()) {
            smallNum.push(num);
            if (smallNum.size() - largeNum.size() > 1) {
                largeNum.push(smallNum.pop());
            }
        }
        else {
            largeNum.push(num);
            if (largeNum.size() - smallNum.size() >= 1) {
                smallNum.push(largeNum.pop());
            }
        }
    }
    double findMedian() {
        if (largeNum.size() == smallNum.size()) {
            return 0.5 * (largeNum.top() + smallNum.top());
        }
        return smallNum.top();
    }
private:
    Heap<int> smallNum, largeNum;
};
```

264. 丑数 II

https://leetcode-cn.com/problems/ugly-number-ii/

```
class Solution {
public:
    int nthUglyNumber(int n) {
        set<long long> uglyNumber;
        Heap<long long> h{less<long long>()};
        h.push(1);
        uglyNumber.insert(1);
        for (int i = 0; i < n - 1; i++) {
            long long val = h.pop();
            if (!uglyNumber.count(val * 2)) {
                h.push(val * 2);
                uglyNumber.insert(val * 2);
            }
            if (!uglyNumber.count(val * 3)) {
                h.push(val * 3);
                uglyNumber.insert(val * 3);
            }
            if (!uglyNumber.count(val * 5)) {
                h.push(val * 5);
                uglyNumber.insert(val * 5);
            }
        }
        return h.top();
    }
};
```

373. 查找和最小的 K 对数字

https://leetcode-cn.com/problems/find-k-pairs-with-smallest-sums/

超时代码

```
bool cmp(vector<int> a, vector<int> b) {
    return a[0] + a[1] > b[0] + b[1];
}

class Solution {
public:
    vector<vector<int>> kSmallestPairs(vector<int>& nums1, vector<int>& nums2, int k) {
        Heap<vector<int>> h{cmp};
        for (int i = 0; i < nums1.size(); i++) {
            for (int j = 0; j < nums2.size(); j++) {
                h.push(vector<int>{nums1[i], nums2[j]});
                if (h.size() > k) h.pop();
            }
        }
        vector<vector<int>> ans;
        while (!h.empty()) {
            ans.push_back(h.pop());
        }
        return ans;
    }
};
```

正确代码

```
struct Node {
    int x, y;
    int idx;
};

bool cmp(Node a, Node b) {
    return a.x + a.y < b.x + b.y;
}

class Solution {
public:
    vector<vector<int>> kSmallestPairs(vector<int>& nums1, vector<int>& nums2, int k) {
        Heap<Node> h{cmp};
        for (int i = 0; i < nums1.size(); i++) {
            h.push(Node{nums1[i], nums2[0], 0});
        }
        vector<vector<int>> ans;
        for (int i = 0; i < k; i++) {
            if (h.empty()) break;
            Node node = h.pop();
            ans.push_back(vector<int>{node.x, node.y});
            node.idx++;
            if (node.idx == nums2.size()) continue;
            node.y = nums2[node.idx];
            h.push(node);
        }

        return ans;
    }
};
```