using namespace std;

#include <iostream>

#include <queue>

#include <vector>

class KthSmallestInMSortedArrays {

public:

struct valueCompare {

bool operator()(const pair<int, pair<int, int>> &x, const pair<int, pair<int, int>> &y) {

return x.first > y.first;

}

};

static int findKthSmallest(const vector<vector<int>> &lists, int k) {

priority\_queue<pair<int, pair<int, int>>, vector<pair<int, pair<int, int>>>, valueCompare>

minHeap;

// put the 1st element of each array in the min heap

for (int i = 0; i < lists.size(); i++) {

if (!lists[i].empty()) {

minHeap.push(make\_pair(lists[i][0], make\_pair(i, 0)));

}

}

// take the smallest (top) element form the min heap, if the running count is equal to k return

// the number if the array of the top element has more elements, add the next element to the

// heap

int numberCount = 0, result = 0;

while (!minHeap.empty()) {

auto node = minHeap.top();

minHeap.pop();

result = node.first;

if (++numberCount == k) {

break;

}

node.second.second++;

if (lists[node.second.first].size() > node.second.second) {

node.first = lists[node.second.first][node.second.second];

minHeap.push(node);

}

}

return result;

}

};

int main(int argc, char \*argv[]) {

vector<vector<int>> lists = {{2, 6, 8}, {3, 6, 7}, {1, 3, 4}};

int result = KthSmallestInMSortedArrays::findKthSmallest(lists, 5);

cout << "Kth smallest number is: " << result;

}

