**1.Matrix**

**(1)输入是二维矩阵，输出二维矩阵，矩阵每个点是左上角所有数之和， 要求一次遍历**

**(2)找出横,竖,对角线,的sum 找出最大的一个**

**2.** **把array分成左右两部分，左边的和等于右边的和, 找到这样的一个index**

**先加一遍sum，再过一次，left == sum-left**

**3.** **有一个很多条纪录（人，电影，data），问哪个人看电影最多,然后排序一下选前100。**

**O(Nlog100) Priority\_queue (min)**

**4.** **给个string，找出出现最多的词，有句号需要处理。 Follow top k出现。**

**O(NlogK) priority queue (min)**

priority\_queue<int,vector<int>, greater<int>> minheap

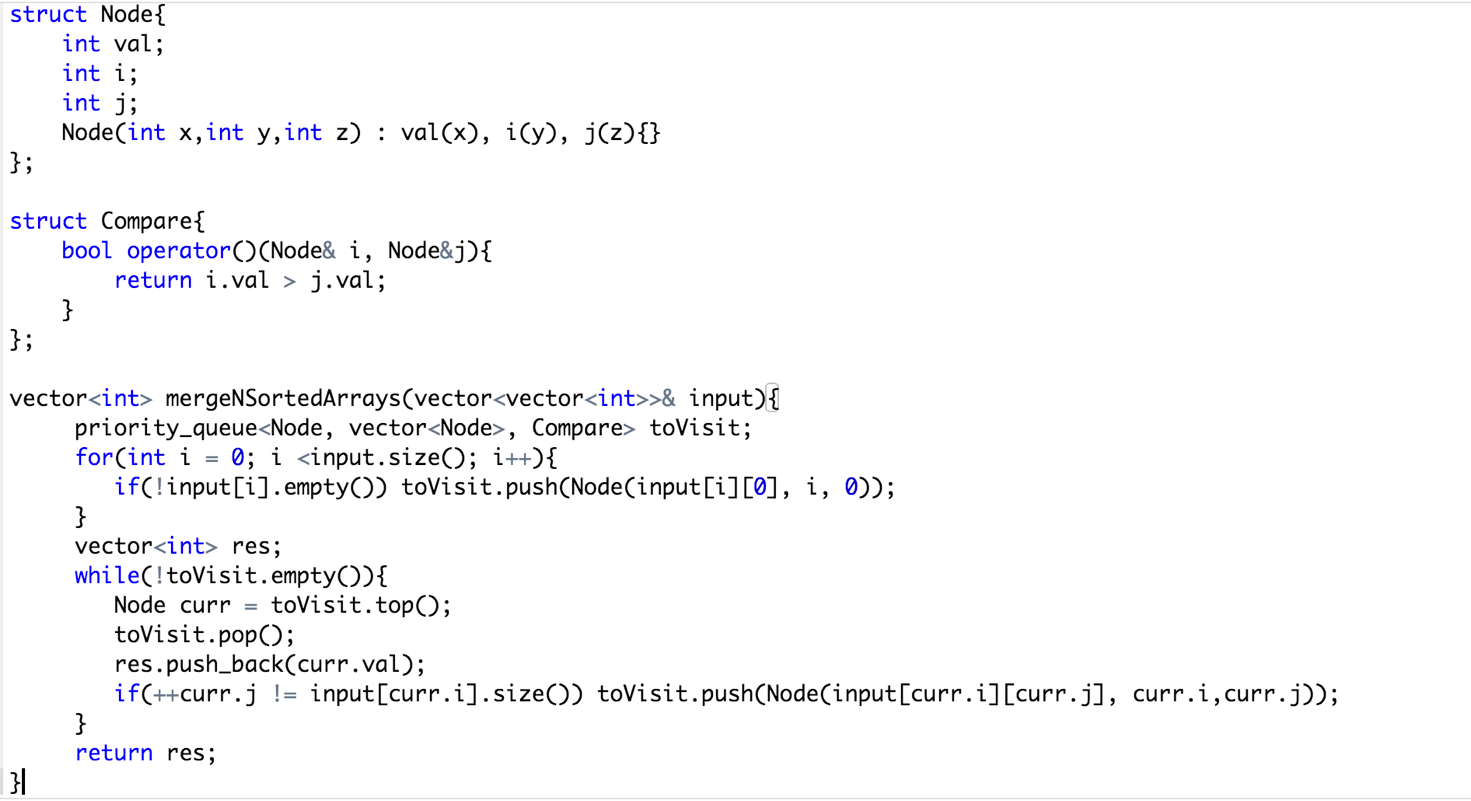
**5.** **Given a list of Points, output k Points closest to (0,0)**

**O(NlogK) Priority\_queue (max)**

**6.** **2nd largest number in array follow up: kth largert number**

**7.Merge k sorted array**

**min heap**

****

**8.** **一个二维数组只包含True和False， 计算所有联通的True的片数。**

**DFS**

**9.** **Write a function to “flood fill” a bitmap, function should allow controlling level of flood fill.**

**A[5][5] of integers where each cell can have only 0/1.**

**If it's 0, its not flodded yet, if it's 1, its already flooded.**

**water at point p at time t1,**

**p, p's neighbors at time t2 (t1+x)**

**p, p's neighbors, p's neighbors' neighbors -> at time t3 (t2+x)**

**current cell (i, j) => i-1, j or i, j+1, or i, j-1, or i+1, j or i+1, j+1 or i-1, j-1**

**input: 2d array(filled), starting cell (i, j), level of flooding (1,2,3....)**

**10.** **小哥给了一个NODE(不是树的，是图)的API，input一个node的SET 返回一个BOOLEAN， 所有的SET里的NODE，是否是连接的图。**

**使用DFS+一个visited table**

**11.Compute number and print all possible paths from (0,0) to (7,7) in a chess board.**

**you can either move down or right.**

**Backtracking : down = 7, right = 7;**

**12.** 简化出来对一组数据找最大最小值，平均值和80%的点，比如[2,3,1,4,5] 中80%的点就是4，

前三个比较简单，最后80%那个 log(n)，变成找bst中第k小

**13.** **现在有两个文件，第一个文件里面是物品的price，第二个文件里面是物品的tax，但他们不对应。比如说，第一个文件是 40 30 20 10， 第二个文件是 3 2 1 4，**

**写一个函数，将文件读进来，然后找出这个州的tax rate。找出两个数组中的最小值，然后相除。**

**14.** **给一个array of int and一个subarray的size**

**要求求出所有sub array的平均值 然后return出现次数最多的平均数的大小和出现次数**

**if(I >= k) sum = sum + array[i] – array[i-k]; // i-k+1 ~ i**

**if(I >= k-1) key = sum/size;**

**15.** 给出一个数组，要求写出一个把这个数组内容随机化的算法

for(int i = 0; i < n; i++)

swap(nums[i], nums[rand()%n]);

**16.** 说麦当劳有6个，9个或者20个包装的chicken nuggests, 让我写code测试一个整数能否正好用这些包装装起来。比如109可以用20个\*5盒+9, 比如22就不行

**bool test(int n){**

**if(n < 0) return false;**

**if(n == 0) return true;**

**if(test(n-20)) return true;**

**if(test(n-9)) return true;**

**if(test(n-6)) return true;**

**return false;**

**}**

**17.** 已知一个ls(dir)函数， 功能等于unix下的ls命令，返回一个string包含dir目录下的所有条目。

让你代码写一个lsR(dir),就是ls()的递归版本，递归的对dir下的所有文件夹和文件做ls操作并显示结果。

**18. Anagram Substring Search**

给两个string, 一个needle, 一个target, target里面找substring, 是needle的任何permutation。

只不可以把needle看做是 a list of character //判断string1是不是string2的subsequence

O(n)解法: moving window, 使用int table[256], 移动一步compare一次，则为O(n) time O(1)space

**19.String to Binary**

string myString = "Hello World";

for (char c : myString)

cout << bitset<8>(c) << endl;