Amazon OA 2020 | Optimize Memory Usage

<https://leetcode.com/discuss/interview-question/823550/amazon-oa-2020-optimize-memory-usage>

Give a computer with total K memory space, and an array of foreground tasks and background tasks the computer needs to do. Write an algorithm to find a pair of tasks from each array to maximize the memory usage. Notice the tasks could be done without origin order.

Input  
The input to the function/method consists of three arguments :  
foregroundTask, an array representing the memory usage of the foreground tasks,  
backgroundTask, an array representing the memory usage of the background tasks,  
K, the total memory space of the computer.

Output  
Return a list of pairs of the task ids.

public static Map<Integer, List<Integer>> fmap;

public static Map<Integer, List<Integer>> bmap;

public static List<List<Integer>> result;

public static List<List<Integer>>optimizeMemoryUsage(int[] ft, int[] bt, int K) {

if (ft == null && bt == null || ft.length == 0 && bt.length == 0) return null;

for (int i = 0; i < ft.length; i++){

fmap.putIfAbsent(ft[i], new ArrayList<Integer>());

fmap.get(ft[i]).add(i);

}

for (int i = 0; i < bt.length; i++){

bmap.putIfAbsent(bt[i], new ArrayList<Integer>());

bmap.get(bt[i]).add(i);

}

Arrays.sort(ft);

Arrays.sort(bt);

int i = 0;

int j = bt.length - 1;

while (i < ft.length || j >= 0 ){

int fi = (i >= ft.length) ? 0 : ft[i];

int bj = (j < 0) ? 0 : bt[j];

if (j < 0 || i >= ft.length || fi + bj == K){

if (j < 0){

if (fi == K){

formList(fi, bj, 1);

}

i++;

}

else if (i >= ft.length){

if (bj == K){

formList(fi, bj, 2);

}

j--;

}

else {

formList(fi, bj, 0);

i++;

j--;

}

} else if (fi + bj > K){

if (fi == K){

formList(fi, bj, 1);

i++;

}

else if (bj == K){

formList(fi, bj, 2);

j--;

} else {

List<Integer> bvalue = bmap.get(bj);

bvalue.remove(0);

if (bvalue.isEmpty()) {

bmap.remove(bj);

}

j--;

}

}

else if (fi + bj < K){

List<Integer> fvalue = fmap.get(fi);

fvalue.remove(0);

if (fvalue.isEmpty()) {

fmap.remove(fi);

}

i++;

}

}

return result;

}

public static void formList(int currenti, int currentj, int flag){

if (flag == 0){

List<Integer> fvalue = fmap.get(currenti);

List<Integer> bvalue = bmap.get(currentj);

result.add(Arrays.asList(fvalue.get(0), bvalue.get(0)));

fvalue.remove(0);

bvalue.remove(0);

if (fvalue.isEmpty()) {

fmap.remove(currenti);

}

if (bvalue.isEmpty()) {

bmap.remove(currentj);

}

}

else if (flag == 1){

List<Integer> fvalue = fmap.get(currenti);

result.add(Arrays.asList(fvalue.get(0), -1));

fvalue.remove(0);

if (fvalue.isEmpty()) {

fmap.remove(currenti);

}

}

else {

List<Integer> bvalue = bmap.get(currentj);

result.add(Arrays.asList(-1, bvalue.get(0)));

bvalue.remove(0);

if (bvalue.isEmpty()) {

bmap.remove(currentj);

}

}

return;

}

public static void main(String[] args) {

int[] ft = {1, 7, 2, 4, 5, 6};

int[] bt = {3, 1, 2};

int K = 6;

fmap = new HashMap<>();

bmap = new HashMap<>();

result = new ArrayList<>();

result = optimizeMemoryUsage(ft, bt, K);

System.out.println(result);

int[] ft1 = {};

int[] bt1 = {3, 6, 6};

int K1 = 6;

fmap = new HashMap<>();

bmap = new HashMap<>();

result = new ArrayList<>();

result = optimizeMemoryUsage(ft1, bt1, K1);

System.out.println(result);

int[] ft2 = {6, 6, 6, 6};

int[] bt2 = {1, 2, 6};

int K2 = 6;

fmap = new HashMap<>();

bmap = new HashMap<>();

result = new ArrayList<>();

result = optimizeMemoryUsage(ft2, bt2, K2);

System.out.println(result);

}