链滴

PAT 甲级刷题实录——1026

作者：aopstudio
原文链接：https://ld246.com/article/1581732832359
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

原题链接

https://pintia.cn/problem-sets/994805342720868352/problems/994805472333250560

思路

这题是真的难，我照自己的做法怎么都解不出，还是在参考了之前做的几个队列模拟题的情况下。无之下只好参考了网上老哥的做法，我觉得这位老哥讲的已经相当好了，我再赘述也没意思，直接把他博文链接放出来：https://blog.csdn.net/richenyunqi/article/details/79562434。我能做的唯一的进就是把他 10005 大小的数组改成 vector。另外我也不是完全复制粘贴下来，而是参考了他的代码后自己写的，其中一开始还是有一个测试点过不去，后来发现是我没考虑一个题目里的细节，即最长球时间不超过两小时，如果超过了按两小时计算。

代码

```
#include <iostream>
#include <algorithm>
#include <string>
#include <vector>
#include <list>
#include <queue>
using namespace std;
struct player
{
    int aTime, pTime;
    bool isVIP=false;
};
struct table
{
    int servedPlayerNum = 0;    //接待的球员数量
    bool isVIP=false,occupy=false;
};
struct playerTable {    //记录球台和球员的动态关系
    int tableNum, playerNum, time;//球台编号、运动员编号、空闲时间
    playerTable(int table, int p, int t) :tableNum(table), playerNum(p), time(t) {}//构造函数
    bool operator <(const playerTable&p)const {//重载<运算符
        return this->time > p.time;
    }
};
int N, K, M;
vector<table> tables;
list<int> wPlayers;
vector<player> players;    //记录所有运动员的下标
priority_queue<playerTable> playerTables;
void output(int t1,int t2);
int searchTable(bool isVIP);
int main()
{
    cin >> N;
    for (int i = 0; i < N; i++)
    {
        int hour, minute, second, pTime, isVIP;
```

```
    scanf("%d:%d:%d %d %d", &hour, &minute, &second, &pTime, &isVIP);
    int aTime = hour * 3600 + minute * 60 + second;
    pTime = min(pTime,120);
    pTime *= 60;
    player newPlayer;
    newPlayer.aTime = aTime;
    newPlayer.pTime = pTime;
    newPlayer.isVIP = isVIP == 1 ? true : false;
    players.push_back(newPlayer);
    playerTables.push(playerTable(0, i, aTime));
}
cin >> K >> M;
tables.resize(K+1);
for (int i = 0; i < M; i++)
{
    int num;
    cin >> num;
    tables[num].isVIP = true;
}
while (playerTables.size()>0)
{
    playerTable p = playerTables.top();
    playerTables.pop();
    if (p.time >= 21 * 3600)
        break;
    if (p.tableNum == 0)    //需要分配球台
    {
        int index = searchTable(players[p.playerNum].isVIP);    //找出空闲球台
        if (index != -1)
        {
            tables[index].occupy = true;    //球台被占用
            int endTime = p.time + players[p.playerNum].pTime;
            playerTables.push(playerTable(index, p.playerNum, endTime));
            output(players[p.playerNum].aTime, p.time);
            tables[index].servedPlayerNum++;
        }
        else //没有空闲球台
            wPlayers.push_back(p.playerNum);
    }
    else //对应的球台可以闲置
    {
        if (wPlayers.size() == 0)    //等待队列为空
            tables[p.tableNum].occupy = false;    //将球台闲置
        else //等待队列不为空
        {
            int temp;    //要分配的球员编号
            if (!tables[p.tableNum].isVIP)    //球台不是VIP球台
            {
                temp = wPlayers.front();
                wPlayers.pop_front();
            }
            else //是VIP球台
            {
                list<int>::iterator it = wPlayers.begin();
```

```
                while (it != wPlayers.end()&&!players[*it].isVIP)    //寻找等待队列中的VIP用户
                    it++;
                if (it == wPlayers.end())    //没有VIP
                {
                    temp = wPlayers.front();
                    wPlayers.pop_front();
                }
                else
                {
                    temp = *it;
                    wPlayers.erase(it);
                }

            }
            int endTime = p.time + players[temp].pTime;
            playerTables.push(playerTable(p.tableNum, temp, endTime));
            output(players[temp].aTime, p.time);
            tables[p.tableNum].servedPlayerNum++;
        }
    }
}
for (int i = 1; i < tables.size(); i++)
{
    if (i == 1)
        cout << tables[i].servedPlayerNum;
    else
        cout << ' ' << tables[i].servedPlayerNum;
}
return 0;
}
void output(int t1, int t2)
{
int hour1 = t1 / 3600, minute1 = t1/ 60%60, second1 = t1 % 60;
int hour2 = t2 / 3600, minute2 = t2/ 60%60, second2 = t2 % 60;
int wait = (t2 - t1 + 30) / 60;
printf("%02d:%02d:%02d %02d:%02d:%02d %d\n", hour1, minute1, second1, hour2, minut
2, second2, wait);
}
int searchTable(bool isVIP)
{
if(isVIP)
    for(int i=1;i<K+1;i++)
        if (!tables[i].occupy&&tables[i].isVIP)
            return i;
for (int i = 1; i < K + 1; i++)
    if (!tables[i].occupy)
        return i;
return -1;
}
```