P2392 考前临时抱佛脚

题目描述

这次期末考试，kkksc03 需要考4科。因此要开始刷习题集，每科都有一个习题集，分别有 s_1,s_2,s_3,s_4 道题目，完成每道题目需要一些时间，可能不等（A_1,A_2,……,A_s1，B_1,B_2,……,B_s2C_1,C_2,……,C_s3，D_1,D_2,……,D_s4。

kkksc03 有一个能力，他的左右两个大脑可以同时计算 2 道不同的题目，但是仅限于同一科。因此，kkksc03 必须一科一科的复习。

由于 kkksc03 还急着去处理洛谷的 bug，因此他希望尽快把事情做完，所以他希望知道能够完成复习的最短时间。

输入格式

本题包含5行数据：

第1行，为四个正整数 s_1,s_2,s_3,s_4

第2行，为 A_1,A_2,……,A_s1共s_1个数，表示第一科习题集每道题目所消耗的时间。

第3行，为 B_1,B_2,……,B_s2共s_2个数。

第4行，为 C_1,C_2,……,C_s3共s_3个数。

第5行，为 D_1,D_2,……,D_s4共s_4个数，意思均同上。

输入样例

```
1 2 1 3
5
4 3
6
2 4 3
```

输出样例

```
20
```

解析

将复杂问题进行抽象化：输入N个数，将这N个数能够分成两组，使得两组总数最接近。

根据样例数据，我们来模拟一下 DFS 的过程。有左脑和右脑可用。每次我们都先用左脑，再用右脑，也就是这样的搜索过程：先全部左脑（方案一）、左1左2（左脑解第一题和第二题）、左1右1、左2右1和右1有2，这么四种组合。

第一个科目有一题，所以我们可以得到：

1、全左脑：耗时 5。

2、全右脑：耗时 5。

因此第一个科目的最小时间为 5 分钟。

第二个科目有两题，所以我们可以得到：

1、全左脑：耗时 4+3=7。

2、左脑解题1，右脑解题2：耗时 max(4,3)=4。

3、左脑解题2，右脑解题1：耗时 max(3,4)=4。

4、全右脑：耗时 4+3=7。

第三个科目有一题，所以我们可以得到：

1、全左脑：耗时 6。

2、全右脑：耗时 6。

因此第三个科目的最小时间为 6 分钟。

第四个科目有三题，所以我们可以得到：

1、全部左脑完成。这样耗时为 2+4+3=9。

2、左脑完成1、2题，右脑完成3题。这样耗时是 max(2+4, 3)=6。

3、左脑完成1、3题，右脑完成2题。这样耗时是 max(2+3, 4)=5。

4、左脑完成1题，右脑完成2、3题。这样耗时是 max(2, 4+3)=7。

5、左脑完成2 3题，右脑完成1题。这样耗时是 max(4+3, 2)=7。

6、左脑完成2题，右脑完成1、3题。这样耗时是 max(4, 3+2)=5。

7、左脑完成3题，右脑完成1、2题。这样耗时是 max(3, 2+4)=6。

8、全部右脑完成。这样耗时为 2+4+3=9。

因此第四个科目的最小时间为 5 分钟。

总时间自然就是 5+4+6+5=20。

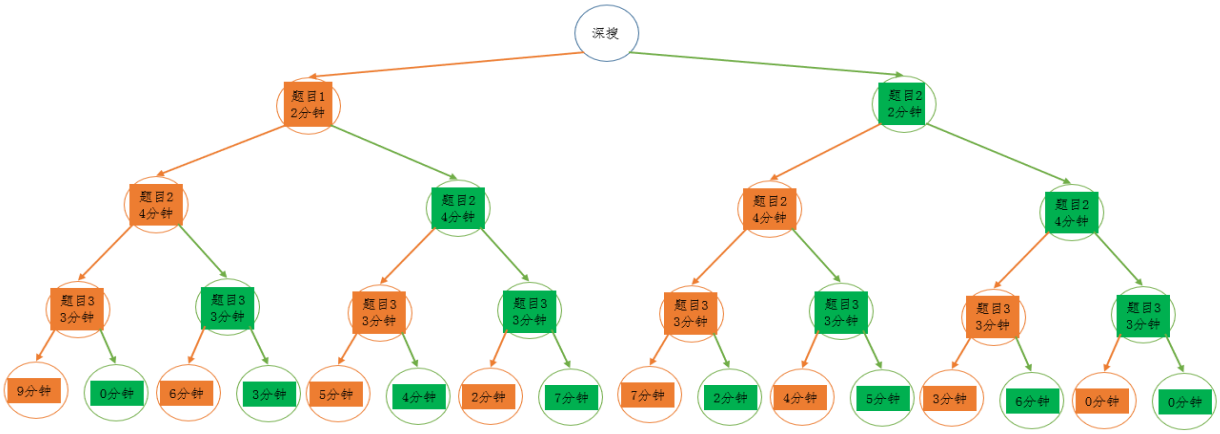
数据规模分析

根据题目描述，我们可以知道，一共 4 个科目，每个科目最多 20 个题目，因此最大的数据集为 4*20=80。也就是说，本题使用 DFS 搜索的时候，可以不需要剪枝。

每个题目耗时不会超过 60 分钟，那么最大的数据可能是 60*20*4=4800。因此用 int 这个数据类型足够。

绘制行为树

假设现在有3道题，耗费时间分别是2分钟，4分钟，3分钟。我们尝试用决策树来秒描述最终的结果。



编码

```
#include <bits/stdc++.h>

using namespace std;

const int MAXN = 4;
const int MAXM = 22;

int subjects[MAXN];//科目
int times[MAXN][MAXM];//时间
int Left;//左脑
int Right;//右脑
int minn;//某一门科目最小时间

//从第i门的第j题目开始复习
void dfs(int i, int j) {
    //判断本次搜索是否结束
    if (j >= subjects[i]) {
        minn = min(minn, max(Left, Right));
        return;
    }

    //左脑工作
    Left += times[i][j];
    dfs(i, j + 1);
    //计算完本侧结果时，需要进行还原，避免影响后续的计算
```

```
    Left -= times[i][j];

    //右脑工作
    Right += times[i][j];
    dfs(i, j + 1);
    Right -= times[i][j];
}

int main() {
    //读入每个科目的题目数
    for (int i = 0; i < 4; i++) {
        cin >> subjects[i];
    }
    //读入每个题的复习时间
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < subjects[i]; j++) {
            cin >> times[i][j];
        }
    }

    //搜索
    int ans = 0;//最后的时间
    for (int i = 0; i < 4; i++) {
        Left = 0;//左脑清零
       Right = 0;//右脑清零
       minn = INT MAX;//由于是要最小值
       dfs(i, 0);//从第i门的第一题开始复习
       ans += minn;
    }

    cout << ans << endl;

    return 0;
}
```