1. 题目描述：p79 3-1
2. 算法设计：

本问题具有最优子结构性质：

设 表示完成当处理完这K个任务后，在A处理机花费时间小于等于i时，B处理器花费的最小时间。

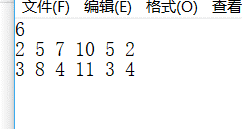
则状态转移方程

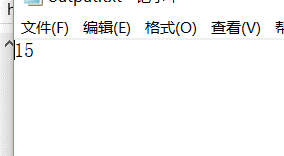
：完成K-1个任务，A处理机花费时间小于等于i-a[k]时，B处理机花费的最小时间。假如将第K个任务交给A处理机来做，B处理机不动，这样完成K个任务时，A处理机花费的时间小于等于 ，而B处理机没动，所以

 ，直接将完成K-1个任务时这个时间复制给完成K个任务的最小时间

：完成K-1个任务，A处理机花费时间小于等于i时，B处理机花费的最小时间。假如将第K个任务交给B处理机来做，A处理机不动，这样完成K个任务时花费的时间，就应该是：完成K-1个任务，A处理机花费时间小于等于i时，B处理机花费的时间，再加上第K个任务在B处理机上花费的时间，即 

1. 运行结果





1. 代码展示

#include "iostream"

#include "stdio.h"

#include "string.h"

#include "fstream"

using namespace std;

int max(int a, int b)

{

return a > b ? a : b;

}

int min(int a, int b)

{

return a < b ? a : b;

}

int missionArrange(int \*a, int \*b, int n)

{

int ta=0, ans, i, k;

int \*T;

for (i = 1; i <= n; i++)

ta += a[i];

T = new int[ta + 1];

for (i = 0; i <= ta; i++)

T[i] = 0;

for (k = 1; k <= n; k++)

for (i = ta; i >= 0; i--)

if (i >= a[k])

T[i] = min(T[i - a[k]], T[i] + b[k]);

else

T[i] = T[i] + b[k];

for (i = 0; i <= ta; i++)

cout << T[i] << " ";

ans = 0X7FFFFFFF;

for (i = 0; i <= ta; i++)

ans = min(ans, max(i, T[i]));

cout << ans;

return ans;

}

int main()

{

int i;

int \*a, \*b, n;

ifstream fin;

ofstream fout;

fin.open("input.txt", ios::in);

fout.open("output.txt", ios::out);

fin >> n;

a = new int[n+1];

b = new int[n+1];

for (i = 1; i <= n; i++)

fin >> a[i];

for (i = 1; i <= n; i++)

fin >> b[i];

fout << missionArrange(a, b, n);

return 0;

}