题意

小明有 n 科作业要做（n ≤ 15）
按名称字母序升序给出n科作业的名称，deadline，耗时
如果某一科作业，比deadline晚几天就扣几分
问他以怎样的顺序写作业，总共扣的分最少
输出
总共扣的分
写作业的顺序（名字）
如果有多种方案，输出名字字典序小的在前面的方案

分析

从n的数据量，考虑状态压缩
考虑转移方式：
　　做完i-1科作业 → 做完 i 科作业
n位2进制数字，第k位为1表示该科作业已做，反之亦然。
刷表方式：
　　枚举每一种状态（总共(1<<n)-1种
　　对于每一种状态，找到他的所有前状态，推出现状态
时间复杂度 $O(n*2^n)$
要输出方案，所以维护一下转移路径，每个dp结点维护 扣的总分score，当前选择科目now，上一个选择科目pre,当前状态耗费时间tim

代码

```
#include<iostream>
#include<cstring>
#include<stack>
#include<algorithm>
#define For(i,a,b) for(int i=(a); i<=(b) ; i++)
#define _For(i,a,b) for(int i=(a); i>=(b) ; i--)
#define Memset(a,b); memset((a),(b),sizeof((a)));
#define Cin(a); scanf("%d",&(a));
#define Cinc(a); scanf(" %c",&(a));
#define Cins(a); scanf("%s",(a));
#define Cout(a,b);  printf("%d",(a));printf(b);
#define Coutc(a,b);  printf("%c",(a));printf(b);
```

```
#define Couts(a,b);  printf("%s",(a));printf(b);
using namespace std;
typedef  long long LL;
typedef  unsigned long long ULL;
typedef  long double LDB;
const int INF = 1<<30;
inline int readint() {int x;cin>>x;return x;}
struct cre{
    char name[100];
    int ddl,cost;
}a[20];
struct cre2{
    int tim,score,pre,now;
}dp[1<<15];
int main()
{
    int _;Cin(_);
    while(_--)
    {
        Memset(dp,0);
        int n;Cin(n);
        for(int i=0;i<n;i++)
        {
            cin>>a[i].name;
            scanf("%d%d",&a[i].ddl,&a[i].cost);
        }
        For(s,1,(1<<n)-1)
        {
            dp[s].score = INF;
            _For(i,n-1,0)
            {
                int temp = 1<<i;
                if(s & temp)
                {
                    int k = s - temp;
                    int st = dp[k].tim + a[i].cost - a[i].ddl;
                    if(st<0) st = 0;
                    if(st + dp[k].score < dp[s].score)
                    {
                        dp[s].score = st + dp[k].score;
                        dp[s].now = i;
                        dp[s].pre = k;
                        dp[s].tim = dp[k].tim+a[i].cost;
                    }
                }
            }
```

```
59          }
60          stack<int>ans;
61          int now = (1<<n)-1;
62          printf("%d\n",dp[now].score);
63          while(now)
64          {
65              ans.push(dp[now].now);
66              now = dp[now].pre;
67          }
68          while(!ans.empty())
69          {
70              cout<<a[ans.top()].name<<endl;
71              ans.pop();
72          }
73      }
74  }
```