7-1　银行排队问题之单队列多窗口服务　(25　分)

假设银行有K个窗口提供服务，窗口前设一条黄线，所有顾客按到达时间在黄线后排成一条长龙。当有窗口空闲时，下一位顾客即去该窗口处理事务。当有多个窗口可选择时，假设顾客总是选择编号最小的窗口。

本题要求输出前来等待服务的N位顾客的平均等待时间、最长等待时间、最后完成时间，并且统计每个窗口服务了多少名顾客。

输入格式:

输入第1行给出正整数N（$\le$1000），为顾客总人数；随后N行，每行给出一位顾客的到达时间T和事务处理时间P，并且假设输入数据已经按到达时间先后排好了顺序；最后一行给出正整数K（$\le$10），为开设的营业窗口数。这里假设每位顾客事务被处理的最长时间为60分钟。

输出格式:

在第一行中输出平均等待时间（输出到小数点后1位）、最长等待时间、最后完成时间，之间用1个空格分隔，行末不能有多余空格。

在第二行中按编号递增顺序输出每个窗口服务了多少名顾客，数字之间用1个空格分隔，行末不能有多余空格。

输入样例：

```
9
0 20
1 15
1 61
2 10
10 5
10 3
30 18
31 25
31 2
3
```

输出样例：

```
6.2 17 61
5 3 1
```

```
#include<stdio.h>

typedef struct{
    int arrive;
    int spend;
}Peo;

Peo queue[1000];
```

```
int input(){
    int num;
    int windom;
    int arrive,spend;
    int waittime,maxwaittime,lasttime;
    waittime=maxwaittime=lasttime=0;
    int rear,front;
    front=rear=0;
    scanf("%d",&num);
    for(int i=0;i<num;i++){
        scanf("%d %d",&arrive,&spend);

        queue[front].arrive=arrive;
        queue[front++].spend=spend;
    }
    scanf("%d",&windom);
    int isfull=0;
    int windoms[3];
    windoms[0]=0;
    windoms[1]=0;
    windoms[2]=0;
    int count[3];
    count[0]=count[1]=count[2]=0;

    while(front!=rear){
        int flag=1;
        Peo peo=queue[rear++];
        for(int i=0;i<windom;i++){
            if(peo.arrive>=windoms[i]){
                    if(peo.spend<=60){
                      windoms[i]=peo.arrive+peo.spend;
                    }else{
                      windoms[i]=peo.arrive+60;
                    }
                count[i]++;
                  flag=0;
                  break;
            }
        }
    if(flag==1){
            int min=1000;
            int choose;
        for(int i=0;i<windom;i++){
            if(windoms[i]<min){
                choose=i;
                min=windoms[i];
            }
```

```
            }
            printf("%d\n",peo.arrive);
            waittime=waittime+windoms[choose]-peo.arrive;
            if(windoms[choose]-peo.arrive>maxwaittime){
                maxwaittime=windoms[choose]-peo.arrive;
            }
            if(peo.spend<=60){
            windoms[choose]+=peo.spend;
            }else{
            windoms[choose]+=60;
            }
            count[choose]++;
        }

        }

        for(int i=0;i<windom;i++){
            if(windoms[i]>lasttime){
                lasttime=windoms[i];
            }
        }
        printf("%.1f %d %d\n",1.0*waittime/num,maxwaittime,lasttime);
        printf("%d %d %d",count[0],count[1],count[2]);


}


int main(){
    input();
}
```