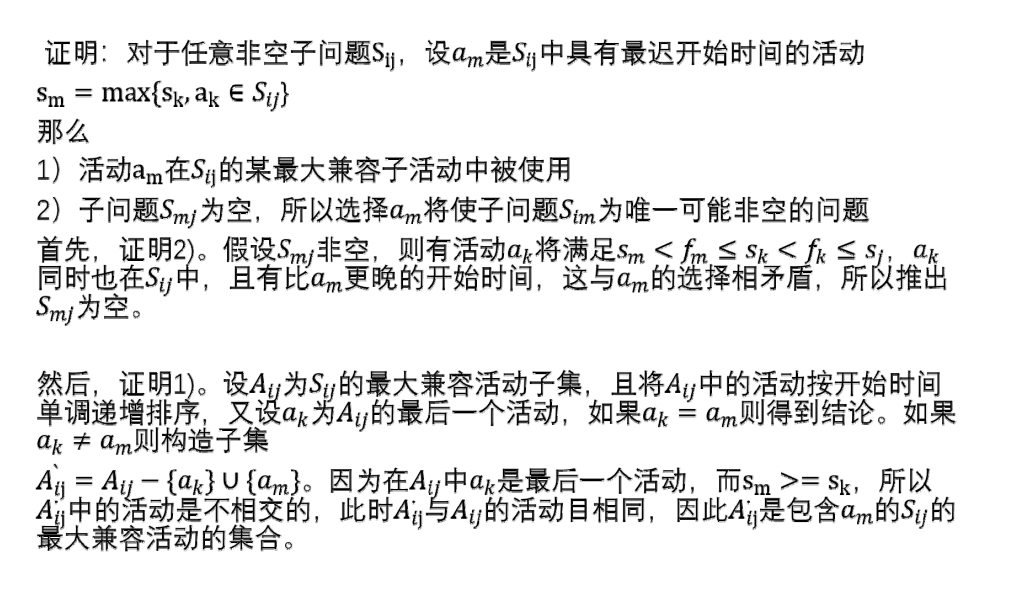
**作业请严格按照要求提交，否则将酌情扣分！**

1. **（30分）**假定我们不再一直选择最早结束的活动，而是选择最晚开始的活动，前提仍然时与之前选出的所有活动兼容。描述如何利用这一方法设计贪心算法，并证明算法会产生最优解。



**2、（30分）**考虑用最少的硬币找n美分零钱的问题。假定每种硬币的面额都是整数。

a.设计贪心算法求解找零问题。假定有25美分、10美分、5美分和1美分4种面额的硬币。

b.设计一组硬币面额，使得贪心算法不能保证的到最优解。这组硬币面额中应该包含1美分，使得对每个零钱值都存在找零方案。

答案：a. 总是选择面额最大的硬币，不断重复这一过程直到剩余要找零钱为0. b. 硬币面额为{1, 3, 4}，找零为6，按照贪心算法结果为{1, 1, 4}但最优解为{3, 3}

**3、（40分）编程题：**柠檬水找零

题目描述：

在柠檬水摊上，每一杯柠檬水的售价为 5 美元。

顾客排队购买你的产品，（按账单 bills 支付的顺序）一次购买一杯。

每位顾客只买一杯柠檬水，然后向你付 5 美元、10 美元或 20 美元。你必须给每个顾客正确找零，也就是说净交易是每位顾客向你支付 5 美元。

注意，一开始你手头没有任何零钱。

如果你能给每位顾客正确找零，返回 true ，否则返回 false 。

提示：

0 <= bills.length <= 10000

bills[i] 不是 5 就是 10 或是 20

示例 1：

输入：[5,5,5,10,20]

输出：true

解释：

前 3 位顾客那里，我们按顺序收取 3 张 5 美元的钞票。

第 4 位顾客那里，我们收取一张 10 美元的钞票，并返还 5 美元。

第 5 位顾客那里，我们找还一张 10 美元的钞票和一张 5 美元的钞票。

由于所有客户都得到了正确的找零，所以我们输出 true。

示例 2：

输入：[5,5,10]

输出：true

示例 3：

输入：[10,10]

输出：false

示例 4：

输入：[5,5,10,10,20]

输出：false

解释：

前 2 位顾客那里，我们按顺序收取 2 张 5 美元的钞票。

对于接下来的 2 位顾客，我们收取一张 10 美元的钞票，然后返还 5 美元。

对于最后一位顾客，我们无法退回 15 美元，因为我们现在只有两张 10 美元的钞票。

由于不是每位顾客都得到了正确的找零，所以答案是 false。

**要求：运用贪心思想作答，请写出分析过程，并用一种语言（最好是C++或JAVA）实现你的思路，上交作业时请将代码一并提交，代码粘贴在交作业的word里面，复杂度尽可能低。**

参考答案：

bool lemonadeChange(int\* bills, int billsSize){

int five = 0;

int ten = 0;

int tewnty = 0;

int i;

for (i = 0; i < billsSize; i++) {

if (bills[i] == 5) {

five++;

} else if (bills[i] == 10) {

ten++;

five--;

} else {

tewnty++;

if (ten == 0) {

five--;

five--;

} else {

ten--;

}

five--;

}

if (five < 0 || ten < 0) {

return false;

}

}

return true;

}