14 难题专场

笔记本： DP Note
创建时间： 11/4/2019 12:47 AM **更新时间：** 11/4/2019 12:59 AM
作者： tanziqi1756@outlook.com
URL： https://www.jiuzhang.com/course/36/dialog/#chapter-117

必做

单选题 **下面二分查找算法(Binary search)的代码是否正确？（查找数组中第一个出现target）**

```
while (start < end) {
    int mid = start + (end - start) / 2;
    if (nums[mid] == target) {
        end = mid;
    } else if (nums[mid] < target) {
        start = mid;
    } else {
        end = mid;
    }
}
```

A 正确　　B 错误

提交　　我不会

正确答案是 B，有61%的同学答对了，加油赶上他们！

解析：这个代码是错误的，我们可以举一个例子看看，比如当start=3，end=4的时候，这时候mid=3，如果nums[mid]<target的话就会一直start=mid=3，陷入死循环。

必做

多选题 **上面代码如果我们要修改，以下那些修改的方法是正确的？**

A 第一行改为 `while (start + 1 < end){`

B 第二行改为 `int mid = start + (end - start) / 2 + 1;`

C 第六行改为 `start = mid + 1`

D 第七行改为 `end = mid - 1`

提交　　我不会

正确答案是 A C ，有16%的同学超过了你，但是千万不要气馁。

解析：首先我们还是代入刚刚说的反例，我们就可以发现BD的修改方法还是陷入了死循环，A选择我们在start和end还差1的时候结束了循环，那么最后我们就需要比较一下start和end哪一个才是我们要查找的位置。而C的选项不需要，结束的时候end就是我们要查找的位置

必做

多选题 **现在我们有一个序列a，长度为n，m个i，j的值，每一个i，j查询一次a[i]+a[i+1]+...+a[j-1]+a[j]的值(0<=i<=j<=n)。下列哪些方法的时间复杂度最小？（如果多个方法时间复杂度一样请都选上）**

A 循环查询(Traversing)，每一次都从i遍历到j

B 构建树状数组(Binary Indexed Tree)查询

C 构造线段树(Segment tree)查询

D 求得a的前缀和(Prefix sum)查询

提交 我不会

正确答案是 D ，有22%的同学做对了这道题目哦，继续努力！

解析：我们来每一种方法都分析一下时间复杂度。
A.m次i到j的遍历，那么时间复杂度O(m*(j-i))，最坏的情况就是j-i=n，那么最坏情况的时间复杂度为O(m*n).
B.构造树状数组的时间复杂度为O(nlogn)，每一次查询的时间复杂度为O(logn)，那么总的时间复杂度就是O(nlogn+mlogn).
C.线段树的时间复杂度和树状数组相同,构造为O(nlogn)，查询的时间复杂度为O(logn)，那么总的时间复杂度就是O(nlogn+mlogn).
D.求一遍前缀和sum的时间复杂度是O(n)，查询结果时只需要sum[j]-sum[i-1]即可，时间复杂度是O(1)，所以总的时间复杂度就是O(n+m)

必做

单选题 **有20个的信封，每个信封都有一个整数对 (w, h) 分别代表信封宽度和高度。20个信封的 (w, h) 如下：** [[15,8],[2,20],[2,14],[4,17],[8,19],[8,9],[5,7],[11,19],[8,11],[13,11],[2,13],[11,19],[8,11],[13,11],[2,13],[11,19],[16,1],[18,13],[14,17],[18,19]] **一个信封的宽高均大于另一个信封时可以放下另一个信封。 求最大的信封嵌套层数。**

A 4

B 5

C 6

D 7

提交 我不会

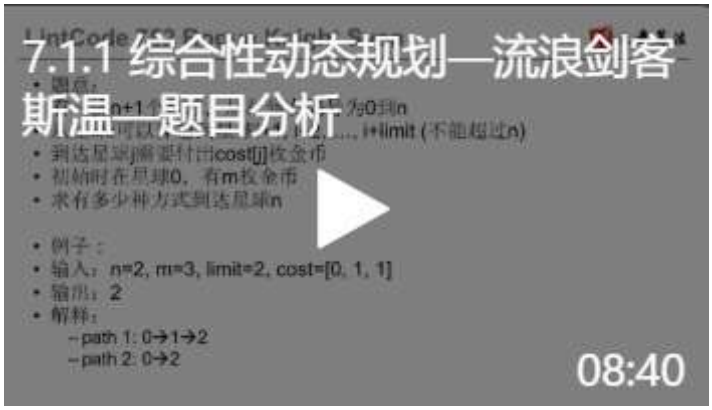
正确答案是 B ，有27%的同学答对了，要加油了。

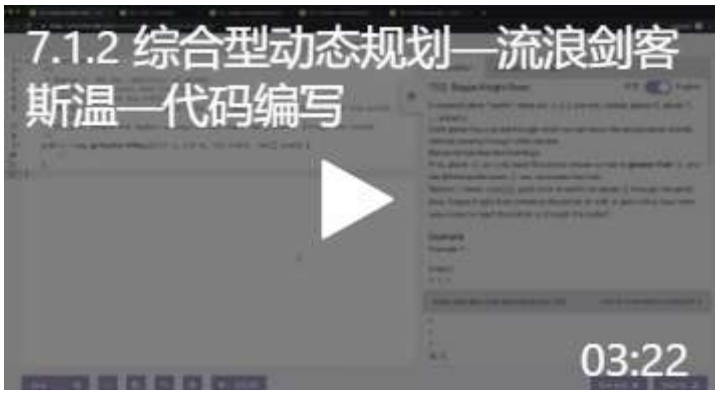
解析：我们选择的信封为[5,7]->[8,9]->[13,11]->[14,17]->[18,19]。

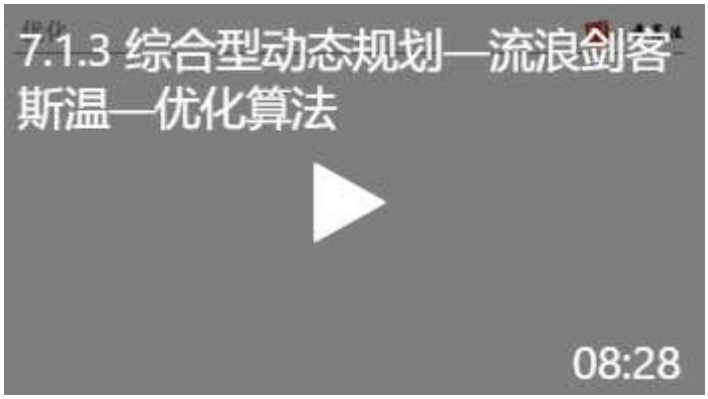
正确答案是 B ，有27%的同学答对了，要加油了。

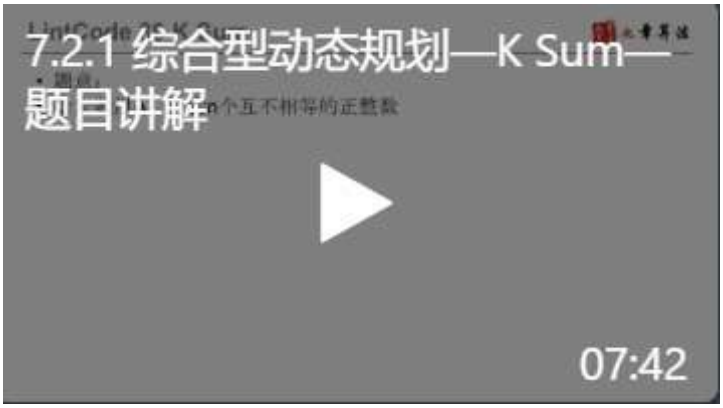
解析：我们选择的信封为[5,7]->[8,9]->[13,11]->[14,17]->[18,19]。

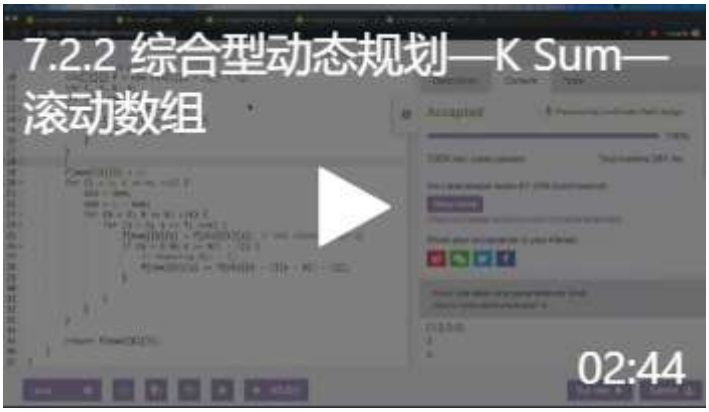
我们出去旅行时抵达一个城市一定需要交通费，在我们的旅行费用有限的情况下，有多少种方式让我们能旅行到不同的城市呢？接下来让我们来看看流浪剑客斯温是如何进行星际传送的吧！

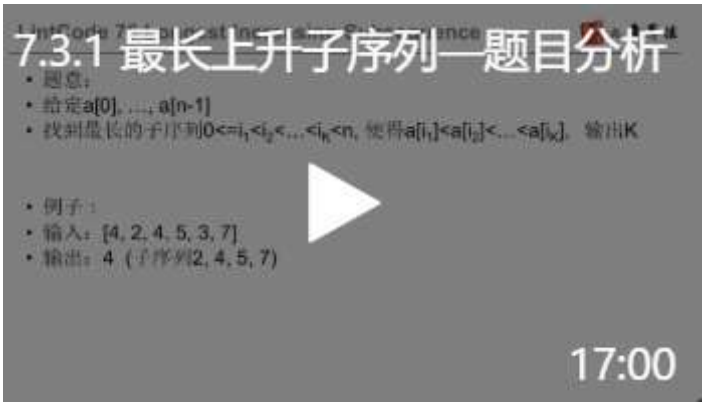


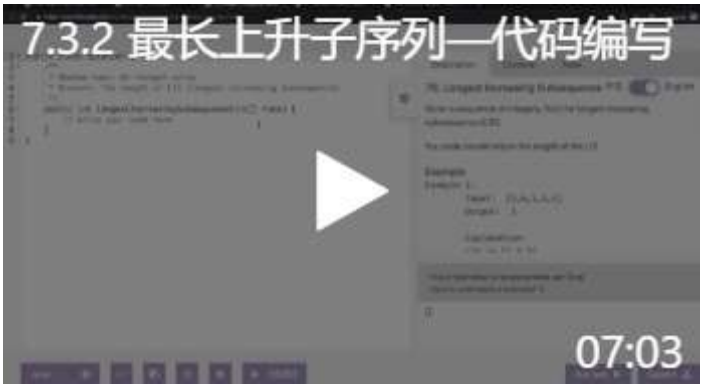












必做

单选题 **还记得课前预习的信封那题吗？它是二维的，如果我们对其排序后能不能转化成为1维最长上升子序列？**

A 能

B 不能

提交

我不会

正确答案是 A ，有55%的同学答对了，加油赶上他们!

解析：这是可以的哦~我们只需要对其中一维进行一个排序，这样另外一维就可以转换成为一个最长上升子序列，我们就实现了一个二维的题目降成一维来完成。

必做

多选题 **那么我们应该怎么样对其排序呢?**

A 先根据w从小到大，如果w相等根据h从小到大。

B 先根据w从小到大，如果w相等根据h从大到小。

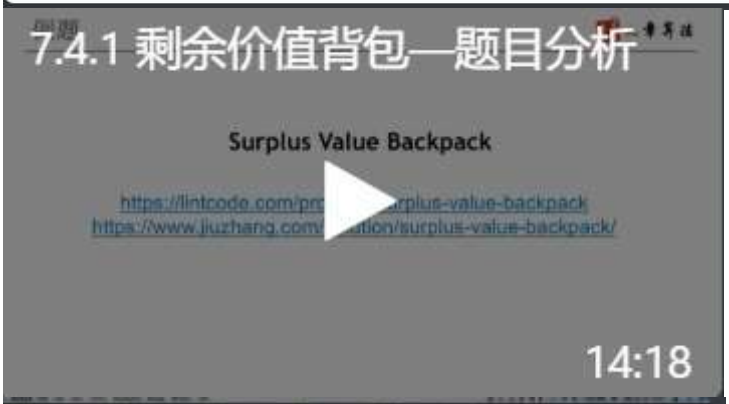
C 先根据w从大到小，如果w相等根据h从小到大。

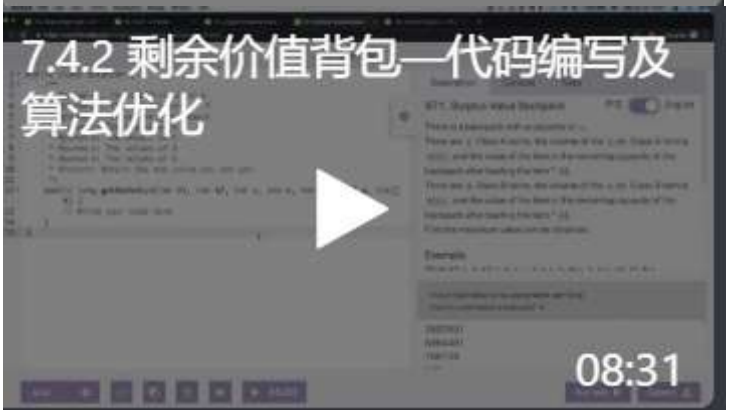
D 先根据w从大到小，如果w相等根据h从大到小。

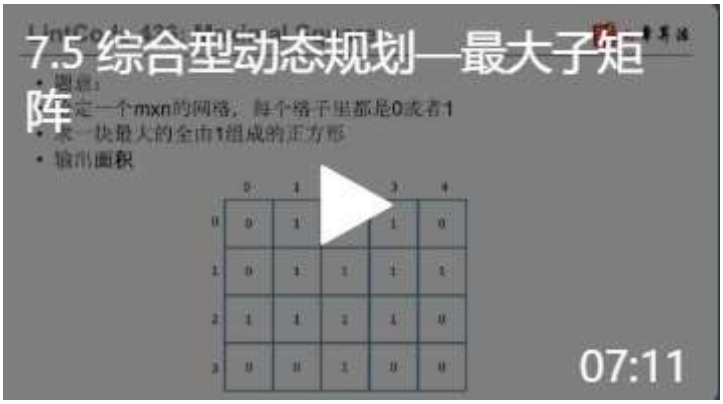
提交　　我不会

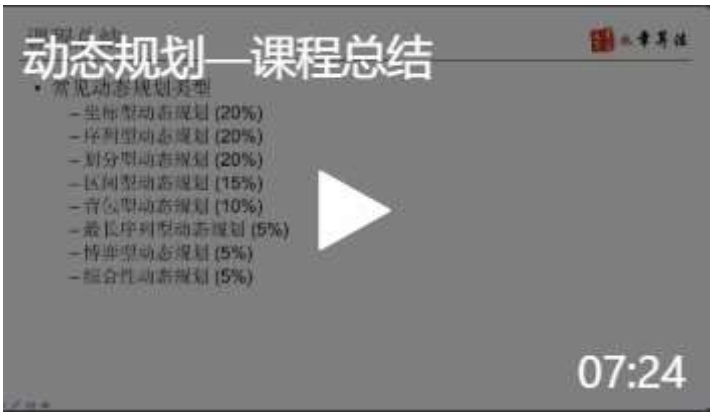
正确答案是 B C ，有6%的同学超过了你，但是千万不要气馁。

解析：首先对w从小到大，然后因为要w、h都大于前一个信封的w、h才行，所以h应该是从大到小，这样我们只需要对h求一次最长上升子序列就可以得到答案了。同理w，h排序对调一下也行。









必做

单选题 **使用以下映射方式将 A-Z 的消息编码为数字:**

'A' -> 1
'B' -> 2
...
'Z' -> 26
除此之外, 编码的字符串也可以包含字符 *, 它代表了 1 到 9 的数字中的其中一个.给出包含数字和字符 * 的编码消息, 返回所有解码方式的数量, 因为结果可能很大, 所以结果需要对 10^9 + 7 取模。如果编码是"**", 那么有多少种解码的方式

A 27

B 78

C 96

D 676

提交 我不会

正确答案是 C ，有35%的同学做对了这道题目哦，继续努力！

解析：如果每个*号表示1个字符，那么**就有99=81种可能性，如果**号表示一个字符，那么就只有11-26（20不算）这里的15个数字，所以答案是96种可能性

必做

单选题 **如果我们只考虑一个字符的话，以下程序的返回值x1,x2,x3,x4依次一个填多少?**

```
private int cnt1(char c) {
    if (c == '0') {
        return x1;
    }
    if (c == '*') {
        return x2;
    }
    if (c == '1' || c == '2') {
        return x3;
    }
    return x4;
}
```

A 1911

B 0911

C 01011

D 11011

E 01090

F 0991

提交　　我不会

正确答案是 B ，有35%的同学答对了，要加油了。

解析：只考虑一个字符时，数字0应该无法解码成字母，而1-9应该是等同的都是转化成为1个字母，*号是1-9的任意一个就有9种可能性。

必做

单选题 **如果我们只考虑一个字符的话，以下程序的返回值x1,x2,x3依次一个填多少？（c1表示前面的字符，c2表示后面的）**

```
private int cnt2(char c1, char c2) {
        if (c1 == '2') {
                if (c2 == '*') {
                        return x1;
                }
                if (c2 <= '6') { ... }
                return x2;
        }
        if (c1 >= '3' && c1 <= '9') {
                return x3;
        }
        ...
}
```

A 901

B 611

C 601

D 600

E 911

F 900

提交　　我不会

正确答案是 D，有14%的同学答对了，加油赶上他们!

解析：2*只考虑两个数字解码成字母只有21,22,23,24,25,26，27-99都不能转换成为字母

必做

单选题 **那么这题的转移方程应该是怎样的?**

A

```
f[i] = f[i - 1] * cnt
1(s[i - 1]);
f[i] += f[i - 2] * cn
t2(s[i - 2], s[i -
1]);
```

B

```
f[i] = f[i - 1] + cnt
1(s[i - 1]);
f[i] += f[i - 2] + cn
t2(s[i - 2], s[i -
1]);
```

C

```
f[i] = f[i - 1] * cnt
1(s[i - 1]);
f[i] *= f[i - 2] * cn
t2(s[i - 2], s[i -
1]);
```

D

```
f[i] = f[i - 1] + cn
t1(s[i - 1]);
f[i] *= f[i - 2] + c
nt2(s[i - 2], s[i -
1]);
```

提交　　我不会

正确答案是 A，有21%的同学超过了你，但是千万不要气馁。

解析：不同位置应该是相乘，比如第一个字母有9种可能性、第二个字母有6中可能性，那么总的可能性就是6*9=54种，相同位置应该相加，这个位置单独一个位置只考虑一个字符有9，考虑两个字符有6种，那么总的可能性就是9+6=15种，所以说cnt1和cnt2的情况要相加，而f[i-1]和cnt1、cn2就应该是相乘的。