**1. 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，前提仍然是与之前选出的所有活动均兼容。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。**

**算法描述**

假设所有活动已经按**开始**时间**逆序**排序

|  |
| --- |
| GREEDY-ACTIVITY-INVERSE-SELECTOR(s, f)  n=s. length  A={a1}  k=1  for m=2 to n  if f[m] ≤ s[k]  A=A∪{am}  k=m  return A |

证明：

记活动集S={a1,a2,a3……an}，ai={si, fi}，假设活动开始时间最晚的是am，那am必然在其中一个最大兼容解中。因为如果存在最优解S’={ai1,ai2,ai3……aik}，am不在S’中，S’中必然存在一个开始时间最晚的活动ak，用am替换ak，S’仍然兼容。

从另一个角度来想，记活动集S’={a’1,a’2,a’3……a’n}，其中a’i={fi , si}，子集B’={a’i1,a’i2,a’i3……a’ik}S’兼容当且仅当B={ai1,ai2,ai3……aik}S兼容，因此S’的最优解与S的最优解一一映射。在S’中使用按结束最早时间排序的GREEDY-ACTIVITY –SELECTOR算法求最优解，得到的结果映射到S中就是按开始最晚时间排序的结果。

**2. 设计算法，在O(n)时间内求解分数背包问题。**

记第i件商品的价值为Vi，重量为Wi，背包的承重为W。

1. 找到商品单位价值v/m的中位数m
2. 将所有物品分为三类 计算三类的总重量，，
3. 讨论以下情况：
   1. ，不放任何物品，从(1)开始对的物品，重量分类
   2. 否则将中所有物品放入背包中
      1. 如果，从中拿出的物品把背包放满
      2. 把的物品都放入背包，从(1)开始对的物品，重量分类

寻找中位数的线性查找算法伪代码为：

|  |
| --- |
| MIDDLE\_PARTITION (V, W, start, end)  i = start  for j=i+1 to end  if V[j]/W[j] ≥ V[start]/W[start] //左边放大的，右边放小的  i = i + 1  EXCHANGE (V, W, i, j)  EXCHANGE (V, W, i, start)  len = end-start+1  if i < len/2  return MIDDLE\_PARTITION(V, W, i+1, end)  else if i > len/2  return MIDDLE\_PARTITION(V, W, start, i-1)  else  return V[i]/W[i] |

求解分数背包问题的伪代码为：

|  |
| --- |
| PACKAGE\_SELECT (V, W, start, end, weight)  if start == end  return V[start]  middle = MIDDLE\_PARTITION (V, W, start, end, V.length)  w=0, v=0  for i = start to middle  w = w + W[i]  v = v + V[i]  if w == weight  return v  else if w < weight && w+W[middle] ≥ weight  return v + V[middle]/W[middle] \* (weight-w)  else if w < weight  return PACKAGE\_SELECT(V,W,middle+1,end,weight-w-W[middle])  else  return PACKAGE\_SELECT(V,W,start,middle-1,weight) |

**3. 推广霍夫曼算法，使之能生成三进制的码字，并证明你的算法能生成最优三进制码。**

算法伪码：

|  |
| --- |
| TRI-HUFFMAN(C)  n=|C|  Q=C  for i=1 to n-1  allocate a new node k  k.first = x = EXTRACT-MIN(Q)  k.second = y = EXTRACT-MIN(Q)  k.third = z = EXTRACT-MIN(Q)  INSERT(Q, k)  return EXTRACT-MIN(Q) |

证明：

(1) 首先证明，令x, y, z是C中频率最低的三个字符，那么存在C的一个最优前缀码，x和y和z的码字长度相同，且只有最后一个进制位不同。

令T表示任意一个最优前缀码对应的编码树，a，b，c是T中深度最大的兄弟叶节点。a.freq≤b.freq≤c.freq，x.freq≤y.freq≤z.freq，因为x,y,z为C中频率最低的三个字符，所以有x.freq≤a.freq，y.freq≤b.freq，z.freq≤c.freq。在T中交换x和a生成一棵新树T’，在T’中交换b和y生成一棵新树T’’，在T’’’中交换c和z生成一棵新树T’’’。T和T’的代价差为：

同理，，，因此T’’’也是最优树，且x,y,z是其中最深的兄弟节点。

(2)接着证明，令C’为C去掉x和y和z，加入一个新字符k后得到的字母表，k.freq=x.freq+y.freq+z.freq，T’为字母表C’的任意一个最优前缀码对应的编码树，我们可以将T’中叶结点k替换为以x,y,z为孩子的内部结点，得到树T，而T就是表示字母表C的一个最优前缀码。

由于，可以得到：

因此可以得到结论

假定T对应的前缀码不是C的最优前缀码。存在最优编码树T’’满足B(T’’)<B(T)，T’’包含兄弟结点x,y,z，令T’’’为T’’中将x,y,z以及他们的父结点替换为k得到的树，可以得知

这与T’对应C’的一个最优编码树相矛盾，因此T是C的一个最优前缀码。

由(1)和(2)可知，TRI-HUFFMAN会生成一个最优前缀码。