[12]贪心

深入浅出程序设计竞赛
第 2 部分 – 初涉算法
V 2021-02

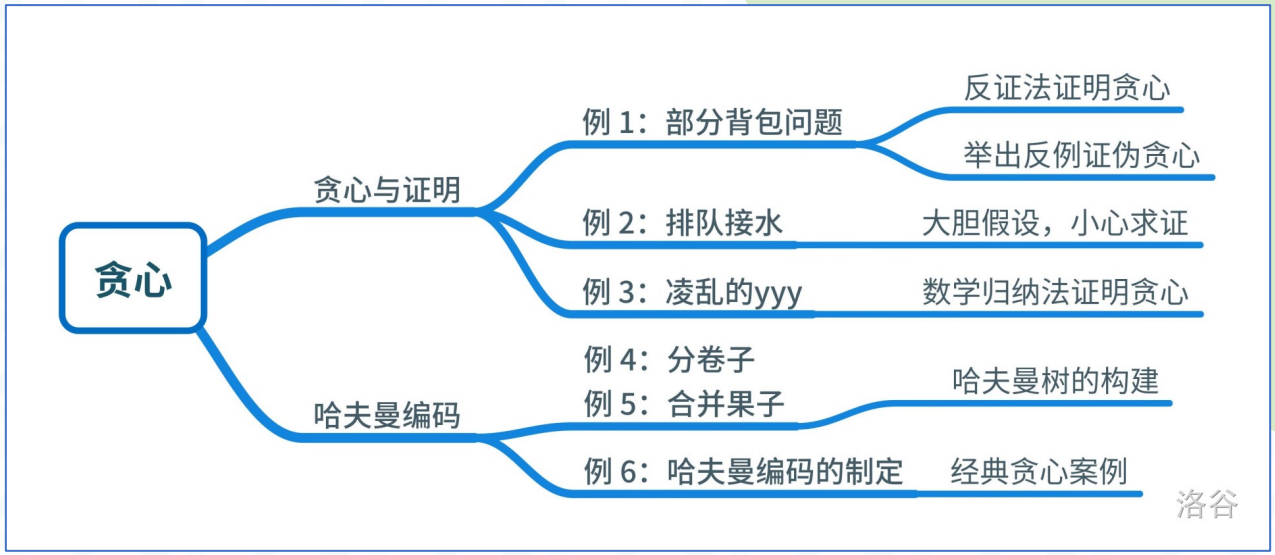
洛谷
www.luogu.com.cn

洛谷

版权声明

- 其它《深基》配套资源、购买本书等请参阅：https://www.luogu.com.cn/blog/kkksc03/IPC-resources
- 如果课件有任何错误，请在这里反馈 https://www.luogu.com.cn/discuss/show/296741

本章知识导图


贪心
贪心与证明
例 1：部分背包问题
反证法证明贪心
举出反例证伪贪心
例 2：排队接水
大胆假设，小心求证
例 3：凌乱的yyy
数学归纳法证明贪心
哈夫曼编码
例 4：分卷子
例 5：合并果子
哈夫曼树的构建
例 6：哈夫曼编码的制定
经典贪心案例
洛谷

第 12 章 贪心

贪心与证明

在算法竞赛中求解某些问题时，只需要做出在当前看来是最好的选择就能获得最好的结果，而不需要考虑整体上的最优。

请翻至课本 P164

如何获得好成绩？

如果想在算法竞赛中得奖，就要尽可能多读书、多思考、多练习。

一般来说，学习越努力，成绩就越好。

每天 洛谷网校 0 小时 刷题 0 小时 收益：0	每天 洛谷网校 2 小时 刷题 3 小时 收益：5	每天 洛谷网校 10 小时 刷题 12 小时 收益：?

但因为花了太多时间在编程上而极度压缩休息的时间，反而会效率低下，得不偿失。

不过，在很多场合，确实是越多越好的。

部分背包问题

例 12.1（洛谷 P2240）

有 N堆金币，第 i 堆金币的总重量和总价值分别是 m_i, v_i。

阿里巴巴有一个承重量为 $T(T \leq 1000)$ 的背包，但无法将全部的金币都装进去。他想装走尽可能多价值的金币。

第1堆	第2堆	第3堆	第4堆
m1=10; v1=60	m2=20; v2=100	m3=30; v3=120	m4=15; v4=45
v1/m1=6	v2/m2=5	v3/m3=4	v4/m4=3

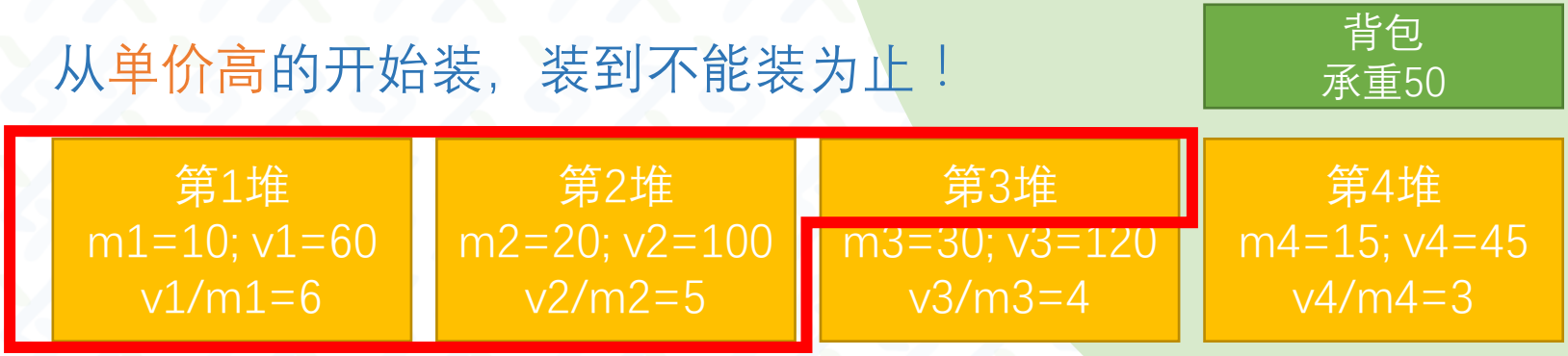
所有金币都可以随意分割，分割完的金币重量价值比（也就是单位价格）不变。

请问阿里巴巴最多可以拿走多少价值的金币？

背包
承重50

部分背包问题

从单价高的开始装，装到不能装为止！



证明：

1. 所有的东西价值都是正的，因此只要金币总数足够，背包就必须要装满而不能留空；

2. （反证法）假设没在背包中放入单价高的金币，而放入了单价低的金币，那么可用等重量的高价值金币替换掉背包里的低价值金币，总价值更高了。

部分背包问题

为了方便排序，定义了coin结构体来存储金币堆的重量和价值——性价比不需要存下来，而是在调用sort的时候进行判断。

```
#include <cstdio>
#include <algorithm>
using namespace std;
struct coin {
  int m, v; // 金币堆的重量和价值
} a[110];
bool cmp(coin x, coin y) {
  return x.v*y.m > y.v*x.m; //判断单价
}
int main() {
  int n, t, c, i; float ans = 0;
  scanf("%d%d", &n, &t);

      // 见右边的贪心过程

  printf("%.2lf", ans);
  return 0;
}
```

```
c = t; // 背包的剩余容量
for (i = 0; i < n; i++)
    scanf("%d%d", &a[i].m, &a[i].v);
sort(a, a + n, cmp); // 对单价排序
for (i = 0; i < n; i++) {
    if (a[i].m > c) break;
    //如果不能完整装下就跳出
    c -= a[i].m;
    ans += a[i].v;
}
if (i < n)
    // 剩余空间装下部分金币
    ans += 1.0 * c / a[i].m * a[i].v;
```

比较性价比时本应判断x.v/x.m>y.v/y.m，为什么要写成这样呢

部分背包问题

如果藏宝洞里面不是一堆堆金币，而是一个个单价不一且无法分割的金块，还能使用类似的策略吗？

从单价高的开始装，装到不能装为止？

背包
承重50

第1块
m1=10; v1=60
v1/m1=6

第2块
m2=20; v2=100
v2/m2=5

第3块
m3=30; v3=120
v3/m3=4

第4块
m4=15; v4=45
v4/m4=3

	重量	价值	单价
①	10	60	6
②	20	100	5
③	20	80	4

(a) 总价:240

	重量	价值	单价
①	10	60	6
②	20	100	5
④	15	45	3

(b) 总价:205

	重量	价值	单价
②	20	100	5
③	30	120	4

(c) 总价:220

(a) 是可分割金币的装包方案，通过贪心策略使利益最大化。

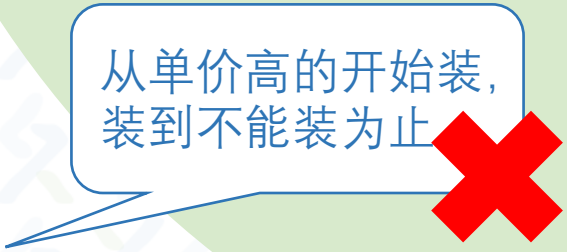
(b) 然而使用同样的办法装包，非最优解

(c) 战略性放弃性价比最高的金块可能会让你获得更多。

部分背包问题

仅仅举出了一个反例就推翻了一个错误的贪心算法，可见使用贪心策略时要特别注意正确性。

贪心需要证明正确性！



本题的正确做法是搜索或者动态规划

会在对应的章节介绍类似的题目。

排队接水

例 12.2（洛谷 P1223）

有 n 个人在一个水龙头前排队接水，每个人接水的时间为 T_i。

找出这 n 个人排队的一种顺序，使 n 个人的平均等待时间最小。

T_i 各不相同，不大于10^6。

```
10
56 12 1 99 1000 234 33 55 99
812
```

```
3 2 7 8 1 4 9 6 10 5
291.90
```

提示：求最短平均时间就是求所有人的最短等待时间和。

排队接水

由于只允许最多一个人同时打水，所以某人等待时间总和就是前面每个单人时间的和。

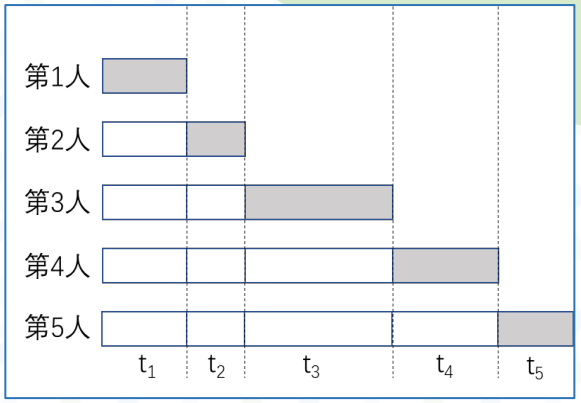
第一个人不需要等待，第二个人需要等待一个人的时间，第三个人要等待前两人。假设经安排，第 i 个同学的打水时间是 t_i。

同学等待时间总和 $s = (n-1)t_1 + (n-2)t_2 \cdots + 1 \cdot t_{n-1} + 0 \cdot t_n$

可发现，t_1的系数较大，t_n系数较小。

猜测：t_1到t_n应该从小到大排序，

可以使时间总和 s 最小。



排队接水

需要证明！！！！（反证法）

假设最佳方案中，t_1到t_n不是小到大排，当 $i < j$ 时，$t_i > t_j$。

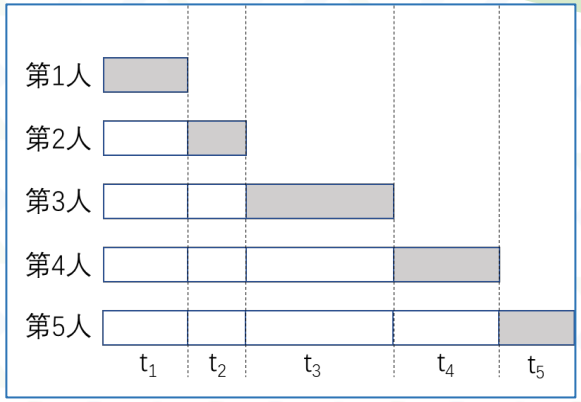
这两项贡献的总时间是$S_1 = a \cdot t_i + b \cdot t_j$，其中系数$a > b$。

若将t_i和t_j调换，那么贡献总时间变为$S_2 = a \cdot t_j + b \cdot t_i$，两者相减

$S_1 - S_2 = a(t_i - t_j) - b(t_i - t_j) = (a - b)(t_i - t_j) > 0$。

调换后总时间会缩短，和原来认为是“最佳方案”矛盾。

所以贪心算法成立。



数楼梯

使用结构体来存储每位同学的信息

按照接水时间从小到大排序，时间相同时编号小的同学优先。

```
#include <cstdio>
#include <algorithm>
using namespace std;
struct water {
  int num, time;
} p[1010];
bool cmp(water a, water b) {
  if (a.time != b.time)
    return a.time < b.time;
  return a.num < b.num;
}
int n, sum = 0;
```

```
int main() {
  scanf("%d", &n);
  for (int i = 1; i <= n; i++) {
    scanf("%d", &p[i].time);
    p[i].num = i;
  }
  sort(p + 1, p + n + 1, cmp);
  for (int i = 1; i <= n; i++) {
    printf("%d ", p[i].num);
    sum += i * p[n - i].time;
  }
  printf("\n%.2lf\n", 1.0 * sum / n);
  return 0;
}
```

最后计算耗时总长然后得到平均值输出。

凌乱的yyy

例 12.3（洛谷 P1803）

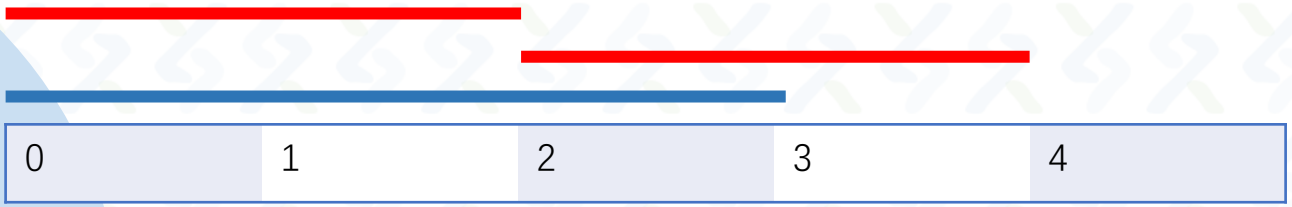
各大 OJ 上有 n 个模拟比赛，知道每个比赛开始结束时间(a_i, b_i)。

yyy 认为参加数量越多的模拟比赛越好。如果要参加一个比赛必须善始善终，而且不能同时参加两个及以上的比赛。

他想知道他最多能参加几个比赛。所有输入数据不超过10^6。

```
3
0 2
2 4
1 3
```

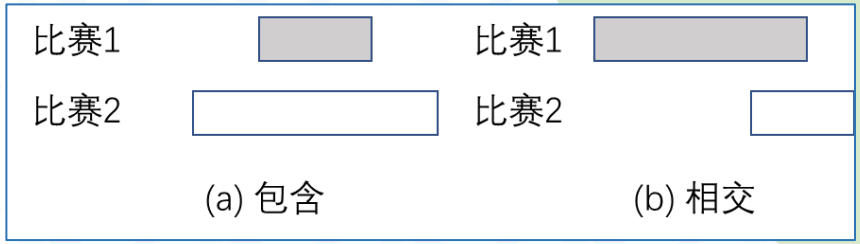
```
2
```



凌乱的yyy

如果所有的比赛时间不冲突，可以全部参加。

如果有冲突呢？



一个比赛被另一个包含：两个比赛冲突，选择比赛 1，因为比赛 1 先结束，这样后续比赛被占用时间的可能就少一些。

一个比赛和另一个比赛相交：还是选择比赛 1，理由同上。

应该选择参加最先结束的那一场比赛。

然后选择能参加的比赛中，最早结束的，直到无比赛可参加为止。

凌乱的yyy

将所有比赛的结束时间排序，然后依次进行贪心——如果能够参加这场比赛，就报名参加；如果和上一场冲突，就放弃。

```
#include<iostream>
#include<algorithm>
using namespace std;

int n, ans = 0, finish = 0;
struct contest {
  int l, r;
} con[1000010];
bool cmp(contest a, contest b) {
  return a.r <= b.r;
}
```

```
int main() {
  cin >> n;
  for (int i = 1; i <= n; i++)
    cin >> con[i].l >> con[i].r;
  sort(con + 1, con + 1 + n, cmp);
  for (int i = 1; i <= n; i++)
    if (finish <= con[i].l)
      ans++, finish = con[i].r;
  cout << ans << endl;
  return 0;
}
```

贪心本身的算法复杂度是O(n)，但是排序的算法复杂度可达O(nlogn)，所以时间复杂度的瓶颈在排序上。

考虑到值域范围不大，也可以考虑使用计数排序来优化。

小提示

一般用两种办法证明贪心成立。

反证法：假设所选方案非贪心算法所要求的方案，只需要证明将需要贪心的方案替换掉所选方案，结果会更好（至少不会更差）

数学归纳法：每一步的选择都是到当前为止的最优解，一直到最后一步就成为了全局的最优解。

可以大胆猜想贪心策略，但要保证正确性（最好能严格证明）

推翻贪心：只需要找到一个反例！

大胆假设，小心求证

~~实在没办法的话，也可不用求证~~

哈夫曼编码

哈夫曼编码是一种重要的贪心思想，而且也相当的有用。

请翻至课本 P169

分卷子

例 12.4 分卷子

将一摞试卷按照等级分类。各个等级对应的成绩：

A	B	C	D
86 到 100 分	71 到 85 分	60 到 70分	0 到 59 分

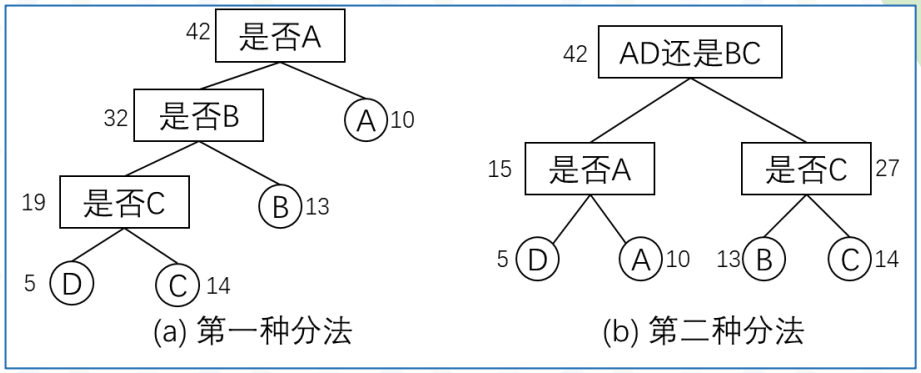
每次分卷子，只能将一摞卷子分为两堆，其中一堆包含了所有某些等级的卷子，另一堆包含所有另一些等级的卷子。

分好的卷子还能继续再分，直到分成 4 堆为止。

已知各等级卷子的数量，请设计方案使分类比较次数总和最小。

分卷子

假设 A、B、C、D 分别有 10、13、14、5 人，观察下面两个例子：



比较次数的总和还能因为不同的分类方法而不一样？

第一种分法：A 分了 1 次，B 分了 2 次，C 和 D 各分了 3 次，一共分卷次数是 $10\times1+13\times2+14\times3+5\times3=93$

第二种分法：每种卷子都分了 2 次，一共分卷次数是 $2\times(5+10+13+14)=84$

显然第二种方式次数要少一些。

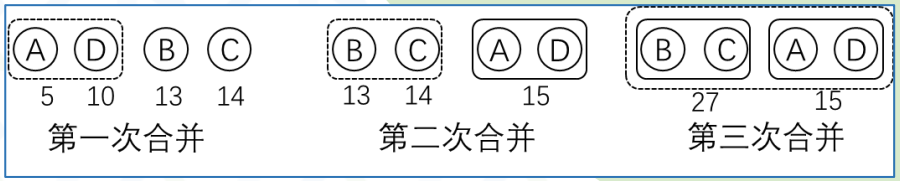
分卷子

大胆假设！

最开始分出最多的等级卷子，然后分出第二多的……直到分完。

能证明吗？

不对！有反例！



尝试逆向思维？

假设已按等级分成 4 堆卷子

先在这 4 堆卷子找到数量最少的 2 堆卷子合并为新的一堆。

然后剩下三堆卷子中再找出最少的 2 堆的卷子合并成一堆新的。

最后把这两堆卷子合并成一堆。

正解！但不太好证明
读者可自行查阅资料

合并果子

例 12.5（洛谷 P1090，NOIP2004 提高组）

果园里有很多堆水果，多多决定把所有的果子合成一堆。

每一次合并，多多可以把两堆果子合并到一起，消耗的体力等于两堆果子的重量之和。

已知果子种类数 $n(n \leq 20000)$ 和每种果子数目 $a_i(a_i \leq 10000)$，请设计出合并的次序方案，使耗费的总体力最少。

只需输出这个最小的体力耗费值即可。

3 1 2 9	15

合并果子

比较上一题，虽然一个分离，一个合并，但使用的模型一样。

所以本题只需要每次将最小的两个果堆合并成一个新堆即可。

每次从数组中进行循环，找到最小的两个？复杂度 O(n*n)！

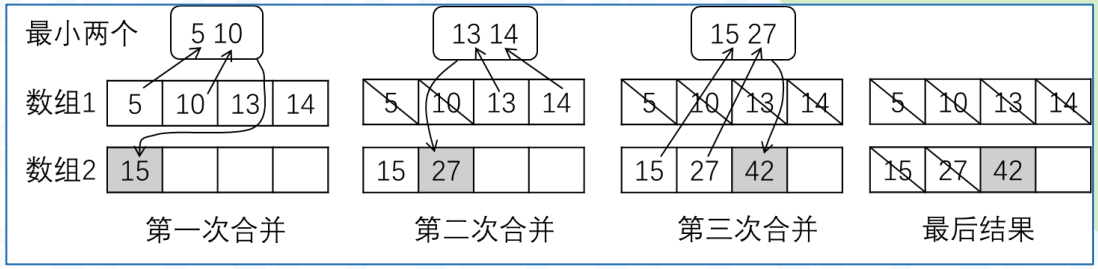
可以使用二叉堆，动态维护集合的最值，目前不讲。

还可以使用两个数组进行操作。

合并果子

数组 1 存储每堆果子重量并从小往大排序。前两个是最小的两堆。

把这两堆果子取出（从数组中划掉）合并一次成为新的一堆，记录消耗的体力，然后把这两堆果子总和放在数组 2 后面。



继续找最小堆，比较两数组中未划掉部分最前面的元素并取出。

合并后丢数组 2 的后面，两个数组都是从小到大排序的。

合并果子

可以使用变量来定位没有划掉的头部（i/j）；

还要记录下两个数组分别的元素个数（n/n2）。

这种有记录头尾位置的数组叫做队列

```
int n, n2, a1[10010], a2[10010], sum = 0;
int main() {
  cin >> n;
  memset(a1, 127, sizeof(a1)); memset(a2, 127, sizeof(a2));
  // 将数组初始化为一个接近int最大值的数，效率较高
  for (int i = 0; i < n; i++) cin >> a1[i];
  sort(a1, a1 + n);
  int i = 0, j = 0, k, w;
  for (k = 1; k < n; k++) {
    w = a1[i] < a2[j] ? a1[i++] : a2[j++]; // 取最小值
    w += a1[i] < a2[j] ? a1[i++] : a2[j++]; // 取第二次最小值
    a2[n2++] = w; // 加入第二个队列
    sum += w; // 计算价值
  }
  cout << sum;
}
```

将数组初始化为很大的数字，否则如果为 0 的可能被果堆被取出

哈夫曼编码

计算机传输数据时，必须将信息的内容编码成 0 或 1 的信息流。

比如说可以将一个字母或者数字转换成 ASCII 编码，成为 8 位的 0/1 串，但是这么编码生成出的 0/1 信息流还是比较长。

L(76) 0100 1100	u(117) 0111 0101	o(111) 0110 1111	g(103) 0110 0111	u(117) 0111 0101

可以将一些出现频数较高的字母缩短编码长度，而频度较低的字母加长编码长度以达到缩短总长度的目的。

假设信息中只有“ABCDE”这几个字母组成，其出现的次数分别是 A:5，B:10，C:13，D:14，E:20。

请参考前面的分卷子的例子，
设计一种 0/1 编码，
使编码后总长度最小。

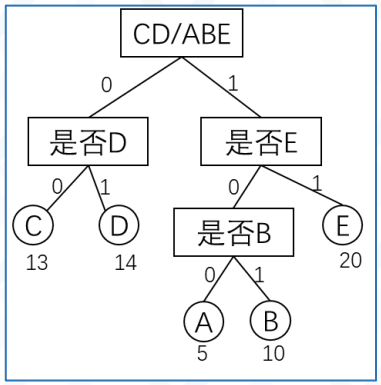
例子：DEBACEDEBDEDACBBEE
DBCCCEBDEAEDCCBEDEDBCBD
CECACBECEDDEDEEAEDCEE

哈夫曼编码

按照分卷子的做法，根据字母频率构建分类方案。

要查询某个字母的编码就相当于从头开始将字母分类，往左边分就是 0，往右边分就是 1，直到不可分为止。

这是哈夫曼树！



C：00，D：01，
A：100，B：101，
E：11

使用前面介绍的方法，可构造出一个方案（可能不唯一）

比如，单词“BAD”可以编码成“10110001”，而编码“001101”可以解码成“CED”，且不会产生歧义。这就是哈夫曼编码。

课后习题与实验

学而时习之，不亦说乎。学而不思则罔，思而不学则殆。——孔子

请翻至课本 P172

小结

贪心算法

每一步都选择最优策略，结果也可以最优。

大胆假设，小心求证。找到一组反例就可以推翻贪心！

证明：反证法、数学归纳法。

哈夫曼编码

一般用于压缩编码，可以使编码变短。

从数量最少的元素开始合并建立哈夫曼树。

使用两个队列维护最小的两个元素。

课后习题

对于以下的题目，请读者尽可能先猜测贪心策略，然后设法证明。

习题 12.1 小A的糖果（洛谷 P3817）

小A 有 $N(N \leq 10^5)$ 个糖果盒，第i个盒中有$a_i(a_i \leq 10^9)$颗糖果。

小A 每次可从其中一盒中吃掉一颗，要让任意两个相邻的盒子中加起来不多于 $x(x \leq 10^9)$ 颗糖果，至少得吃掉几颗糖？

习题 12.2 删数问题（洛谷 P1106）

输入一个高精度的正整数 N（不超过 250 位），去掉其中任意 k 个数字后剩下的数字按原左右次序将组成一个新的正整数。

编程对给定的 N 和 k，寻找一种方案使得剩下的数字组成的新数最小，输出新数即可。

课后习题

习题 12.3 陶陶摘苹果 - 升级版（洛谷 P1478）

一棵苹果树结出 $n(n \leq 5000)$ 个苹果。陶陶手伸直最大长度 $b(b \leq 200)$。陶陶有个 $a(a \leq 50)$ 厘米高的板凳，当她不能直接用手摘到苹果时就会踩到板凳上。

陶陶只剩下 $s(s \leq 1000)$ 点力气了，但是摘掉每个苹果都要花费一些力气值。陶陶没打算透支体力，所以体力值必须一直不小于 0。

现在已知苹果到地面的高度 $x_i(x_i \leq 280)$ 和摘这个苹果需要的力气 $y_i(y_i \leq 100)$，请算一下最多能够摘到的苹果的数目。

课后习题

习题 12.4 铺设道路 （洛谷 P5019，NOIP2018 提高组）

有一条 n 块的道路，第 i 块区域下陷深度为 d_i 。每次可选一段连续区间$[L,R]$，让这个区间下陷深度减少 1，深度不小于 0。

请设计一种方案以最少次数将整段道路的下陷深度变为 0 。

习题 12.5 混合牛奶（洛谷 P1208，USACO Training）

某公司从 n 名奶农手中采购牛奶。每一位奶农为乳制品单价 p_i 报价是不同的。此外，每位奶农能提供的牛奶数量 a_i 也是一定的。

该公司对牛奶的需求量 N，可采购每个奶农小于等于 a_i 的牛奶，保证总产量大于需求。求采购足够数量的牛奶所需最小花费。

课后习题

习题 12.7 跳跳！（洛谷 P4995）

青蛙遇到 $n(n \leq 300)$ 块不同的石头，其中石头 i 高度为 $h_i(h_i \leq 10\text{^}4)$，地面高度是 $h_0 = 0$。从石头 i 跳到石头 j 上耗费的体力为 $\left(h_i - h_j\right)^2$，从地面跳到第 i 块石头耗费的体力值是 $(h_i)^2$。

青蛙跳到每个石头各一次，耗费最多体力值是多少？

习题 12.8 分组（洛谷 P4447）

有 $n(n \leq 10^5)$个学生，每个人的实力值是 $a_i(|a_i| \leq 10^9)$，要分成若干个小组。每个小组成员实力值排序后，必须是连续且互不相同的整数数列。

要求设计一个合法的分组方案，满足所有人都恰好分到一个小组，使得人数最少的组人数最多，输出人数最少的组人数的最大值即可。