一、基本概念：

所谓贪心算法是指，在对问题求解时，总是做出在当前看来是最好的选择。也就是说，不从整体最优上加以考虑，他所做出的仅是在某种意义上的局部最优解。

贪心算法没有固定的算法框架，算法设计的关键是贪心策略的选择。必须注意的是，贪心算法不是对所有问题都能得到整体最优解，**选择的贪心策略必须具备无后效性**，即某个状态以后的过程不会影响以前的状态，只与当前状态有关。

所以对所采用的贪心策略一定要仔细分析其是否满足无后效性。

二、贪心算法的基本思路：

1.建立数学模型来描述问题。

2.把求解的问题分成若干个子问题。

3.对每一子问题求解，得到子问题的局部最优解。

4.把子问题的解局部最优解合成原来解问题的一个解。

三、贪心算法适用的问题

贪心策略适用的前提是：局部最优策略能导致产生全局最优解。

实际上，贪心算法适用的情况很少。一般，对一个问题分析是否适用于贪心算法，可以先选择该问题下的几个实际数据进行分析，就可做出判断。

四、贪心算法的实现框架

```
从问题的某一初始解出发；
while （能朝给定总目标前进一步）
{
    利用可行的决策，求出可行解的一个解元素；
}
由所有解元素组合成问题的一个可行解；
```

五、贪心策略的选择

因为用贪心算法只能通过解局部最优解的策略来达到全局最优解，因此，一定要注意判断问题是否适合采用贪心算法策略，找到的解是否一定是问题的最优解。

六、例题分析

下面是一个可以试用贪心算法解的题目，贪心解的确不错，可惜不是最优解。

[背包问题]有一个背包，背包容量是M=150。有7个物品，物品可以分割成任意大小。

要求尽可能让装入背包中的物品总价值最大，但不能超过总容量。

物品 A B C D E F G

重量 35 30 60 50 40 10 25

价值 10 40 30 50 35 40 30

分析：

目标函数： ∑pi最大

约束条件是装入的物品总重量不超过背包容量：∑wi<=M(M=150)

（1）根据贪心的策略，每次挑选价值最大的物品装入背包，得到的结果是否最优?

（2）每次挑选所占重量最小的物品装入是否能得到最优解?

（3）每次选取单位重量价值最大的物品，成为解本题的策略。

值得注意的是，贪心算法并不是完全不可以使用，贪心策略一旦经过证明成立后，它就是一种高效的算法。

贪心算法还是很常见的算法之一，这是由于它简单易行，构造贪心策略不是很困难。

可惜的是，它需要证明后才能真正运用到题目的算法中。

一般来说，贪心算法的证明围绕着：整个问题的最优解一定由在贪心策略中存在的子问题的最优解得来的。

对于例题中的3种贪心策略，都是无法成立（无法被证明）的，解释如下：

（1）贪心策略：选取价值最大者。反例：

W=30

物品：A B C

重量：28 12 12

价值：30 20 20

根据策略，首先选取物品A，接下来就无法再选取了，可是，选取B、C则更好。

（2）贪心策略：选取重量最小。它的反例与第一种策略的反例差不多。

（3）贪心策略：选取单位重量价值最大的物品。反例：

W=30

物品：A B C

重量：28 20 10

价值：28 20 10

根据策略，三种物品单位重量价值一样，程序无法依据现有策略作出判断，如果选择A，则答案错误。