贪心算法

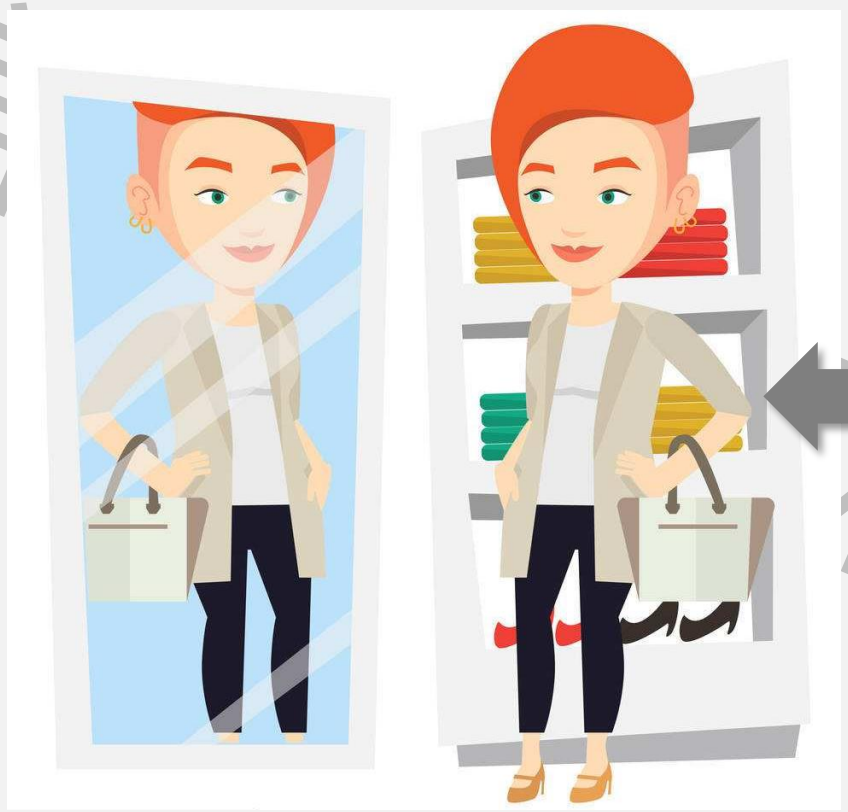
Greedy

1.什么是贪心?

什么是贪心算法?

所谓贪心算法是指，在对问题求解时，总是做出在**当前**看来最好的选择。

也就是说，不从整体最优解出发来考虑，它所做出的仅是在某种条件下的局部最优解。简单的说就是：**只顾当前**!



引例A：找零

某便利店店员需要向客户找零97元**人民币**，问最少需要多少张钱?

贪心原则：尽量用面值大的纸币。

1.讨论面值100，100>97，不行;
2.讨论面值50，50<=97，可行，张数+=97/50，余钱=97%50=47;
3.讨论面值20，20<=47，可行，张数+=47/20，余钱=47%20=7;
4.讨论面值10，10>7，不行;
5.讨论面值5，5<=7，可行，张数+=7/5，余钱=7%5=2;
6.讨论面值1，1<=2，可行，张数+=2/1，余钱=2%1=0，结束;

为什么上述方法是正确的呢?

我们考虑如果任意两种面值交换一下使用顺序， 这样所需张数会不会更少？

不会!

这是一种常用的证明贪心正确性的方法，称为“**邻项交换法**”。

引例B：最大整数

设有n个正整数(n<=10000,long long范围内)，将它们联接成一排组成一个最大的多位整数。

例如:n=3时,3个整数13, 312, 343联接成的最大整数为:34331213。

又如:n=4时,4个整数7, 13, 4, 246联接成的最大整数为:7424613。

贪心策略：“字典序”大的放左边

1.将n个数按“字典序”由大到小排序

2.将排序后的数字依次输出即可

注意特殊情况 “9”、“97”、“977”

排序时 cmp return x+y>y+x;

证明：邻项交换法

任意交换相邻两个数，都不可能得到更优的结果。

```
string s[100];

bool cmp(string x,string y)
{
    return x+y>y+x;
}

int main()
{
    int n;
    cin>>n;
    for(int i=1;i<=n;i++)cin>>s[i];
    sort(s+1,s+1+n,cmp);
    cout<<endl<<"-------排序后----------"<<endl<<endl;
    for(int i=1;i<=n;i++)cout<<s[i]<<endl;
}
```

2. 贪心思想的应用
第一节

例1：排队打水 nkoj5215

有n个人排队到r个水龙头去打水，他们装满水桶的时间T_1、T_2……T_n为整数且各不相等，应如何安排他们的打水顺序才能使每个人花费的时间总共和最少？

输入格式：

第一行n，r (n<=1000,r<=100)

第二行为n个人打水所用的时间Ti (1<=Ti<=100)

输出格式： 一个整数，最少花费的总时间

样例输入：

```
3 2
1 2 3
```

样例输出：

```
7
```

样例说明：

第1，2两人在1号龙头接水，2排1后。

第1个人接水，耗时1。　第2个人排队耗时1，接水耗时2，总耗时3。

第2个人在2号龙头接水，耗时3。

总耗时1+3+3=7

例1：排队打水 解题分析

贪心原则：一遇到空闲的水龙头，就让**接水耗时少的人优先打水。**

证明：

设有1个水龙头，4个人打水，接水时间 T1 < T2 < T3 < T4

我们**按耗时小的优先打水**的贪心原则处理，总耗时Sum为：

Sum=T1+(T1+T2)+(T1+T2+T3)+(T1+T2+T3+T4)=4*T1+3*T2+2*T3+T4

越早打水的人，被累加的次数越多。

根据"邻项交换法"：任意交换两个人的的打水顺序，得到的结果一定**不比**上面的小

所以，贪心策略是正确的。

例1：排队打水 解题分析

贪心原则：一遇到空闲的水龙头，就让**接水耗时少的人优先打水。**


```
sort(T+1,T+n+1);                    //按接水时间由小到大排序
Ans=0;
for (int i=1,j=0; i<=n; i++)
{
      j++;
      if(j==r+1) j=1;               //共r个水龙头，前r个人为一组，第r+1个人回到第一个水龙
      Sum[j]+=T[i];                 //第i个人的耗时，即第j号水龙头接下来排队需要的耗时
      Ans+=Sum[j];
  }
```

3. 应用贪心的步骤

贪心问题解题步骤：

- 1.观察分析规律，**发现贪心策略**；
- 2.验证贪心策略是否正确：

  常用方法：邻项交换法、列举反例、范围缩放法、数学归纳和反证法等
- 3.程序实现

接下来，我们重点讨论：**发现贪心策略**和**验证贪心策略正确性**这两个核心问题

4. 贪心思想的应用
第三节

例7：打怪 nkoj5233

在一款电脑游戏中，你需要打败n只怪物（从1到n编号）。

为了打败第i只怪物，你需要消耗D[i]点生命值，但怪物死后会掉落血药，使你恢复A[i]点生命值。任何时候你的生命值都不能降到0（或0以下）。

问是否存在一种打怪顺序，使得你可以打完这n只怪物而不死掉

1<=n,z<=100000

例7：打怪 解题分析

贪心策略：

1.先打会加血(D[i]<=A[i])的怪，再打会减血(D[i]>A[i])的怪；

2.对于会加血的怪，按D[i]值由小到大的顺序去打；

3.对于会减血的怪，按A[i]值由大到小的顺序去打；

加血的情况显然，下面证明会减血的情况：

设有两个怪，对应得参数分别为D1,D2,A1,A2，且A1>A2， D1>A1 ，D2>A2

若先打1号再打2号，失血分别为D1 和 D1-A1+D2

若先打2号再打1号，失血分别为D2 和 D2-A2+D1

因为D1>A1且D2>A2

有D2+D1-A1<D2+D1-A2