> 假设盘子上有n块面积大小不一的烧饼，你如何用一把锅铲进行若干次翻转，让这些烧饼的大小有序（小的在上，大的在下）？

分析：

1. 找到n个饼中最大的一个；
2. 将这个饼翻转到最下方；
3. 对上面n-1个饼进行排序；
4. 当1 == n时，无需排序。

```
std::vector<int> pancakeSort(std::vector<int> &cakes) {
  std::vector<int> res; // 记录翻转序列
  sort(cakes, n , res);

  return res;
}

void sort(std::vector<int> &cakes, int n, std::vector<int> &res){
  if(1 == n) {
    // base case
    return ;
  }

  // 寻找最大饼索引
  int maxCakes = 0;
  int maxCakesIndex = 0;
  for(int i =0;i<n;i++){
    if(cakes[i] <= maxCakes) {
      maxCakes = cakes[i];
      maxCakesIndex = i;
    }
  }

  // 第一次翻转将最大饼翻转到最上面
  revese(cakes, 0, maxCakesIndex);
  res.push_back(maxCakesIndex+1);
  // 第二次翻转将最大饼移动到最下面
  revese(cakes,0,n-1);
  res.push_baxk(n);

  // 递归调用
  sort(cakes, 0, n-1);
}

void reverse(std::vector<int> &cakes, int i, int j) {
  while(i < j) {
    swap(cakes[i++], ckase[j--]);
  }
```

```
}
```