user: sakura

第45题——最小的k个数字

输入整数数组 arr ， 找出其中最小的 k 个数字。例如，输入4、5、1、6、2、7、3、8这8个数字，则最小的4个数字是1、2、3、4。

示例 1：

```
输入：arr = [3,2,1], k = 2
```

```
输出：[1,2] 或者 [2,1]
```

示例 2：

```
输入：arr = [0,1,2,1], k = 1
```

```
输出：[0]
```

限制：

0 <= k <= arr.length <= 10000 0 <= arr[i] <= 10000

笨办法(1)

最为一个菜鸟,我看到这道题第一时间想到的就是给它排序

然后返回排序后的数组的前K个元素即可

但是呢,我先用了冒泡排序,超时了,,尴尬(冒泡就是个垃圾算法)

第二次直接换了希尔排序,虽然通过了,但是耗时比较长...

```
class Solution {
public:
    vector<int> getLeastNumbers(vector<int>& arr, int k) {
        vector<int> res;
        if(arr.empty()|| k==0)
        {
            return res;
        }
        //希尔排序
        shellSort(arr,arr.size());
        for(int i=0;i<k;i++)
```

```
        {
            res.push_back(arr[i]);
        }
        return res;
    }
    void shellSort(vector<int>& a, int len)
    {
        int i, j, k, tmp, step;  // step 为步长
        for (step = len / 2; step > 0; step /= 2) {
            // 步长初始化为数组长度的一半，每次遍历后步长减半，
            for (k = 0; k < step; k++) { // 变量 i 为每次分组的第一个元素下标
                for (i = k + step; i < len; i += step) {
                    //对步长为step的元素进行直插排序，当step为1时，就是直插排序
                    tmp = a[i];  // 备份a[i]的值
                    j= i- step;  // j初始化为i的前一个元素（与i相差step长度）
                    while (j >= 0 && a[j] > tmp) {
                        a[j + step] = a[j]; // 将在a[i]前且比tmp的值大的元素向后移动一位
                        j -= step;
                    }
                    a[j+step] = tmp;
                }
            }
        }
    }
};
```

作者：sakura7301
链接：https://leetcode-cn.com/problems/zui-xiao-de-kge-shu-lcof/solution/pai-xu-fan-hui-qian-nge-yuan-su-by-sakur-dclz/
来源：力扣（LeetCode）

algorithm内置sort排序(2)

有一说一,内置的sort算法确实比我手撸的快多了

```
class Solution {
public:
    vector<int> getLeastNumbers(vector<int>& arr, int k) {
        vector<int> res;
        if(arr.empty()|| k==0)
        {
            return res;
        }
        //algorithm内置sort算法排序
        sort(arr.begin(),arr.end());
        for(int i=0;i<k;i++)
        {
            res.push_back(arr[i]);
        }
        return res;
    }
```

```
};
```