**顺序查找**又称为线性查找，它是一种最简单、最进本的查找方法。它从顺序表的一端开始，依次将每一个数据元素的关键字值与给定值key做比较，若某个元素的关键字值等于给定值key，则表明查找成功；若知道所有数据元素都比较完毕，仍然找不到关键字值为key的数据元素，则表明查找失败。

（1）不带监视哨

public static int seqSearch(Object[] nums,Object key){

int i=0;

while(i<nums.length&&!nums[i].equals(key))

i++;

if(i < nums.length)

return i;

return -1;

}

不带监视哨会比较i的位置是否越界，同时还比较了元素是否相等，比较麻烦

（2）带监视哨

带监视哨就是将表的开头位置设置一个监视哨，此时带查找的记录应该存放在顺序表的1到n的位置。

public static int seqSearchWithGurad(Object[] nums,Object key){

int i = nums.length-1; //从尾巴开始遍历

nums[0]=key; //将表第一个数据nums[0]改为key，当成监视哨，当比较到监视哨的时候比较会停

while(!nums[i].equals(key)) //不等于向下遍历

i--;

if(i>0) //

return i;

return -1;

}

带监视哨可以减少一个比较条件：查看当前位置是否越界。因为从尾巴开始到开头，头的位置等于key，即便没有找到到头也会停下，不用担心是否越界的问题。监视哨也可以放在表尾，这样从头开始遍历到表尾也会停下。

**带监视哨的顺序查找ASL**

顺序查找第i条记录，在等概率的条件下Pi=1/n

**查找成功**，需要与关键字的比较次数为  **n-i+1**

ASL =1/n=**(n+1)/2**

**查找失败**，需要与关键字的比较次数为 **n+1**

时间复杂度为O(n)，效率较低

顺序查找的**优点**

1.适用于线性表，也适用于单链表

2.对表中数据元素的排列次序无任何要求

3.方便在表中插入新的数据

但是，顺序查找的平均查找长度较大，不适合用于表长较大的查找表。

顺序查找的**优化方法**

1.**已知**，各个数据**元素的查找概率不等**的情况下，将各个数据元素按照查找概率从小到大进行排序，降低了查找的平均比较次数

2.在事先**未知**各个数据**元素查找概率**的情况下，在每次查找的一个数据元素时，就将他与前驱元素对调位置，这样过一段时间后，查找频率高的数据元素就会被逐渐前移，最后形成数据元素的位置按照查找概率从大到小排列，达到减少平均查找长度的目的。