本文内容是对二分查找的梳理和总结，本文内容包括：

二分查询(Binary Search)

1 二分查询(Binary Search)

一：二分查找

（Binary Search）也称折半查找，它是一种效率较高的查找方法。但是，折半查找要求线性表必须采用顺序存储结构，而且表中元素按关键字**有序排列**。

二：查找过程

首先，假设表中元素是按升序排列，将表中间位置记录的关键字与查找关键字比较，如果两者相等，则查找成功；否则利用中间位置记录将表分成前、后两个子表，如果中间位置记录的关键字大于查找关键字，则进一步查找前一子表，否则进一步查找后一子表。重复以上过程，直到找到满足条件的记录，使查找成功，或直到子表不存在为止，此时查找不成功。

三：算法要求

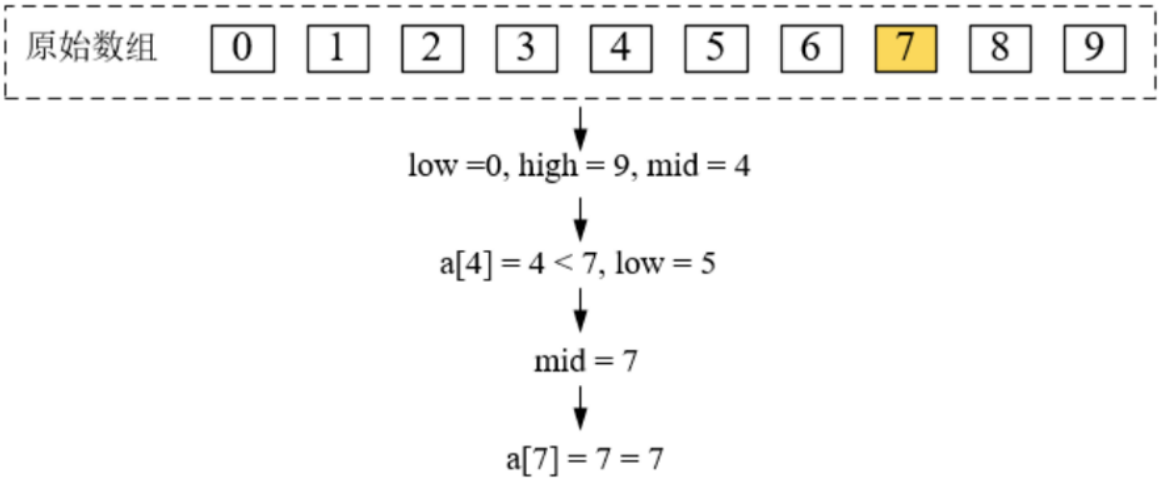
1. 必须采用顺序存储结构；
2. 必须按关键字大小有序排列。

如果数据没有排序，只能使用顺序查找；如果数据已经排好序，可以使用速度比较快的折半查找（二分查找）。

四：代码示例

例如： 使用二分查找算法在数组 { 0,1,2,3,4,5,6,7,8,9 }中查找元素7。

二分查找的详细过程如下图所示。




```java
    public static void main(String[] args) {
        int num[] = {0,1,2,3,4,5,6,7,8,9};
        int index = binarySearch(num,7);
        System.out.println(index);
    }
    public static int binarySearch(int[] srcArray, int des) {
        //定义初始最小、最大索引
        int start = 0;
        int end = srcArray.length - 1;
        //确保不会出现重复查找，越界
        while (start <= end) {
            //计算出中间索引值
            int middle = (end + start)>>>1 ;//防止溢出
            if (des == srcArray[middle]) {
                return middle;
                //判断下限
            } else if (des < srcArray[middle]) {
                end = middle - 1;
                //判断上限
            } else {
                start = middle + 1;
            }
        }
        //若没有，则返回-1
        return -1;
    }
```