二分查找

【基本思想】

基本思想是二分法，元素必须是有序的。对于一个有序序列，我们需要将待查找元素与序列的中间元素比较。如果小于中间元素，那么待查找元素在中间元素的前半段；如果大于中间元素，待查找元素在中间元素的后半段。然后对剩下的包括待查找元素一段序列继续用上述二分法继续查找，递归进行，直到查找到或查找结束。

【复杂度】

期望时间复杂度为$O(log_2n)$

【局限性】

折半查找的前提条件是需要有序表顺序存储，对于静态查找表，一次排序后不再变化，折半查找能得到不错的效率。但对于需要频繁执行插入或删除操作的数据集来说，维护有序的排序会带来不小的工作量，那就不建议使用

【代码】

```
int BinarySearch1(int a[], int value, int n)
{
    int low, high, mid;
    low = 0;
    high = n-1;
    while(low<=high)
    {
        mid = (low+high)/2;
        if(a[mid]==value)
            return mid;
        if(a[mid]>value)
            high = mid-1;
        if(a[mid]<value)
            low = mid+1;
    }
    return -1;
}
```