THE BINARY SEARCH TECHNIQUE

If members of an array being searched are arranged in alphabetical or numeric order the Binary search technique can provide a much faster searching method.

Imagine you are looking for a name in a telephone book. If you followed the logic of the sequential search, you would start looking for the name on page one and look at each page in turn until you found the name you were looking for.

Of course you would never do that, it would be too slow. What you would actually do is open the book where you think the name might be. If you found that you had overshot the name, you would next look closer to the front of the book, and if you found that you had undershot the name, you would next look closer to the back of the book. This is the basic idea behind the Binary search technique.

Example (Case 1)

Suppose we are given an array called A of 12 names in alphabetical order. We are given the name 'Fell' (stored in the variable NAME) and we want to search the array A to determine the position where 'Fell' is stored.

We will compare 'Fell' with a particular member of the Array A. If the member is greater than 'Fell', we next look closer to the beginning of the Array A. If the member is less than 'Fell', we next look closer to the end of array A.

As the search proceeds we will discover that some sections of the array could not possibly contain the name we are looking for, for we will eliminate those sections from the search. It will be helpful to introduce two new variables L and H. L is the subscript of the lowest member of the Array A that is still in the search and H is the subscript of the highest member that is still in the search. At the start L=1 and H=12 since the whole array has to be searched.

As in the sequential search we will use J as the subscript of the member we are currently comparing against the given name 'Fell'. We begin by comparing 'Fell' against the middle member of the array. You calculate the subscript of the middle member by the equation J=int((L+H)/2). The first value of J calculated by this equation is 6. So we compare the 6^{th} member of the Array A, 'Holmes', against 'Fell'. Since 'Holmes' is greater than 'Fell' (in the alphabetical sense), we know that 'Fell' must be in the lower half of the Array A. Now we can concentrate on the first five members of Array A.

To search the lower half of Array A we move the upper limit H down to J-1 or 5. Then we use our equation to calculate a new J. With L=1 and H=5, J=3. So we now compare the 3^{rd} member of array A, 'Campion' against 'Fell'. Since 'Campion' is less than 'Fell' we know that 'Fell' must be in the upper half of the array we are searching. It must be between J+1 and H. To search this section of Array A we move the lower limit L up to J+1 or 4. With L=4 and H=5 the equation gives J=4. So we now compare the 4^{th} member of Array A and since it is 'Fell' we have found a match and the search is over.

<table>
<tr><td></td><td></td><td>1^{ST} COMPARISON</td><td>2^{ND}</td><td>3^{RD}</td></tr>
<tr><td>POSITION</td><td>MEMBER</td><td>L=1 H=12 J=6</td><td>L=1 H=5 J=3</td><td>L=4 H=5 J=4</td></tr>
<tr><td>1</td><td>Alleyn</td><td rowspan="12">J=6</td><td rowspan="5">J=3</td><td rowspan="3">ELIMINATED</td></tr>
<tr><td>2</td><td>Appleby</td></tr>
<tr><td>3</td><td>Campion</td></tr>
<tr><td>4</td><td>Fell</td><td rowspan="2">J=4</td></tr>
<tr><td>5</td><td>Fen</td></tr>
<tr><td>6</td><td>Holmes</td><td rowspan="7">ELIMINATED</td><td rowspan="7">ELIMINATED (from last comparison)</td></tr>
<tr><td>7</td><td>Marple</td></tr>
<tr><td>8</td><td>Pairot</td></tr>
<tr><td>9</td><td>Queen</td></tr>
<tr><td>10</td><td>Stout</td></tr>
<tr><td>11</td><td>Wimsey</td></tr>
<tr><td>12</td><td>Wolfe</td></tr>
</table>

Sequential vs Binary Search

SIZE OF ARRAY	MAX # OF COMPARISONS BINARY SEARCH	MAX # OF COMPARISONS SEQUENTIAL SEARCH
10	4	5
100	7	50
1000	10	500

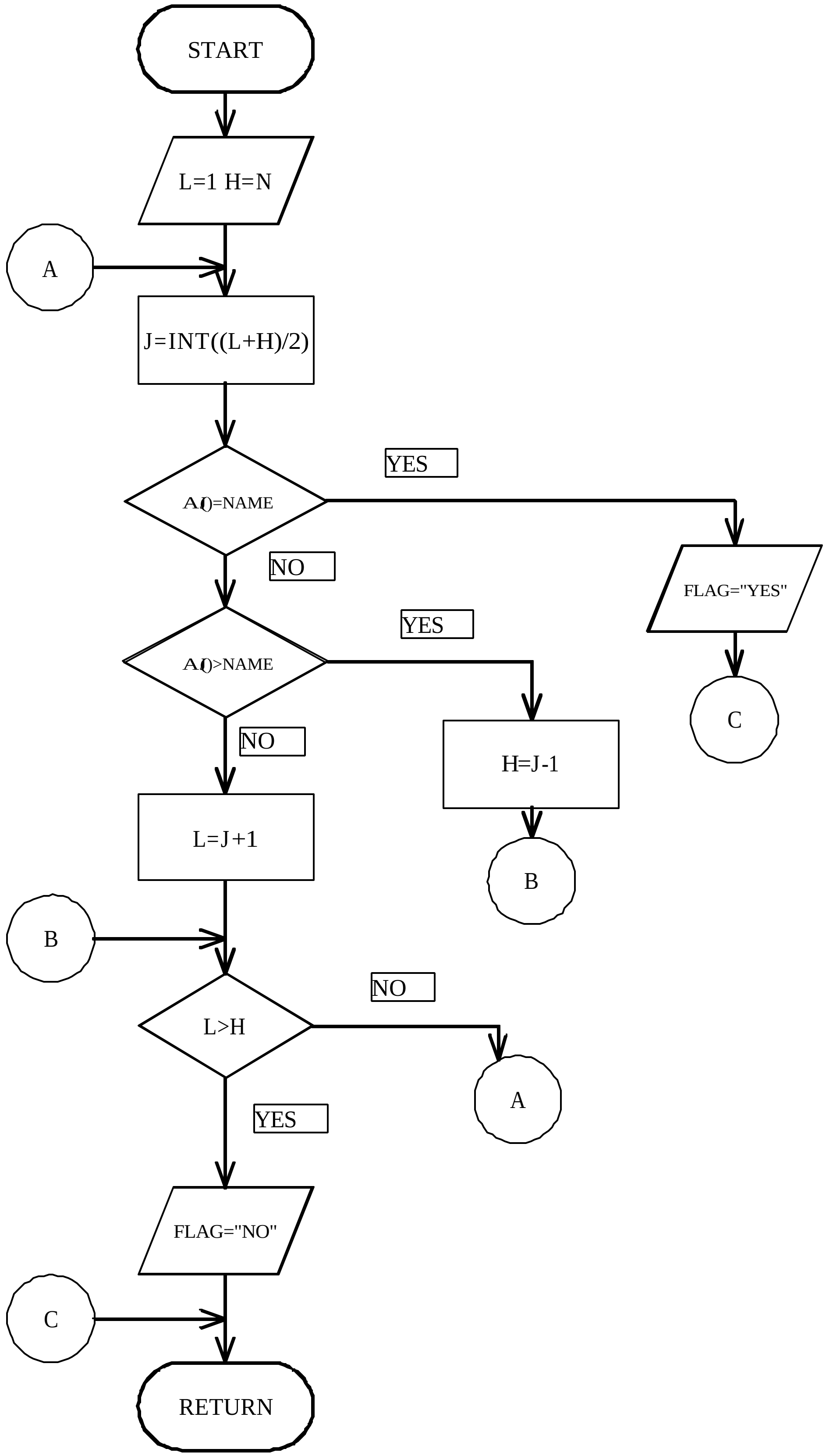
Example (Case 2)

We must now investigate what happens when we search for a name that is ***not*** in the array. Suppose we are given the name ‘Thatcher’ and we want to find the matching member in the array A. We begin as before, by comparing the sixth member of the Array A, ‘Holmes’ against ‘Thatcher’. Since ‘Holmes’ is less than ‘Thatcher’, we know that if ‘Thatcher’ is anywhere in the array A, it must be in the upper half. To search the upper half of array A we move the lower limit L up to J+1 or 7.

With L=7 and H=12 , J=9. So we next compare the 9^{th} member of array A, ‘Queen’ against ‘Thatcher’. Since ‘Queen’ is less than ‘Thatcher’ we know that if ‘Thatcher’ is anywhere in our array it must be between J+1 and H. With L=10 and H=12 we find J=11, so we next compare the 11^{th} member of array A, ‘Wimsey’ against ‘Thatcher’. Since ‘Wimsey’ is greater than ‘Thatcher’ we move H down to J-1=10.

With L=10 and H=10 J=10, so we next compare ‘Stout’ against ‘Thatcher’. Since ‘Stout’ < ‘Thatcher’ we move L up to J+1=11. But here we have run into an impossibility, the lower limit is 11 while the upper limit is 10. This crossing of L and H tells us that we have searched the complete array and that the name we were searching for is not in the array. So the Binary search ends either when we find a match, which means the search was successful, or when the upper and lower limits cross, which means that the search was unsuccessful.

<table>
<tr><td></td><td></td><td>1^{st} COMPARISON</td><td>2^{ND}</td><td>3^{RD}</td><td>4^{TH}</td></tr>
<tr><td>POSITION</td><td>MEMBER</td><td>L=1 H=12 J=6</td><td>L=7 H=12 J=9</td><td>L=10 H=12 J=11</td><td>L=10 H=10 J=10</td></tr>
<tr><td>1</td><td>Alleyn</td><td rowspan="12">J=6</td><td rowspan="6">ELIMINATE</td><td rowspan="9">ELIMINATE</td><td rowspan="9">ELIMINATE</td></tr>
<tr><td>2</td><td>Appleby</td></tr>
<tr><td>3</td><td>Campion</td></tr>
<tr><td>4</td><td>Fell</td></tr>
<tr><td>5</td><td>Fen</td></tr>
<tr><td>6</td><td>Holmes</td></tr>
<tr><td>7</td><td>Marple</td><td rowspan="6">J=9</td></tr>
<tr><td>8</td><td>Pairot</td></tr>
<tr><td>9</td><td>Queen</td></tr>
<tr><td>10</td><td>Stout</td><td rowspan="3">J=11</td><td>J=10</td></tr>
<tr><td>11</td><td>Wimsey</td><td rowspan="2">ELIMINATE</td></tr>
<tr><td>12</td><td>Wolfe</td></tr>
</table>



START
L=1 H=N
A
J=INT((L+H)/2)
A(J)=NAME
YES
FLAG="YES"
C
NO
A(J)>NAME
YES
H=J-1
B
NO
L=J+1
B
L>H
NO
A
YES
FLAG="NO"
C
RETURN