S12_2

Searching Techniques

Objectives

To learn and appreciate the following concepts

Searching Technique

- **Linear Search**
- **Binary Search**

Session outcome

- At the end of session student will be able to understand
  - Searching Techniques

Arrays – A recap

1D Array:

Syntax: **type array_name[size];**

- **Initialization:**

type array-name [size]={list of values}

- **Read:**

```
for(i=0;i<n;i++)

scanf("%d",&a[i]);
```

Write:

```
for(i=0;i<n;i++)

printf("%d",a[i]);
```

Searching

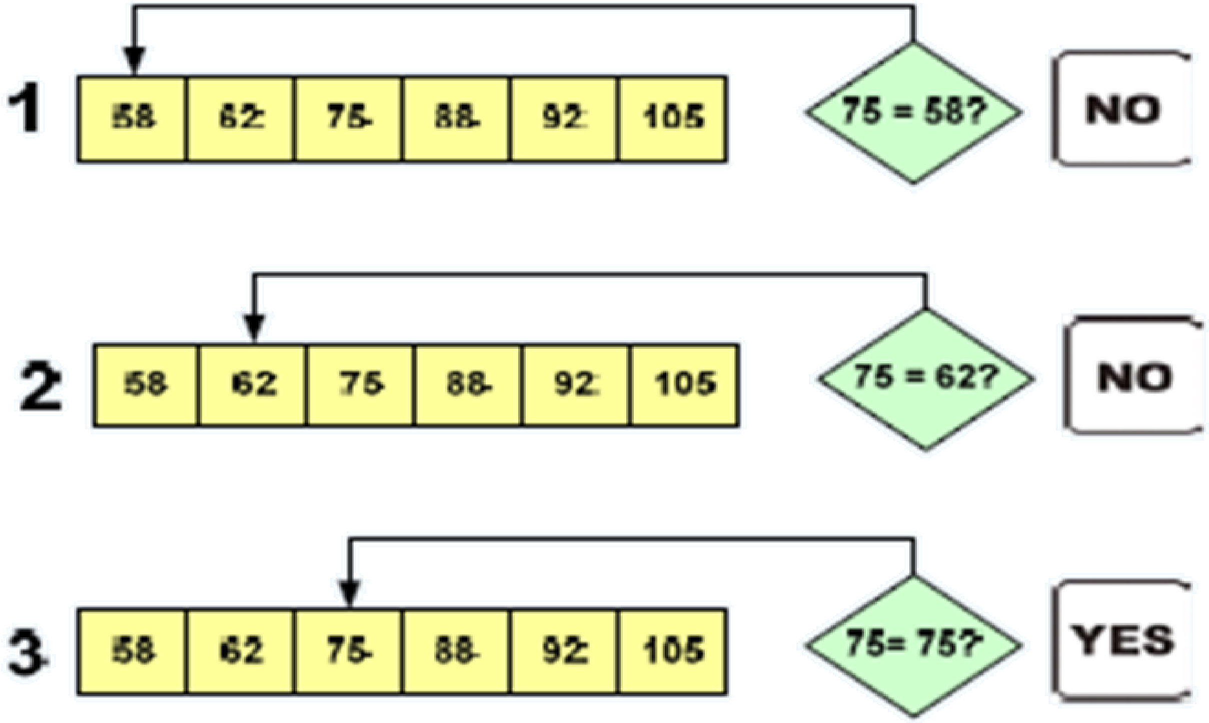
- Finding whether a data item is present in a set of items

→ **linear** search / sequential search

Linear search- illustration 1

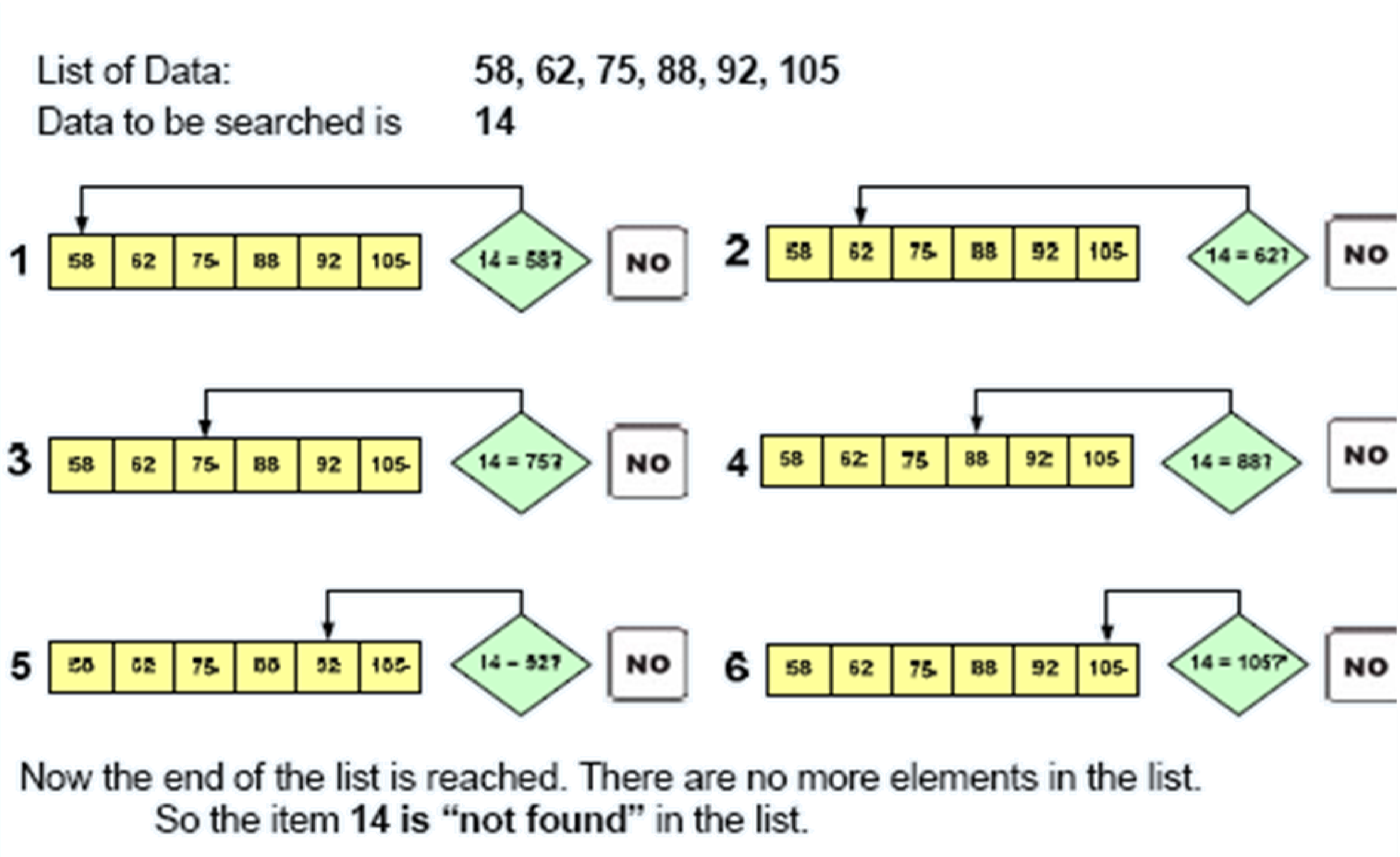
List of Data: 58, 62, 75, 88, 92, 105

Data to be searched is 75



The “item is found” and stop the searching process

Linear search- illustration 2



List of Data: 58, 62, 75, 88, 92, 105

Data to be searched is 14

Now the end of the list is reached. There are no more elements in the list.
So the item **14 is "not found"** in the list.

Pseudo code for linear search

```
int found=0; //setting flag

Print "enter no of numbers";

Input n;

for(i=0;i<n;i++){

Print "enter number\n";

Input a[i]; // entered data items

}

Print "enter the element to be

searched";

Input key; // data to be searched
```

```
/*search procedure*/
for(i=0; i<n; i++) {
if(a[ i ]==key)  // comparison
  {
     found=1;
     pos=i+1;
     break;
   } }
if(found==1)
   Print"data_found_in",pos,
"position";
otherwise
    Print "data is not found";
```

Binary Search

- A binary search is a searching technique that can be applied only to a **sorted list** of items
- This searching technique is similar to dictionary search.

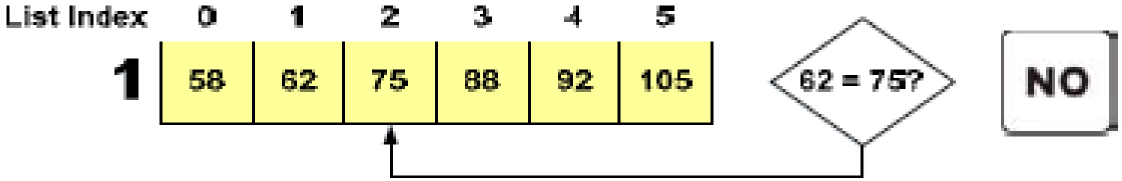
- **Algorithm:**
  - **Step 1**: Set First = 0 and Last = Number of Items – 1
  - **Step 2**: Find the middle of the list as mid = (First + Last) /2. Take only the integer part, if the result is a real number.
  - **Step 3**: Compare the middle item with the searching item. If they are equal then "**Item is found**" and go to step 8.
  - **Step 4**: If the searching item is less than the middle item then the searching item comes before this middle element. So, set Last = mid -1 and there is no change in the value of First. Go to step 6.
  - **Step 5**: Since the above conditions are false the searching element should be greater than the middle element. So, set First = mid +1 and there is no change in the value of Last. Go to the next step.
  - **Step 6**: If First <= Last then go to step 2.
  - **Step 7**: Since end of the list is reached, the searching item is "**not found**" in the list
  - **Step 8**: End of the algorithm.

Binary Search – example-1

List of Data: **58, 62, 75, 88, 92, 105**

Data to be searched is **62**

- Step 1: First = 0 and Last = 5
- Step 2: Step 2: Mid = (0 + 5) / 2. That is Mid = 2 (Only the integer part is taken)
- Step 3:



List Index	0	1	2	3	4	5
1	58	62	75	88	92	105

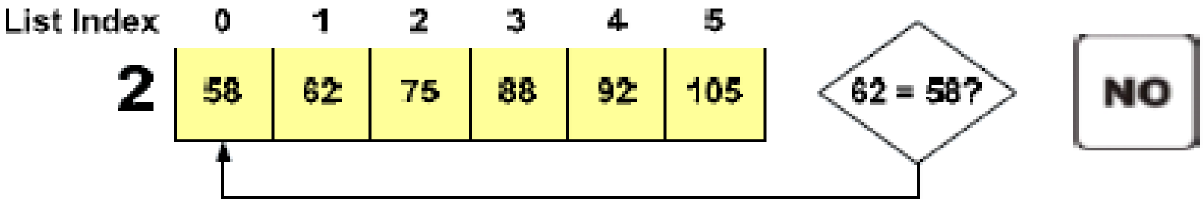
62 = 75?

NO

- Step 4: The searching item 62 is less than 75. So it should appear before 75. Now First = 0 and Last = mid-1 that is Last = 2-1. So Last = 1
- Step 5: Compute Mid = (0 + 1) / 2 that is Mid = 0 (Integer part)

Binary Search – example-1

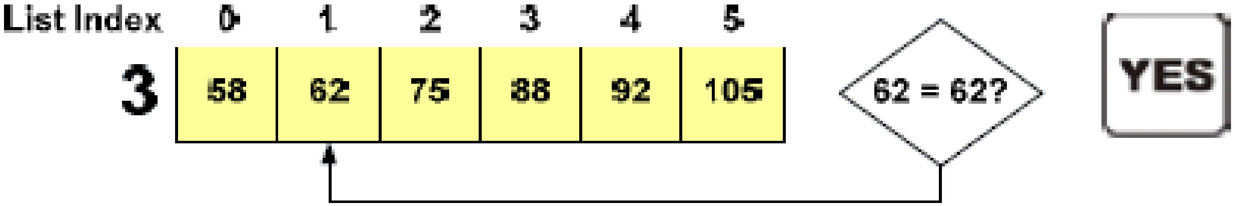
- Step 6: Compare 0th item with 62. That is compare 58 and 62. Since they are not equal proceed to the next step



	List Index	0	1	2	3	4	5
2		58	62	75	88	92	105

- Step 7: Since the searching item 62 is greater than 58, the searching item comes after 58. First = mid+1 that is First = 0+1. So, First = 1 and Last=1. Now, mid=(1+1)/2=1

- Step 8: Compare 62 with the item in position 1. That is also 62. So, the "**item is found**" and stop the searching process



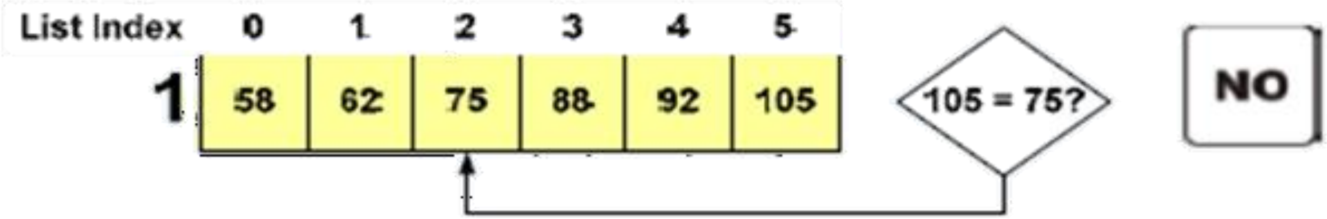
	List Index	0	1	2	3	4	5
3		58	62	75	88	92	105

Binary Search – example-2

List of Data: **58, 62, 75, 88, 92, 105**

Data to be searched is **105**

- Step 1: First = 0 and Last = 5
- Step 2: Step 2: Mid = (0 + 5) / 2. That is Mid = 2 (Only the integer part is taken)
- Step 3:

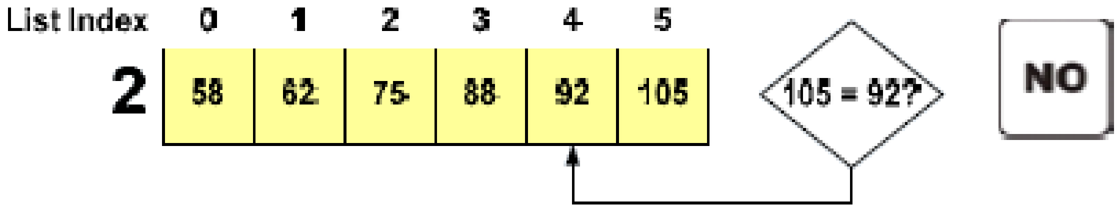


List Index	0	1	2	3	4	5
1	58	62	75	88	92	105

- Step 4: The searching item 105 is greater than 75. So it comes after 75. First = 2 +1=3. That is, First=3 and Last=5

Binary Search – example-2

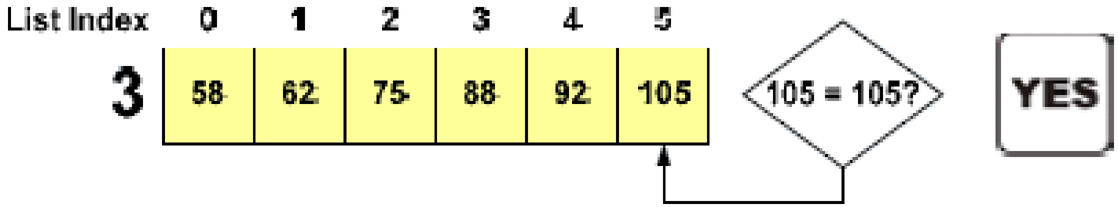
- Step 5: Compute Mid = (3+5) / 2 = 4



List Index	0	1	2	3	4	5
2	58	62	75	88	92	105

- Step 6: The searching item 105 is greater than 92. So the searching item 105 comes after 92. First= (4+1) = 5 and Last =5. So Mid=(5+5)/2 = 5

- Step 7: Compare Searching element 105 with the 5th element. Since they are equal "**Item is found**" and stop the searching process



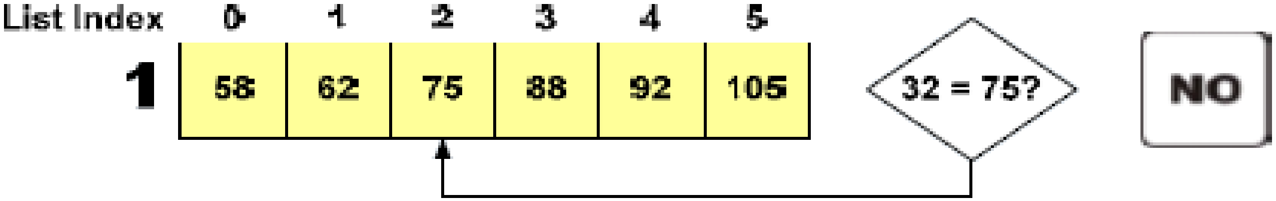
List Index	0	1	2	3	4	5
3	58	62	75	88	92	105

Binary Search – example-3

List of Data: **58, 62, 75, 88, 92, 105**

Data to be searched is **32**

- Step 1: First = 0 and Last = 5
- Step 2: Mid = (0 + 5) / 2. That is Mid = 2 (Only the integer part is taken)
- Step 3: Compare the searching item 32 and 75. Since they are not equal proceed with the next step



List Index	0	1	2	3	4	5
1	58	62	75	88	92	105

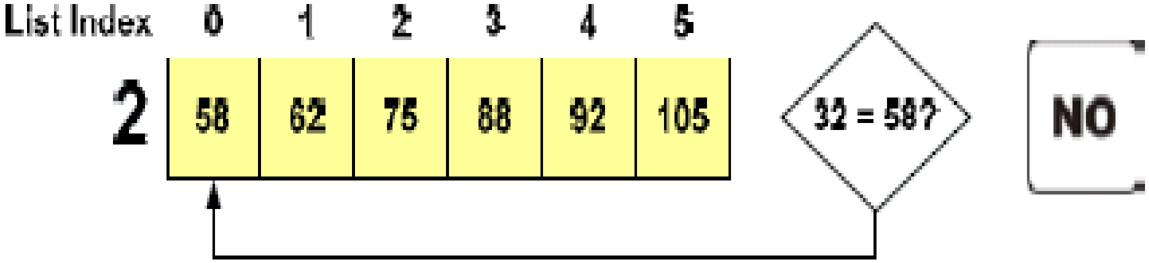
32 = 75?

NO

- Step 4: The searching item 32 is less than 75. So First=0 and Last=1. Mid=(0+1)/2=0

Binary Search – example-3

- Step 5: Compare 32 and 58. Since they are not equal proceed with the next step



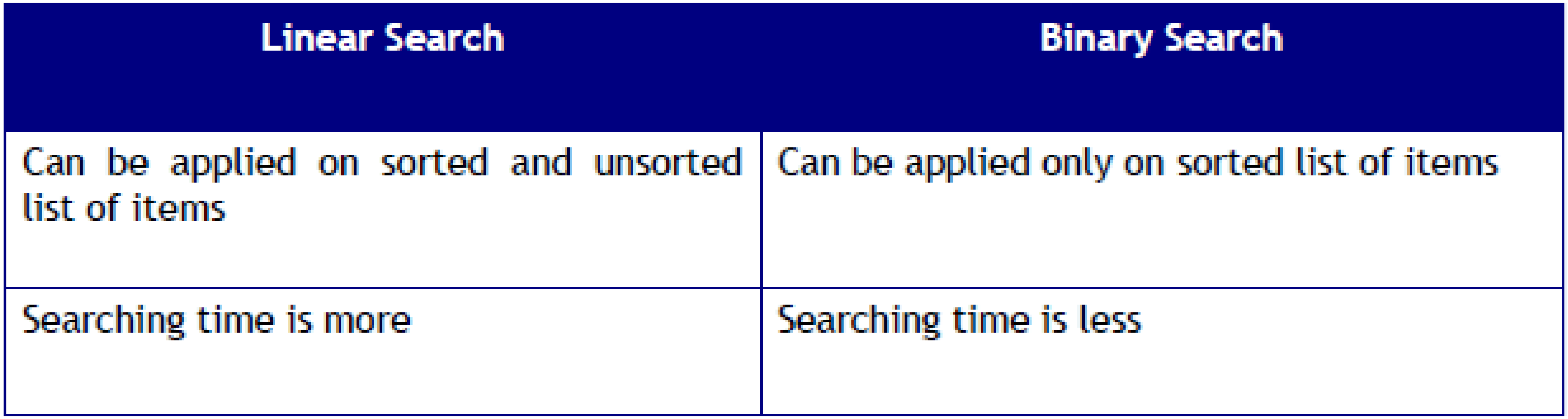
- Step 6: The searching item 32 is less than 58. So First = Mid-1 that is Last=0-1= -1 and First = 0. Since First > Last, **"Item is not found"** and stop the searching process

Binary Search – procedure

```
/* Binary search on sorted array */
low=0;
high=N-1;
do
{
mid= (low + high) / 2;
if ( key < array[mid] )
high = mid - 1;
else if ( key > array[mid])
low = mid + 1;
} while( key!=array[mid] && low <= high);
```

```
if( key == array[mid] )
{
 printf("SUCCESSFUL SEARCH\n");
}
else
{
  printf(“Search is FAILED\n");
}
```

Linear *versus* Binary Search



Linear Search	Binary Search
Can be applied on sorted and unsorted list of items	Can be applied only on sorted list of items
Searching time is more	Searching time is less

Go to posts/chat box for the link to the question

submit your solution in next 2 minutes

The session will resume in 3 minutes

Summary

- ❖Linear Search
- ❖Binary Search