COMPUTER SCIENCE

LINEAR SEARCH

Lesson Objectives

Students will learn:

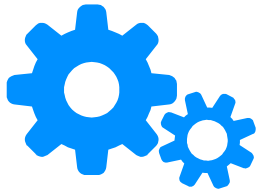
- What are searching and sorting algorithms?
- How is an element searched in a list using a linear search algorithm?
- Pseudocode for a linear search algorithm

1.

Content

Searching and sorting algorithms

- Sorting algorithms arrange the data in particular order.
- Searching algorithms are used to search for data in a list.



Linear search

1. The criteria in which the item has to be searched is set up.
2. The search begins with the first item and checks whether a match is found.
3. Then it moves to the second item, and so on.
4. Search continues until a match is found or the end of the list is reached with no match found.

Linear search algorithm: Pseudocode

Let's use an example of searching a file with name **'Project 625'** in a folder using a linear search algorithm.

Setting the variables.

INPUT user inputs 'Project 625' in File explorer

file_name='Project 625'

list_complete=FALSE

(to check until the end of list is reached)

Linear search algorithm: Pseudocode

Using while loop and if condition to compare the file names.

```
WHILE list_complete=FALSE:
    IF file_name='Project 625' then
        OUTPUT file_name, file_size, file_data and file_type
        EXIT the loop
    ELSE
        COMPARE file_name with the next file name in the list
        Output 'No match found'
    ENDIF
ENDWHILE
```

Pseudocode

- Let us consider a list with 6 elements:

B	V	D	E	T	P

- The pseudocode for a linear search algorithm for such a list is given.

```
position = 0
item=enter ("enter the item to be found")
len=number_of_elements-1
WHILE (position<len AND list[position]!= item)
        position = position + 1
ENDWHILE
IF position> len then
        print ("item not found")
ELSE
        print ("item found at position " position)
 ENDIF
```

- Let us analyse this algorithm in this list when item to be found is E.

B	V	D	E	T	P
position					

position=0, item= E and len= 5

list [position] = list [0] = B

list [position] is not equal to item. Therefore, variable position is incremented.

```
position = 0
item=enter ("enter the item to be found")
len=number_of_elements-1
while (position<len and list[position]!= item)
        position = position + 1
endwhile
if position> len then
        print ("item not found")
else
        print ("item found at position " position)
 endif
```

- Let us analyse this algorithm in this list when item to be found is E.

B	V	D	E	T	P
	position				

position=1, item= E and len= 5

list [position] = list [1] = V

list [position] is not equal to item. Therefore, variable position is incremented.

```
position = 0
item=enter ("enter the item to be found")
len=number_of_elements-1
WHILE (position<len AND list[position]!= item)
        position = position + 1
ENDWHILE
IF position> len
         THEN
        print ("item not found")
ELSE
        print ("item found at position " position)
 ENDIF
```

- Let us analyse this algorithm in this list when item to be found is E.

B	V	D	E	T	P
		position			

position=2, item= D and len= 5

list [position] = list [2] = D

list [position] is not equal to item. Therefore, variable position is incremented.

```
position = 0
item=enter ("enter the item to be found")
len=number_of_elements-1
WHILE (position<len AND list[position]!= item)
        position = position + 1
ENDWHILE
IF position> len
         THEN
        print ("item not found")
ELSE
        print ("item found at position " position)
 ENDIF
```

- Let us analyse this algorithm in this list when item to be found is E.

B	V	D	E	T	P
			position		

position=3, item= E and len= 5

list [position] = list [3] = E

list [position] is equal to item. Therefore, loop is exit and the output is: Item found at position 3.

```
position = 0
item=enter ("enter the item to be found")
len=number_of_elements-1
WHILE (position<len AND list[position]!= item)
        position = position + 1
ENDWHILE
IF position> len
         THEN
        print ("item not found")
ELSE
        print ("item found at position " position)
 ENDIF
```

2.

Activity

Activity-1

Duration: 15 minutes

1. Here is a list of 5 elements.

P	E	T	Y	Z

How is the element Y searched in the above list using a linear search algorithm? Explain each and every step.

Activity-1

Duration: 15 minutes

1. Here is a list of 5 elements.

P	E	T	Y	Z

How is the element Y searched in the above list using linear search algorithm? Explain each and every step.

Variable names:

position = 0
Item
length
number_of_elements
list[number_of_elements]

Step 1:

P	E	T	Y	Z
position				

list[position] is not equal to element. The value of position is incremented.

Step 2:

P	E	T	Y	Z
	position			

list[position] is not equal to element. The value of position is incremented.

Step 3:

P	E	T	Y	Z
		position		

list[position] is not equal to element. The value of position is incremented.

Step 4:

P	E	T	Y	Z
			position	

list[position] is equal to element. The element is found.

2. Assume that the item searched is present in the list. Is the number of times the while loop in linear search algorithm is executed related to the position of the element searched? If yes, how?

Yes. If the position is initiated to be 0, number of times while loop is executed is equal to the (position of element +1).

3.

End of topic questions

End of topic questions

1. Using the pseudocode of a linear search algorithm, create a flowchart for this algorithm.

```
position = 0
item=enter (“enter the item to be found”)
len=number_of_elements-1
WHILE (position<len and list[position]!= item)
position = position + 1
ENDWHILE
IF position> len
    THEN
    print (“item not found”)
ELSE
print (“item found at position ” position)
ENDIF
```

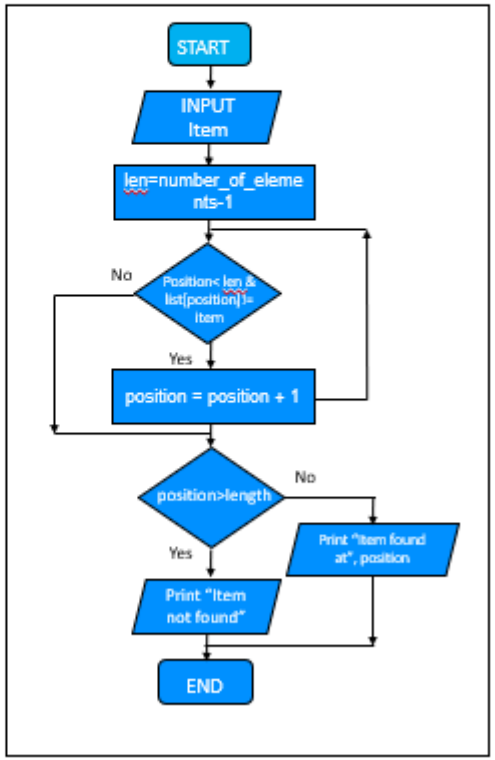
End of topic questions

2. What factors determine the number of times the while loop in the above pseudocode is executed?

```
position = 0
item=enter (“enter the item to be found”)
len=number_of_elements-1
WHILE (position<len and list[position]!= item)
position = position + 1
ENDWHILE
IF position> len
    THEN
    print (“item not found”)
ELSE
print (“item found at position ” position)
ENDIF
```

1. Using the pseudocode of linear search algorithm, create a flowchart for this algorithm.

```
position = 0
item=enter ("enter the item to
be found")
len=number_of_elements-1
while (position<len and
list[position]!= item)
position = position + 1
endwhile
if position> len then
print ("item not found")
else
print ("item found at position "
position)
endif
```



2. What factors determine the number of times this while loop in the above pseudocode is executed?

Position of elements and number of elements

Credit

- Teach Computer Science