Searching and Sorting Algorithms

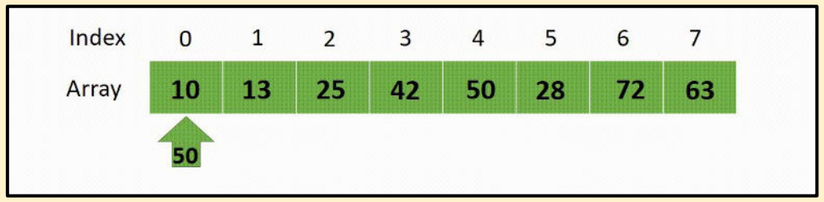
Key Words

algorithm	A set of instructions that can be used to solve a given problem
binary search	A searching algorithm used in a sorted array by repeatedly dividing the search interval in half.
bubble sort	A sorting algorithm that repeatedly passes through a list to be sorted, comparing and swapping items that are in the wrong order.
efficiency	How much time it takes to run a particular algorithm and how much space is needed.
insertion sort	A simple comparison sort, where the sorted list is built up one item at a time.
linear search	Sequentially checks each element of the list until a match is found or the whole list has been searched
merge sort	Sorts a list of data by breaking it down into multiple smaller lists, quickly sorting them, and then merging them back together into one larger list
pseudocode	A method of writing up a set of instructions for a computer program using plain English.
searching	To find a specific item in a set of data.
sorting	To put a set of data into some order.

Searching Algorithms

Linear Search

- Checks each item in the list one by one until it finds what it is looking for

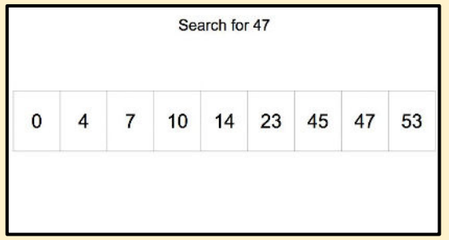


- Advantages
  - It performs well with small and medium sized lists
  - Fairly simple to code
  - The data set does not need to be in any particular order
  - It doesn't break if new items are added to the list
- Disadvantages
  - Not efficient, takes time with lots of data
- The pseudocode for a linear search is:

```
find = 2
found = Falselength = list.length
counter = 0
while found == False and counter < length
     if list[counter] == find then
          found = True
          print ('Found at position', counter)
     else:
          counter = counter + 1
     endif
endwhile
if found == False then
     print('Item not found')
endif
```

Binary Search

- Calculate a midpoint in the data set.
- Check if that is the item to be found.
- If not…
  - If the item to be found is lower than the midpoint, repeat on the left half of the data set.
  - If the item to be found is greater than the midpoint, repeat on the right half of the data set.



- Advantages
  - More efficient than a linear search
- Disadvantages
  - Only works on an ordered list, complex to program
- The pseudocode for a binary search is:

```
find = 11
found = False
length = list.length
lowerBound = 0
upperBound = length

while (lowerBound <= upperBound)
     midpoint = int((upperBound + lowerBound))/2
     if list[midPoint] == find then
          print('Found at' , midpoint)
          found = True
     endif
     if list[midpoint]> find then
          upperBound = midpoint-1
     else
          lowerBound = midpoint+1
     endif
endwhile

if found == False then
     print('Not found')
endif
```

Quizizz **Extended Reading** **Exam Questions** **Video** **Revision Techniques** **Quizlet** **Specification**

BOHUNT EDUCATION TRUST

Searching and Sorting Algorithms

Sorting Algorithms

Bubble Sort

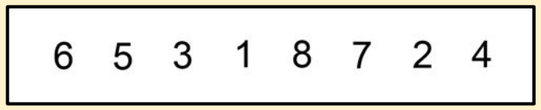
- A very simple algorithm used to sort a list of data into ascending or descending order.
- The algorithm works its way through the list, making comparisons between a pair of adjacent items. Any items found to be in the wrong order are then exchanged. It keeps doing this over and over until all items in the list are eventually sorted into the correct order.

- Advantages
  - Simple to write and understand.
  - The data is sorted in the same memory location that it is held, so you don't need much extra memory to run the algorithm.
- Disadvantages
  - One of the slowest ways to sort a list
- The pseudocode for a linear search is:

```
data_set = [9,2,5,23,34,56]
last_exam_position = data_set.length - 2
swap = true
WHILE swap == true
  swap = false
  FOR i = 0 to last_exam_position
    IF data_set[i] > data_set[i +1] THEN
      temp = data_set[i+1]
      data_set[i+1] = data_set[i]
      data_set[i] = temp
      swap = true
    END IF
  NEXT i
END WHILE
```

Insertion Sort

- Compares values in turn, starting with the second value in the list.
- If this value is greater than the value to the left of it, no changes are made.
- Otherwise this value is repeatedly moved left until it meets a value that is less than it. The sort process then starts again with the next value.
- This continues until the end of the list is reached.

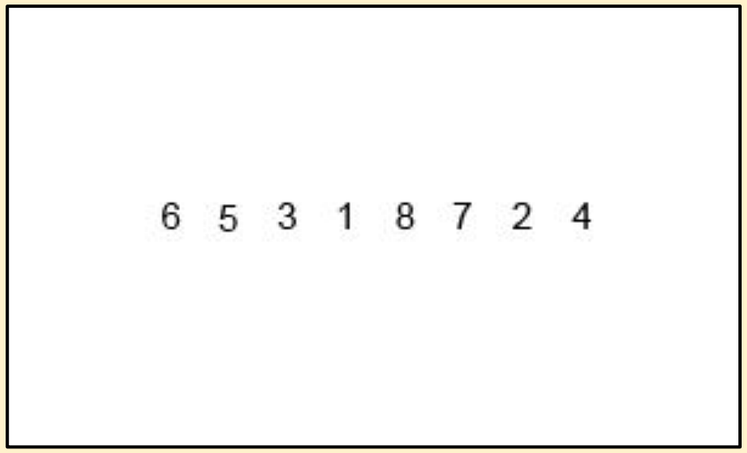


- Advantages
  - Simpler to code than a merge sort
  - Useful for small lists that only take a very short time to sort
  - Faster to process a small list than using a bubble sort
- Disadvantages
  - Not efficient with large lists - merge sort would be a better choice
- The pseudocode for a linear search is:

```
dataSet = [45,23,14,99,15]
FOR i = 1 to dataSet.length - 1
  pos = i
  WHILE pos > 0 AND dataSet[pos-1] > dataSet[pos]
    swap dataSet[pos] and dataSet[pos-1]
    pos = pos - 1
  END WHILE
NEXT i
```

Merge Sort

- Was developed to handle the sorting of large lists.
- It does this by breaking them down into multiple smaller lists, quickly sorting them, and then merging them back together into one larger list
- Merge sort is an example of a 'divide-and-conquer' algorithm because it splits down a larger problem into a number of smaller ones which are then solved.



- Advantages
  - It is the fastest of three types of sort (bubble, insert, merge)
  - It is the best option to use for long lists of data (more than 1000 long)
- Disadvantages
  - More complicated to code compared to bubble and insert
  - It may use twice the memory size of the list - depending on the way it is coded. This becomes important if the list is millions of items long.

Quizizz

Extended Reading

Exam Questions

Video

Revision Techniques

Quizlet

Specification

BOHUNT EDUCATION TRUST