

122COM: Searching

David Croft

Coventry University

david.croft@coventry.ac.uk

2018

1 Introduction

2 Linear search

3 Binary search

4 String searching

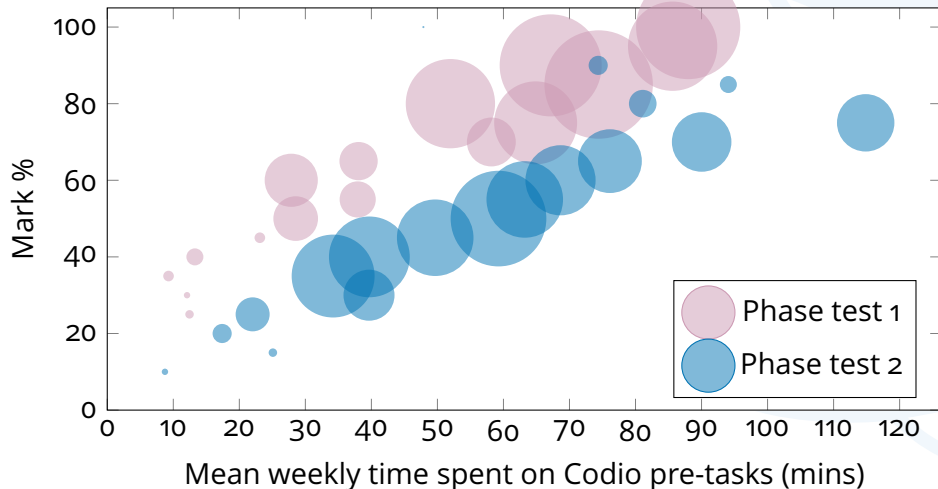
5 Quiz

6 Recap

Overview

You have all attempted the green Codio exercises for this week.

122COM results 2016-17 September starters.



Searching is used everywhere in computing.

- Obvious applications.

- Text files.
- Databases.
- File systems.
- Search engines.

- Hidden applications.

- Computer games.
 - Field Of View (FOV) search for objects in view.
 - Path finding <https://www.youtube.com/watch?v=19h1g22hby8>.
- Network routing.
- Sat Nav.
- Recommender systems.
 - Netflix What-to-watch.
 - Amazon recommended items.

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R
↑														
Z														

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
Z

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
Z

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

Simplest searching algorithm.

- Also called sequential search.
- Iterate over elements.
- Until found or until end of sequence.
- Potentially slow.
 - Worst case if the value isn't in the sequence at all.
- $O(n)$
 - Discuss $O()$ notation last week.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	Z	Q	K	L	G	H	U	A	P	L	F	N	R

↑
R

A Divide & conquer algorithm.

- Pro: Muuuuuuch faster than linear search.
- Con: Only works on sorted sequences.
- The algorithm:
 - 1 Find middle value of the sequence.
 - 2 If search value == middle value then success.
 - 3 If search value is < middle value then forget about the top half of the sequence.
 - 4 If search value is > middle value then forget about the bottom half of the sequence.
 - 5 Repeat from step 1 until `len(sequence)==0`.

Binary search II

I

Find E.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O

Binary search II

I

Find E.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O



Find E.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O

↑

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

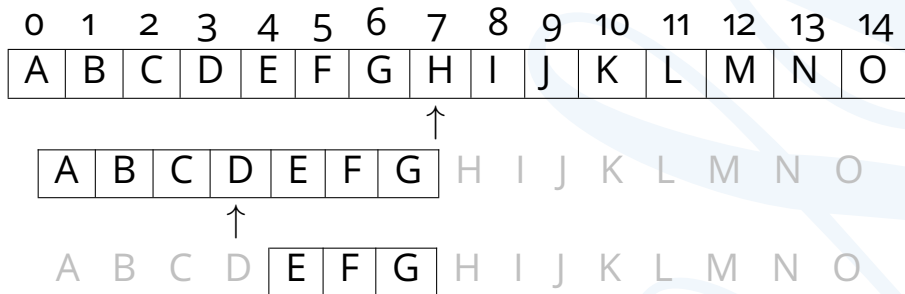
Find E.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O

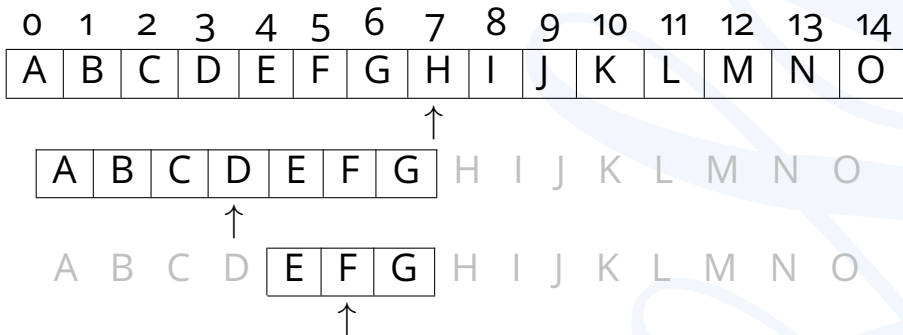
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---



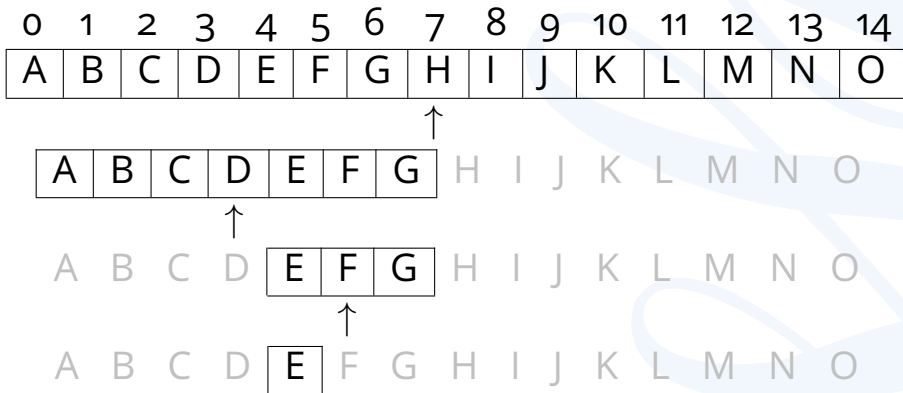
Find E.



Find E.

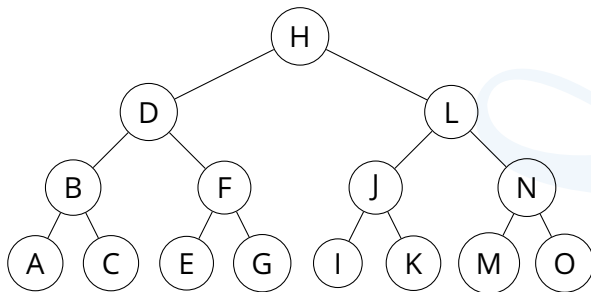


Find E.



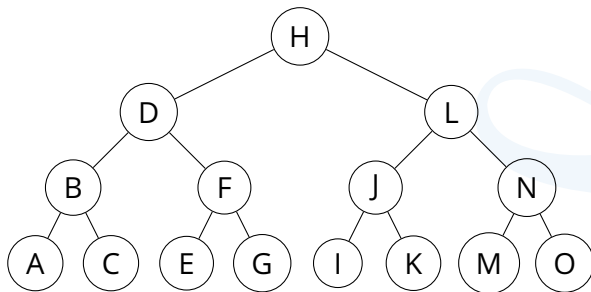
Maximum number of comparisons needed? Binary Search Trees.

- How many times can we divide our sequence in half?



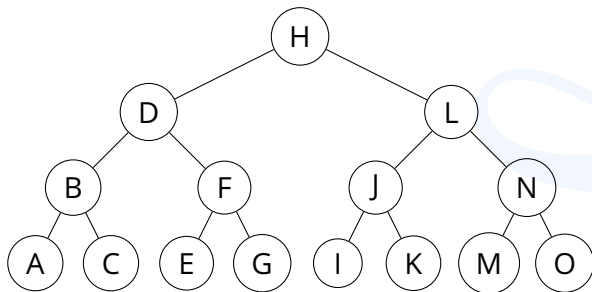
Maximum number of comparisons needed? Binary Search Trees.

- How many times can we divide our sequence in half?
- Ideal depth of the tree is $\log_2(n)$
 - $n = 15$ in this example.
 - $\log_2(15) = 3.9 \Rightarrow 3$



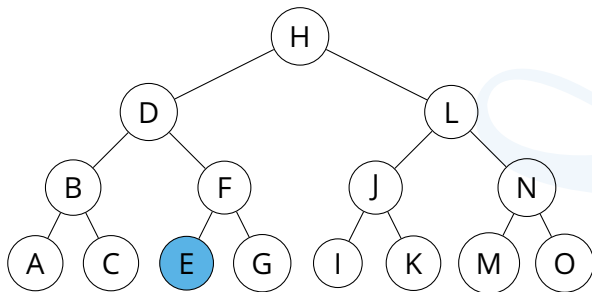
Maximum number of comparisons needed? Binary Search Trees.

- How many times can we divide our sequence in half?
- Ideal depth of the tree is $\log_2(n)$
 - $n = 15$ in this example.
 - $\log_2(15) = 3.9 \Rightarrow 3$
- Binary search has a complexity of $O(\log n)$.
 - Covered $O()$ complexity last week.



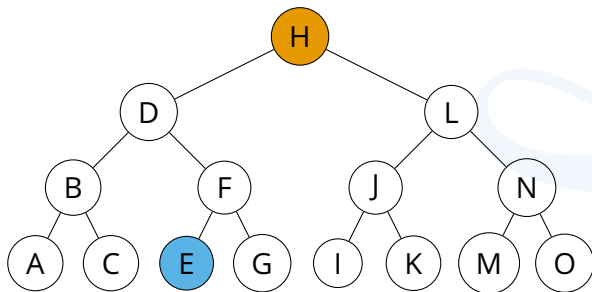
Maximum number of comparisons needed? Binary Search Trees.

- How many times can we divide our sequence in half?
- Ideal depth of the tree is $\log_2(n)$
 - $n = 15$ in this example.
 - $\log_2(15) = 3.9 \Rightarrow 3$
- Binary search has a complexity of $O(\log n)$.
 - Covered $O()$ complexity last week.
- Find E.



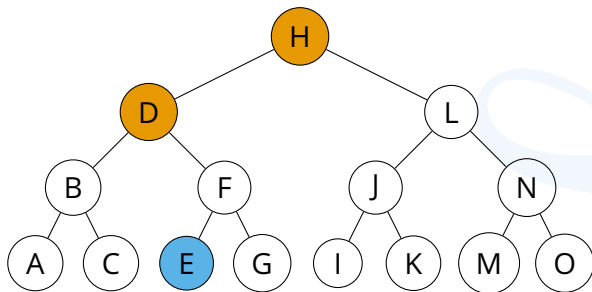
Maximum number of comparisons needed? Binary Search Trees.

- How many times can we divide our sequence in half?
- Ideal depth of the tree is $\log_2(n)$
 - $n = 15$ in this example.
 - $\log_2(15) = 3.9 \Rightarrow 3$
- Binary search has a complexity of $O(\log n)$.
 - Covered $O()$ complexity last week.
- Find E.



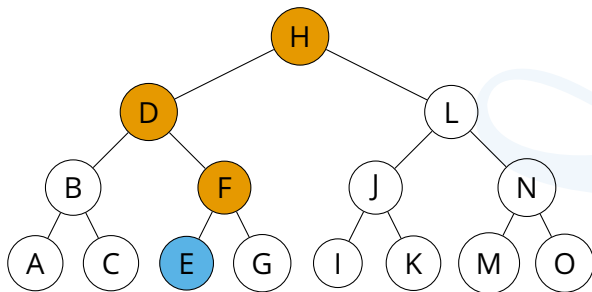
Maximum number of comparisons needed? Binary Search Trees.

- How many times can we divide our sequence in half?
- Ideal depth of the tree is $\log_2(n)$
 - $n = 15$ in this example.
 - $\log_2(15) = 3.9 \Rightarrow 3$
- Binary search has a complexity of $O(\log n)$.
 - Covered $O()$ complexity last week.
- Find E.



Maximum number of comparisons needed? Binary Search Trees.

- How many times can we divide our sequence in half?
- Ideal depth of the tree is $\log_2(n)$
 - $n = 15$ in this example.
 - $\log_2(15) = 3.9 \Rightarrow 3$
- Binary search has a complexity of $O(\log n)$.
 - Covered $O()$ complexity last week.
- Find E.



It's HOW much faster?!?!

Clearly much faster than linear search.

- To search a trillion elements linearly could mean a trillion comparisons.
- Binary search does it in 39.

But...

- Have to sort the list first.
- Sorting lists can be expensive.
 - Will cover sorting in a later week.
- Can't always sort sequences.
- Ordering can be important.
 - E.g. Words in text documents.
 - E.g. Genes in genetic chromosomes.

Break

I.e. Text searching.

- Finding one sequence in another sequence.
- Naive search.
 - Like linear search but with multiple values to compare.
 - Is very slow.

text = t h i s _ i s _ a n _ e x a m p l e
search = e x a m p l e

t h i s _ i s _ a n _ e x a m p l e
e x a m p l e

t h i s _ i s _ a n _ e x a m p l e
e x a m p l e

t h i s _ i s _ a n _ e x a m p l e
e x a m p l e

etc, etc, etc.

Quiz

By what other name is linear search known?

- 1 Divide & Conquer.
- 2 Binary search.
- 3 Sequential search.
- 4 Path finding.

By what other name is linear search known?

- 1 Divide & Conquer.
- 2 Binary search.
- 3 Sequential search.
- 4 Path finding.

What is the downside of binary search compared to linear?

- 1 Can only search sequences.
- 2 Can only search numbers.
- 3 Can only search sorted sequences.
- 4 Can only search an even number of things.

What is the downside of binary search compared to linear?

- 1 Can only search sequences.
- 2 Can only search numbers.
- 3 **Can only search sorted sequences.**
- 4 Can only search an even number of things.

Binary search is faster than linear search because ____.

- 1 No it isn't.
- 2 It only searches 1s and 0s.
- 3 It only searches two things.
- 4 It's a divide & conquer algorithm.

Binary search is faster than linear search because ____.

- 1 No it isn't.
- 2 It only searches 1s and 0s.
- 3 It only searches two things.
- 4 It's a divide & conquer algorithm.

The $O()$ complexity of binary search is ____.

- 1 $O(n)$
- 2 It depends on how many elements are being searched.
- 3 $O(\log n)$
- 4 $O(n!)$

The $O()$ complexity of binary search is ____.

- 1 $O(n)$
- 2 It depends on how many elements are being searched.
- 3 $O(\log n)$
- 4 $O(n!)$

Why do I care?

Everyone

- Searching algorithms are key to understanding many data type.
 - I.e. sets and maps/dicts.
- Key to writing efficient code.
- Key to understanding memory/processor trade offs.

Recap

- Searching
 - Applications everywhere.
- Linear search.
 - Simple.
 - Slow.
- Binary search.
 - Ordered sequence.
 - Very fast.
 - Divide & Conquer.
- String searching.
 - Finding subsequence in sequence.

- Complete the yellow Codio exercises for this week.
- Attempt the green Codio exercises for next week.
- If you have spare time attempt the red Codio exercises.
 - Will need to look at the Boyer-Moore advanced lecture slides.
- If you are having issues come to the PSC.
<https://gitlab.com/coventry-university/programming-support-lab/wikis/home>

The End