CS 2420
Fall semester section 01
Ricardo Gonzalez Mendez

Chapter 5 notes

What is algorithm:
It's a clearly specified instructions the computer will follow to solve a problem
Want to know which way is faster.

With a method to do more if you pass it a bigger number than it takes longer

1000 runs at 684658
2000 runs at 264249
We expected it to run longer because we gave it 2x as much but the second time ran faster

This is a guess
Since it is in the jvm has already loaded the information it is faster the second time

When trying to figure the times we are using the mean but some outliers throw us off so then we use the medium to see if it could improve

If you can run in N log N that is the goal that we want to go for

If you have some bad input then some will be better. Like quadratic or cubic

Linear is usually always the fastest

Moderate inputs: linear is faster, log n log next, then quadric, then cubic

Constant will be better then linear

How much better will be better

Cubic was the worst so 10 N^3 + N^2 + N

Log represents a function whose dominant term is N. (N log)

Big-Oh notation is the term used when we only discuss the dominants term

C	Constant
Log N	Logarithmic
Log^2N	Log-Squared
N	Linear
N Log N	N Log N
N^2	Quadratic
N^3	Cubic

2^N	Exponential

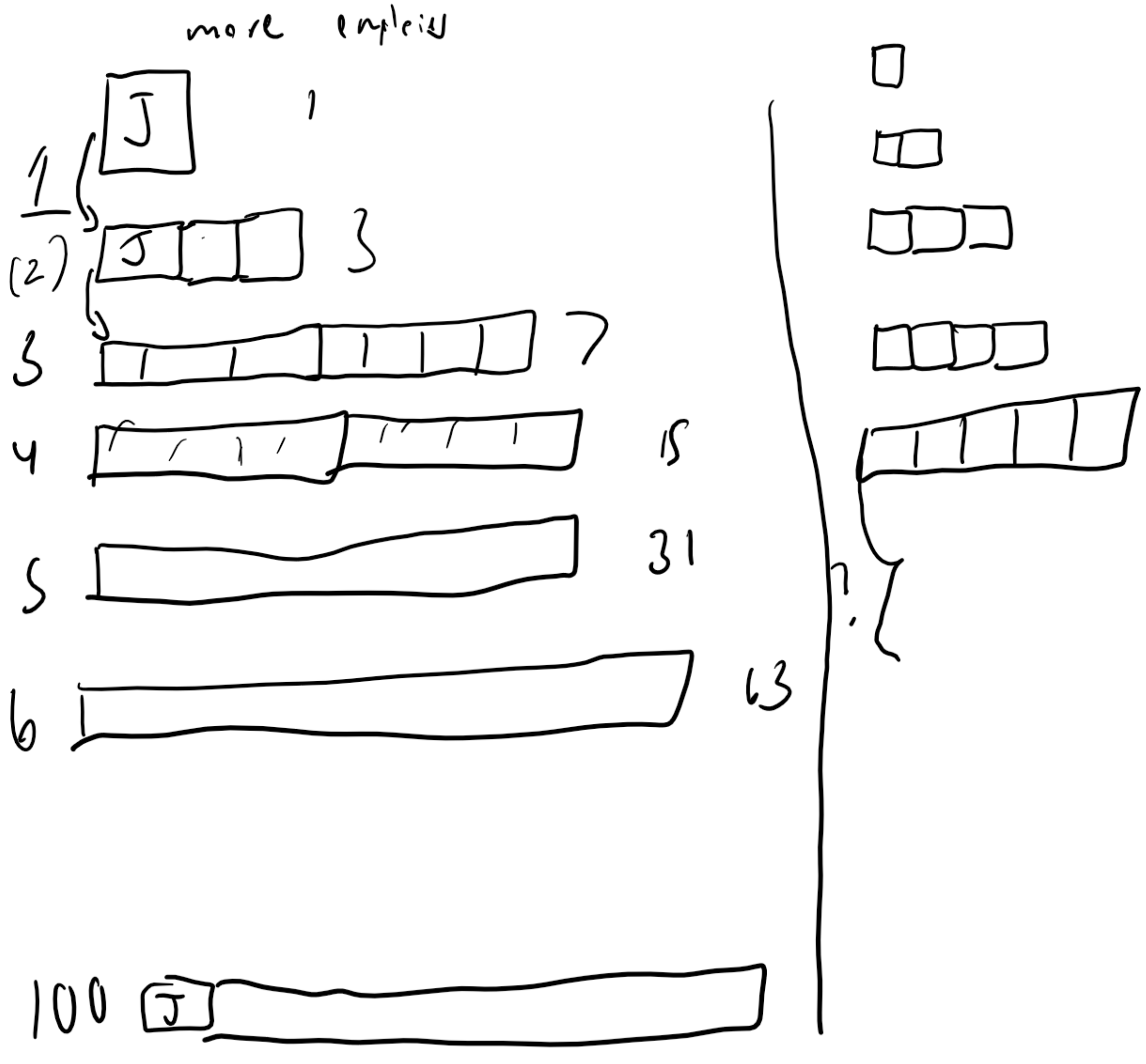
Theorem 5.1 red ball is lower then number and blue is more the highest
1<=I<=k<=j

Example: pull out red, 4, 8
Get 4, 4, 8

5.4 HW
X^2, x maybe more

- 5.23 - Let's define 'larger' to mean 'has more employees.' Further note: When a company swallows up Joe's company, they don't fire anyone, they keep all the employees, including Joe. For example, if Joe works in a company of 10 employees and that company gets swallowed up by a second company with 11 employees, the resulting company will have 10+11 = 21 employees.

Unlucky Joe Joe works for a company with n employs. Joe company always gets bought out by a company with more employs. So how many companies has he been bought by.



n?..

To do linear only get 1 for loop. We need to be more clever we will need more memory, variables, storage space. Time/space tradeoff.

Purpose is to be able to look at code and say what kind of growth is it like linear for ex.

General Big-oh riles:
Big-oh upper bound might be the same <=
Big-Omega Lower bound >= always bigger linear
Big-theta: both bounds are the same== always linear
Little-oh upper bound - will be the smaller <

5.5 the logarithm
Def for any B, N>0, logbN = k if B^K = N

$Log_b N$

When things halves or double that how we figure things are log.

Log of cuttings 1000
500
250
125
65
32
15
7
3
1
10 times

5.6 static search
Int x = 50 array of numbers return position in x or say its not their

Linear looking from left to right
5, 15, -8, 2, 4, 9, 5

Use sorting

Use sequential search
What is the cost of an unsuccessful search? Looked at al them didn't find it
What is the worst case of a successful search? Find it at last
N-1
Average case of successful search? Would 1/N

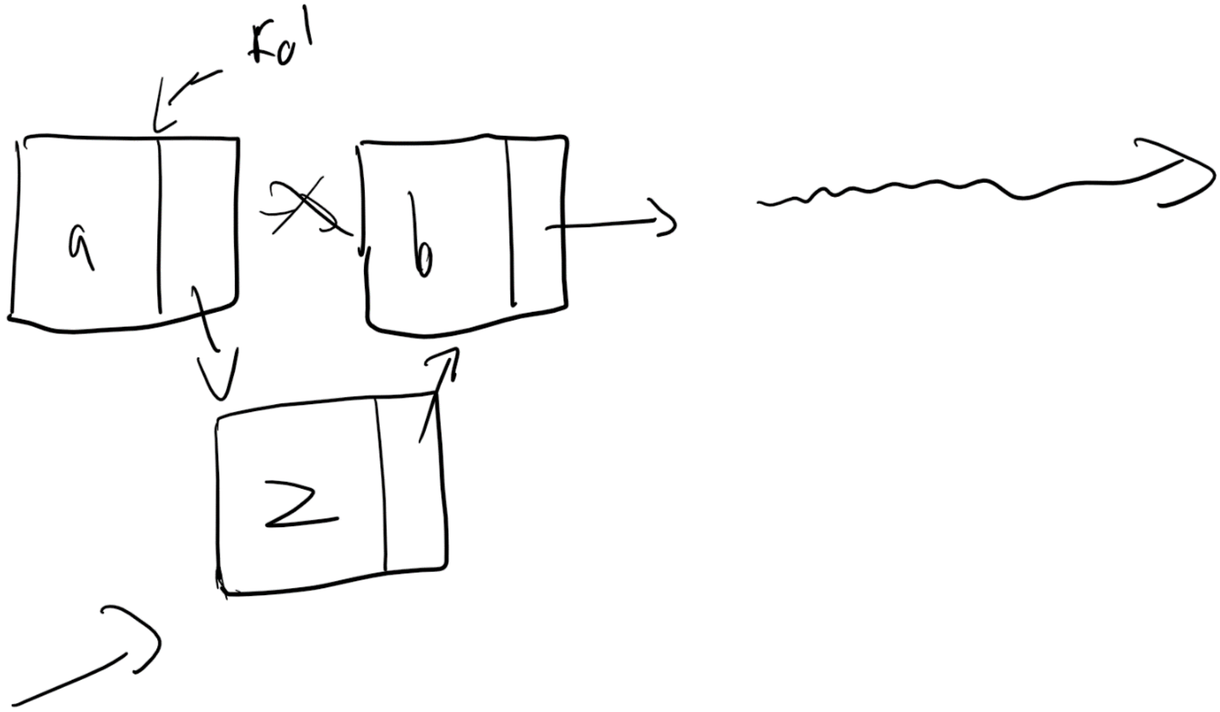
Linked list:

```
Class ListNode{
        Object element;
        ListNode next;
        }
```

Class name also name as variable? It is like recursion, java uses pointers. So this line
ListNode next;
Is just a pointer to a ListNode

To get to the end of linked list bigOh is N
For an array to get to the end bigOh is constant you go straight through it.

add link list between a & b



```
temp = New List Node ();
      temp.next = fol.next
      fol.next = temp
```

Constant BigOh if you are putting stuff at the front for link list
Array for adding to the front would be BigOh N

Where would you want to use array or link list
Keep track of people by there height.
New person average middle height would matter which one you picked

CS 2420
Fall semester section 01
Ricardo Gonzalez Mendez

If you have a group and you know the tallest person comes in first and everyone else is after you want to add shortest in first slot. Best would be link list because you know you can add to the front rather then array.