Sort Procedures and Quicker Sorting

One-Slide Summary

- **g is in O(f)** iff there exist positive constants c and n_0 such that $g(n) \leq cf(n)$ for all $n \geq n_0$.
- If g is in O(f) we say that f is an **upper bound** for g.
- We use **Omega Ω** for lower bounds and **Theta Θ** for tight bounds.
- Knowing a running time is in O(f) tells you that the running time is *not worse than f*. This can only be good news.
- We can add two numbers with electricity.

#2

Outline

- Administrivia
  - Your views, voting.
- Sorting: timing and costs
- Adding Two Numbers With Electricity
- Insertion Sort

#3

Exam 1

- Handed out at end of Monday's class on Feb 23, due at the beginning of Wednesday's class
  - You have two days, should take two hours
- Open Book – ***No DrScheme***
- Open TAs & Profs – No Friends
- Covers everything through this Wednesday including:
  - Lectures 1-11, Book Chapters 1-8, PS 1-4
- Review Sessions, Wednesday Feb 18th
  - 7:00-8:00 and 8:00-9:00 in Olsson 001

#4

Managing CS 150 Frustration

- *"I really think the Problem Set 0 gave people a false idea of how hard this course is."*

• Introduce *Automatic Adjudication* and problem set submission.

Short and Sweet: This problem set is intentionally quite short. Subsequent problem sets will cover more detailed topics, such as making mosaics, playing poker, or managing online auctions. This problem set gives you a feel for the nuts and bolts of problem set grading and will give you concrete feedback before the Arts and

- *"I am concerned that I don't understand everything."*

- Remember: drinking from a firehose! You are **not** expected to understand everything!
  - Managing expectations is a key technique to avoiding frustration.

#5

Time On Problem Sets: The Bad

- *"I believe this PS3 has taken me over 10 hours to complete, not including reading for class."*
- *"I'd say this lab took around a total of just over 3 hours for me, which is not too bad I suppose."*

- 17 (of 80) people mentioned that they found the PS long: 10 hours was the max listed time.
  - "One credit of laboratory work can equal one to four hours per week." - UVA Registrar http://www.virginia.edu/registrar/about.html
  - "a 3 hour course requires about 10 hours/week for the entire semester." - UVA Kinesiology http://records.uva.acalog.com/preview_program.php?catoid=11&poid=1052&bc=1
  - "a ratio of 4 clock hours per credit hour per week." - UVA Clinical Nursing http://www.nursing.virginia.edu/media/NEW%20Student%20Handbook%20CNL%2007-08.pdf
  - "Total contact hours for a course should account for readings, online time, outside preparation and study. Total contact hours required per credit hour are as follows: 135 hours for a 3-credit course [9 hours a week for 15 weeks]." - UVA Syllabus Template http://www.faculty.virginia.edu/bbcp/documents/Final_Syllabus_Template.doc

#6

Time On Problem Sets: The Good

- *"At least, personally, I could not have done this PS without their help. Is that really what the problem sets are supposed to be?"*

- PS3 is one of the two hardest problem sets. Remember, you are not expected to know or do it all.
  - 89% of you: perfect score on PS3, 93% on PS2. That's ***unprecedented***! You are working too hard!
- PS Design: **Open-Ended Grading**
  - Final problems allow us to distinguish between superstars: currently you are all superstars!
  - Example: Skipping 10-12 (convert-lcommands, rewrite-lcommands, fractal) on PS3: 20/25
  - **Course curve**: An "A" does not require perfect PS

Tutoring and Hints

- *"Is there any way to get one on one tutoring for this type of problem set?"*

- In the past, the ACM and ACM-W have offered one-on-one tutoring. Send me (or the course staff) email if you are interested; I will try to set something up.
  - *"More hints written into PS if possible please? This way I can work on it independently of TAs"*
- I will add more hints on a optional links for PS4 on. On your honor!

TA Time Limit?

- *"It is absolutely ridiculous that my partner and I had to wait an hour and forty minutes for help when we entered our our name on the wait list shortly after we arrived."*

- Recall that previously we voted for no time limit. We will vote again:
- I believe each TA should spend some maximum amount of time with each group (e.g., 10 minutes) before moving on. A group that still has questions after 10 minutes can add themselves to the queue again.

Writing The Code

- *"I'd rather have maybe 4 or 5 comprehensive questions where I wrote the entire snippet, because I would get more chances to work off of my own code."*

- Multiple people had this comment. Your wish is granted. Check out PS4, where there is no "fill in the blanks" code at all.
  - *"Also, 1 dropped problem set grade please!"*
- Nine people made such comments. Vote?
  - If so: drop lowest PS that is not the final project and that you got at least three points on.

Generic Comments

- "I feel the lectures are going well."
- "I really enjoy the lectures Wes gives and the TA's are amazingly helpful."
- "This class is pretty easy, but it is still enjoyable and not Java."
- "My favorite part of the lecture is the useless trivia and random activities that may partially have something to do with computer science like the music harmony and stack -- do more of that."
- "I actually utilized the TAs for the first time, and I found that they were a tremendous help and explained things fully."
- "It was good when we did a lot of examples in class that had everyone in the class try to make it out themselves first. This helped me get used to the procedures in Scheme. The problem sets have been good."
- "I like the problem sets with basic question at the beginning. They really help me to understand what we are doing in class."
- "I'm glad you made partners optional on the problem sets since I often work better alone and it gives me more freedom to work when I want to. Lecture is usually interesting, TA's are helpful. Problem sets are interesting and challenging, not so much as to be impossible though."
- "The class, although quite difficult for a noob who has not had computer science before, provides a worthwhile, interesting, and novel experience."
- "I am learning a lot in class and having fun at the same time."
- "I really have no complaints or anything about the class."
- "I like how Wes checks with us frequently to see if we're still awake (haha) and actually understand the material before moving on."
- "I enjoy how the lecture seems ot have slowed down a bit in order to provide an opportunity for more example-based learning."

Recall: Asymptotic Complexity

***g* is in *O(f)* iff:** There are positive constants c and n_0 such that $g(n) \leq cf(n)$ for all $n \geq n_0$.

***g* is in Ω(*f*) iff:** There are positive constants c and n_0 such that $g(n) \geq cf(n)$ for all $n \geq n_0$.

g* is in Θ(*f*) iff:** g is in ***O(f) and g is in **Ω(*f*)**.

Is our sort good enough?

Takes over 1 second to sort 1000-length list. How long would it take to sort 1 million items?

1s = time to sort 1000
4s ~ time to sort 2000

1M is 1000 * 1000

Sorting time is n^2
so, sorting 1000 times as many items will take 1000^2 times as long = 1 million seconds ~ 11 days

Note: there are 800 Million VISA cards in circulation.
It would take 20,000 years to process a VISA transaction at this rate.

Which of these is true?

- Our sort procedure is too slow for VISA because its running time is in $O(n^2)$
- Our sort procedure is too slow for VISA because its running time is in $\Omega(n^2)$
- Our sort procedure is too slow for VISA because its running time is in $\Theta(n^2)$

Which of these is true?

- ~~Our sort procedure is too slow for VISA because its running time is in $O(n^2)$~~
- Our sort procedure is too slow for VISA because its running time is in $\Omega(n^2)$
- Our sort procedure is too slow for VISA because its running time is in $\Theta(n^2)$

Knowing a running time is in $O(f)$ tells you the running time is not *worse* than f. This can *only* be good news. It doesn't tell you anything about how bad it is. *(Lots of people and books get this wrong.)*

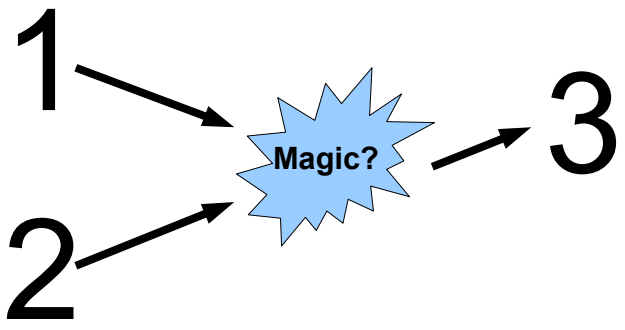
Liberal Arts Trivia: Dance

- This four wall line dance was created in 1976 by American dancer Ric Silver. It was popularized by Marcia Griffiths and remains a perennial wedding favorite. Steps: 1-4 grapevine right (tap and clap on 4), 5-8 grapevine left (tap and clap on 8), 9-12 walk back (tap and clap on 12), etc. The lyrics include "I'll teach you the ..."

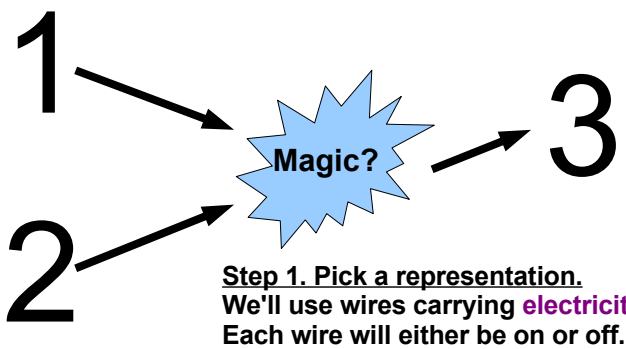
Liberal Arts Trivia: Medieval Studies

- This son of Pippin the Short was King of the Franks from 768 to his death and is known as the "father of Europe": his empire united most of Western Europe for the first time since the Romans. His rule is associated with the Carolingian Renaissance, a revival of art, religion and culture. The word for *king* in various Slavic languages (e.g., Russian, Polish, Czech) was coined after his name.

How To Add Two Numbers With Electricity

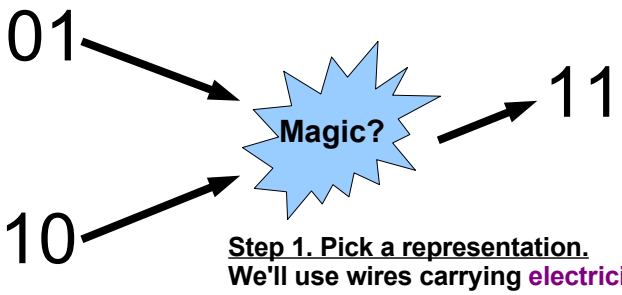


How To Add Two Numbers With Electricity



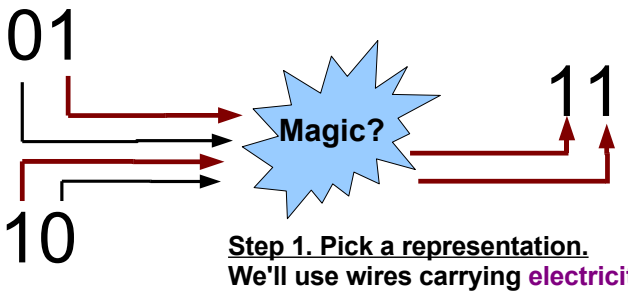
Step 1. Pick a representation.
We'll use wires carrying **electricity**.
Each wire will either be on or off.
Each wire will thus encode one **bit**.
We'll represent our numbers in **binary**.

How To Add Two Numbers With Electricity



Step 1. Pick a representation.
We'll use wires carrying **electricity**.
Each wire will either be on or off.
Each wire will thus encode one **bit**.
We'll represent our numbers in **binary**.

How To Add Two Numbers With Electricity



Step 1. Pick a representation.
We'll use wires carrying **electricity**.
Each wire will either be **on** or off.
Each wire will thus encode one **bit**.
We'll represent our numbers in **binary**.

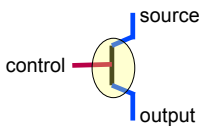
Still Adding Numbers

- What does it mean for a wire to be "on"?
  - We'll use **voltage**.
  - Ex: bit 0 is 0V to 0.8V and bit 1 is 2V to 5V
- Great. So how do I combine and manipulate voltages?
  - Example: 0+0 = 0
  - Example: 0+1 = 1
  - Example: 1+0 = 1
  - Somehow I need the output to be "on" if either of the inputs is "on". How do I do it?

Want to know more about voltage? Physics Courses!

The Transistor

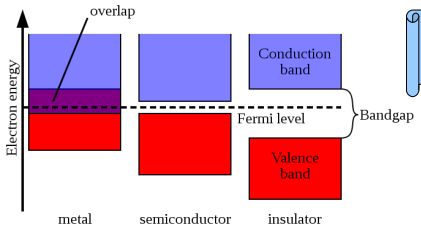
- A **transistor** is a device used to amplify or switch electronic signals.
- A transistor used as a switch has three connections to the outside world:
  - source
  - control
  - output



- If the control is "on", the source flows to the output. Otherwise, the output is "off".
  - A transistor is like a faucet.

The Transistor Continued

- A **transistor** is made of a solid piece of semiconductor material.
- A **semiconductor** is a material that has electrical conductivity that varies dynamically between that of a conductor (on) or an insulator (off). **Silicon** is a semiconductor.

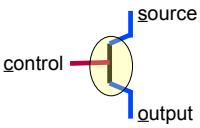


Want to know more about semiconductors? Materials Science Courses!

The Transistor

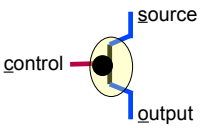
- With transistors it is possible to make two switches: normal control, and inverted control.
  - The black dot means inverted.
- Exhaustive Listing:

S	C	O
1	1	1
1	0	0
0	1	0
0	0	0



What logical operation is this?

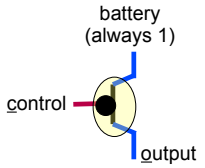
S	C	O
1	1	0
1	0	1
0	1	0
0	0	0



The Notty Transistor

- One Trick: what if we wire the source of an inverted control switch up to a battery that is always on?
- Exhaustive Listing:

C	O
1	0
0	1



What logical operation is this?

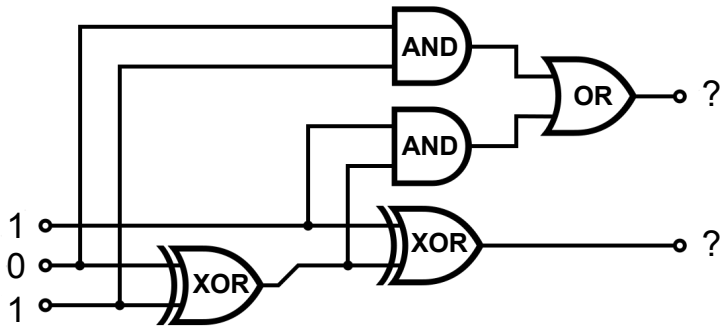
Boolean Logic

- So we have (**and** X Y) and (**not** X) for bits.
- Also (**or** X Y) = (not (and (not X) (not Y)))
- Also (**xor** X Y) = (and (or x y) (not (and x y)))
- An electronic circuit that operates on bits and implements basic boolean logic is called a **gate**.
- So far we have and, or, xor and not gates.
- That's all we need to add numbers!

Want to know more about boolean logic? Discrete Math Courses!

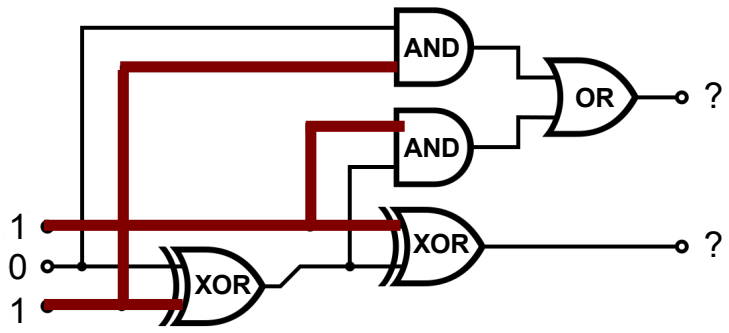
Adding Numbers!

- 1 + 0 + 1 = 10



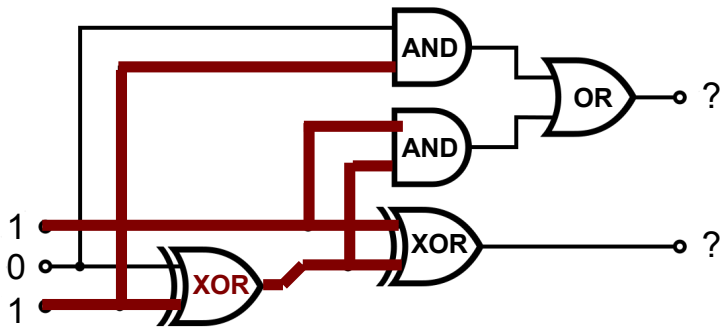
Adding Numbers!

- 1 + 0 + 1 = 10



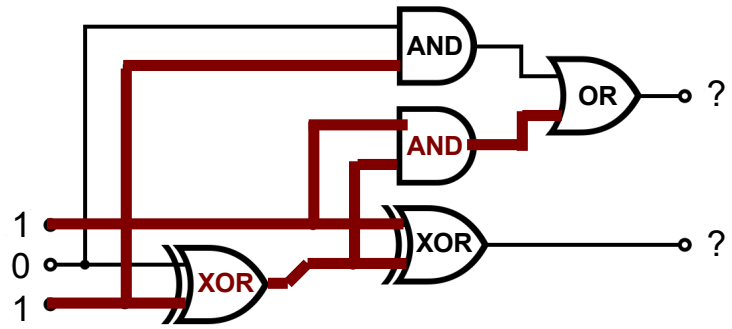
Adding Numbers!

- 1 + 0 + 1 = 10



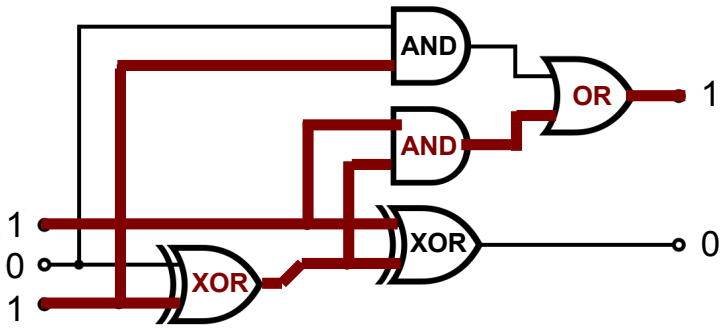
Adding Numbers!

- 1 + 0 + 1 = 10



Adding Numbers!

- 1 + 0 + 1 = 10

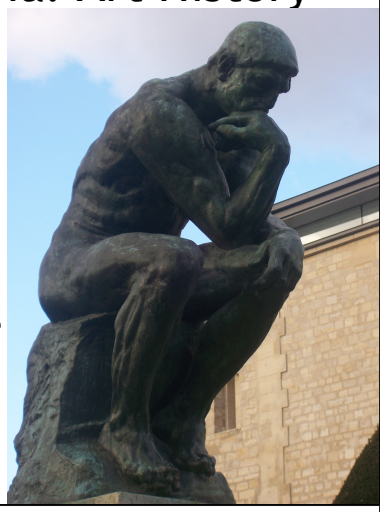


Electronic Computers

- By using *semiconductors*
  - which work using physical properties of silicon
- We can build *transistors*
  - which are like switches or faucets
- To manipulate electrical *voltages*
  - which represent bits
- Through logical *gates*
  - which encode and, or, not, etc.
- To add (and subtract, etc.) numbers!
  - In O(1) time. This is the **basis** of our cost model.

Liberal Arts Trivia: Art History

- Name the work shown and its sculptor. The artist is generally considered the progenitor of modern sculpture: he departed from mythology and allegory and modeled the human body with realism, celebrating individual character and physicality.



Liberal Arts Trivia: Chinese History

- This period of Chinese history roughly corresponds to the Eastern Zhou dynasty (8th century BCE to 5th century BCE). China was feudalistic, with Zhou kings controlling only the capital (Luoyang) and granting the rest as fiefdoms to several hundred nobles (including the Twelve Princes). As the era unfolded, powerful states annexed smaller ones until a few large principalities controlled China. By 6th century BCE, the feudal system had crumbled and the Warring States period had begun.

Sorting Cost

```
(define (best-first-sort lst cf)
  (if (null? lst) lst
      (let ((best (find-best lst cf)))
        (cons best (best-first-sort (delete lst best) cf)))))
(define (find-best lst cf)
  (if (null? (cdr lst)) (car lst)
      (pick-better cf (car lst) (find-best (cdr lst) cf))))
```

The running time of best-first-sort is in $\Theta(n^2)$ where n is the number of elements in the input list.

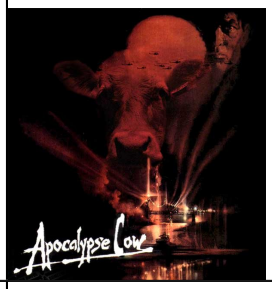
Assuming the comparison function passed as cf has constant running time.

Divide and Conquer sorting?

- **Best first sort**: find the lowest in the list, add it to the front of the result of sorting the list after deleting the lowest.
- **Insertion sort**: insert the first element of the list in the right place in the sorted rest of the list.
  - Let's write this together!

insert-sort

```
(define (insert-sort lst cf)
  (if (null? lst) null
      (insert-one (car lst)
                  (insert-sort (cdr lst) cf) cf)))
```



Try writing insert-one.

(define (insert-one element lst cf) ...)

(insert-one 2 (list 1 3 5) <) --> (1 2 3 5)

insert-one

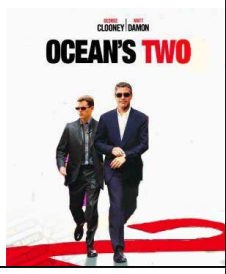
```
(define (insert-one el lst cf)
  (if (null? lst) (list el)
      (if (cf el (car lst)) (cons el lst)
          (cons (car lst)
                (insert-one el (cdr lst) cf)))))
```


How much work is insert-sort?

```
(define (insert-sort lst cf)
  (if (null? lst) null
      (insert-one (car lst) (insert-sort (cdr lst) cf) cf)))

(define (insert-one el lst cf)
  (if (null? lst) (list el)
      (if (cf el (car lst)) (cons el lst)
          (cons (car lst) (insert-one el (cdr lst) cf)))))
```

How many times does insert-sort evaluate insert-one?



How much work is insert-sort?

```
(define (insert-sort lst cf)
  (if (null? lst) null
      (insert-one (car lst) (insert-sort (cdr lst) cf) cf)))

(define (insert-one el lst cf)
  (if (null? lst) (list el)
      (if (cf el (car lst)) (cons el lst)
          (cons (car lst) (insert-one el (cdr lst) cf)))))
```

How many times does insert-sort evaluate insert-one?

n times (once for each element)

running time of insert-one is ?

How much work is insert-sort?

```
(define (insert-sort lst cf)
  (if (null? lst) null
      (insert-one (car lst) (insert-sort (cdr lst) cf) cf)))

(define (insert-one el lst cf)
  (if (null? lst) (list el)
      (if (cf el (car lst)) (cons el lst)
          (cons (car lst) (insert-one el (cdr lst) cf)))))
```

How many times does insert-sort evaluate insert-one?

n times (once for each element)

running time of insert-one is in $\Theta(n)$

How much work is insert-sort?

```
(define (insert-sort lst cf)
  (if (null? lst) null
      (insert-one (car lst) (insert-sort (cdr lst) cf) cf)))

(define (insert-one el lst cf)
  (if (null? lst) (list el)
      (if (cf el (car lst)) (cons el lst)
          (cons (car lst) (insert-one el (cdr lst) cf)))))
```

How many times does insert-sort evaluate insert-one?

n times (once for each element)

running time of insert-one is in $\Theta(n)$

insert-sort has running time in $\Theta(n^2)$ where n is the number of elements in the input list

Which is better?

- Is insert-sort faster than best-first-sort?

```
> (insert-sort < (revintsto 20))
(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20)
        Requires 190 applications of <

> (insert-sort < (intsto 20))
(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20)
        Requires 19 applications of <

> (insert-sort < (rand-int-list 20))
(0 11 16 19 23 26 31 32 32 34 42 45 53 63 64 81 82 84 84 92)
        Requires 104 applications of <
```

```
> (best-first-sort < (intsto 20))
(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20)
        Requires 210 applications of <

> (best-first-sort < (rand-int-list 20))
(4 4 16 18 19 20 23 32 36 51 53 59 67 69 73 75 82 82 88 89)
        Requires 210 applications of <
```

best-first-sort vs. insert-sort

- Both are $\Theta(n^2)$ worst case (reverse list)
- Both are $\Theta(n^2)$ when sorting a randomly ordered list
  - But insert-sort is about twice as fast
- insert-sort is $\Theta(n)$ best case (ordered input list)

Can we do better?

```
(insert-one < 88
            (list 1 2 3 5 6 23 63 77 89 90))
```

Suppose we had procedures

(first-half lst)

(second-half lst)

that quickly divided the list in two halves?

Homework

- **Problem Set 4 Due Wednesday**
- **Read Chapter 8 by Wednesday**
- Exam 1 Out Monday