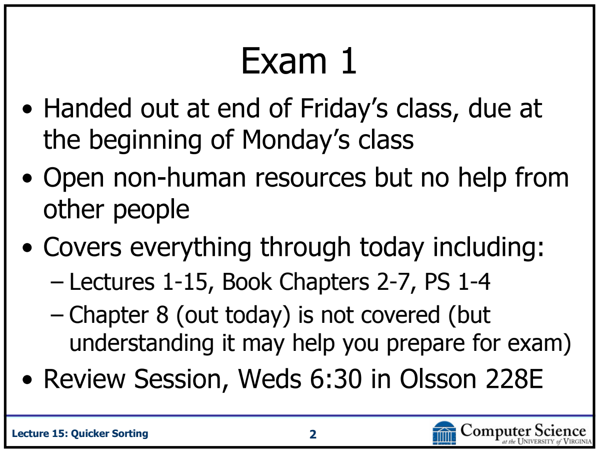
Lecture 15: Quicker Sorting

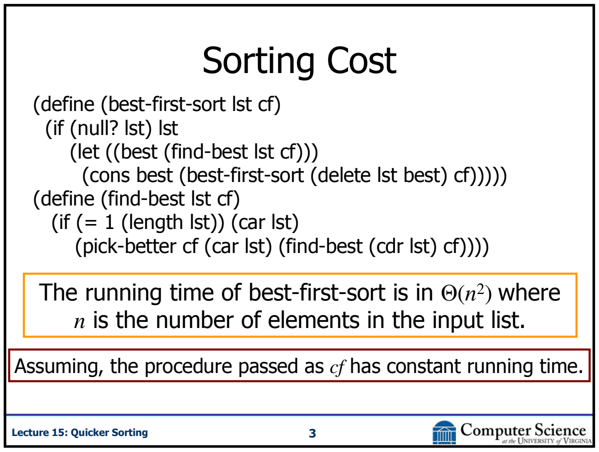
CS150: Computer Science
University of Virginia
Computer Science

David Evans
http://www.cs.virginia.edu/evans



Exam 1

- Handed out at end of Friday's class, due at the beginning of Monday's class
- Open non-human resources but no help from other people
- Covers everything through today including:
  - Lectures 1-15, Book Chapters 2-7, PS 1-4
  - Chapter 8 (out today) is not covered (but understanding it may help you prepare for exam)
- Review Session, Weds 6:30 in Olsson 228E

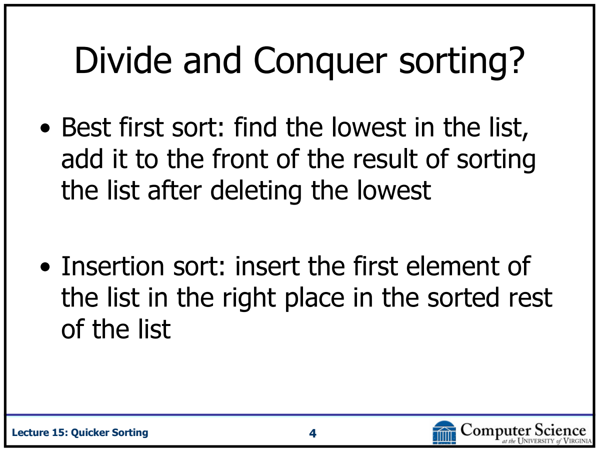


Sorting Cost

```
(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 (= 1 (length lst)) (car lst)
       (pick-better cf (car lst) (find-best (cdr lst) cf))))
```

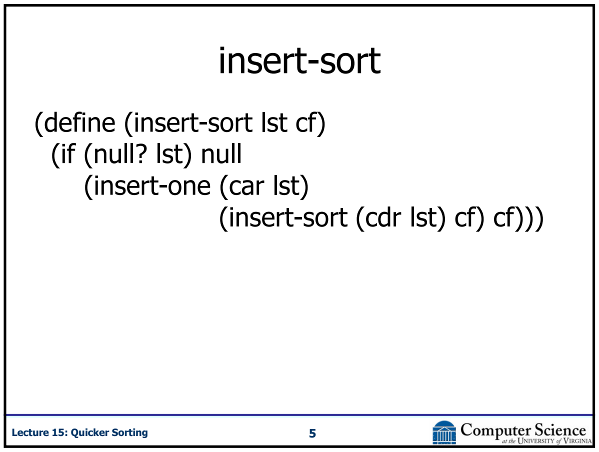
The running time of best-first-sort is in $\Theta(n^2)$ where n is the number of elements in the input list.

Assuming, the procedure passed as cf has constant running time.



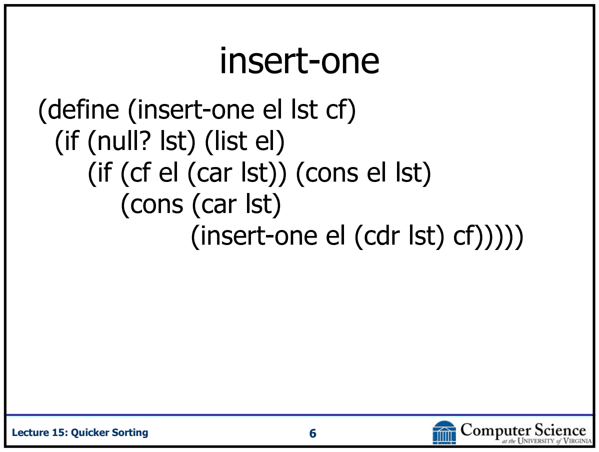
Divide and Conquer sorting?

- 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



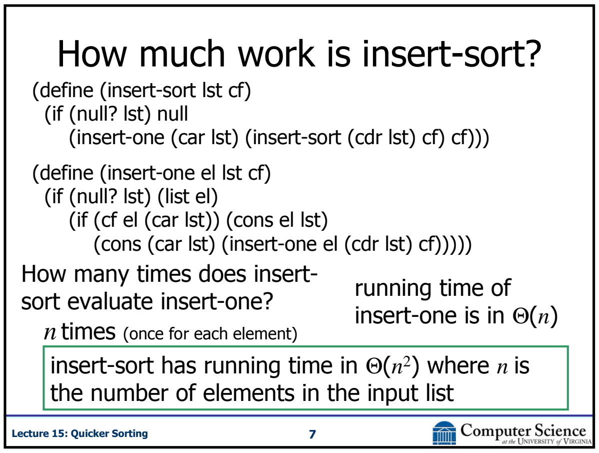
insert-sort

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



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?

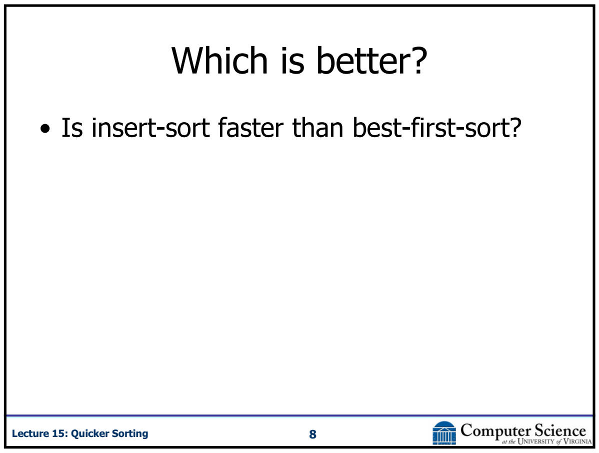
```
(define (insert-sort lst cf)
  (if (null? lst) null
      (insert-one (car lst) (insert-sort (cdr lst) cf) cf)))
(define (insert-one el lst cf)
  (if (null? lst) (list el)
      (if (cf el (car lst)) (cons el lst)
          (cons (car lst) (insert-one el (cdr lst) cf)))))
```

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

n times (once for each element)

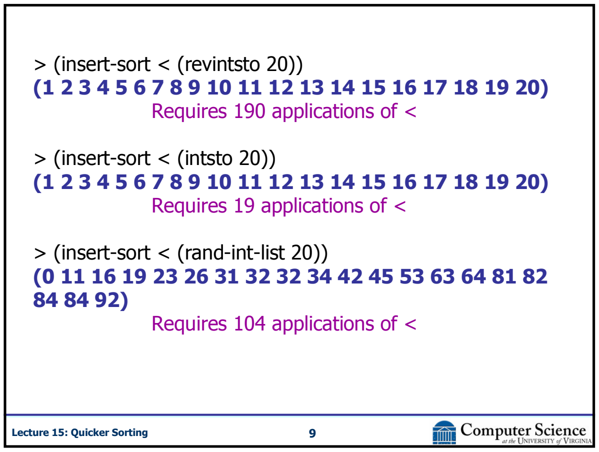
running time of insert-one is in $\Theta(n)$

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)
```

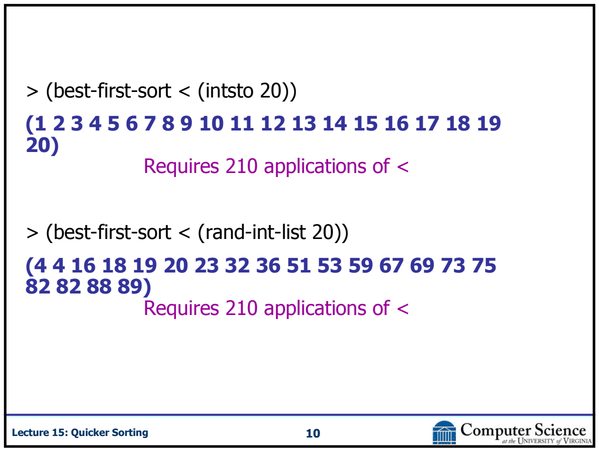
Requires 190 applications of <

```
> (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 <

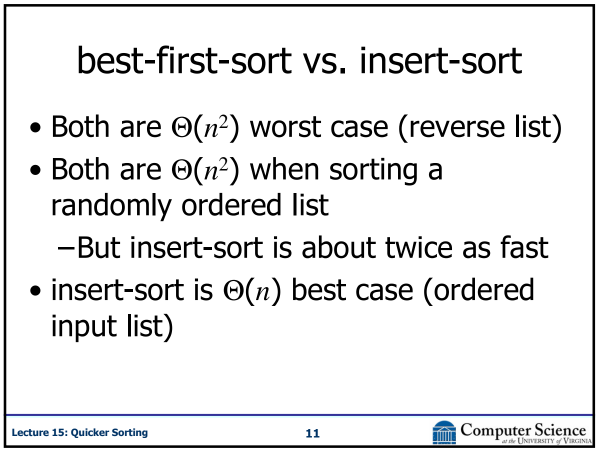


```
> (best-first-sort < (intsto 20))
(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
20)
```

Requires 210 applications of <

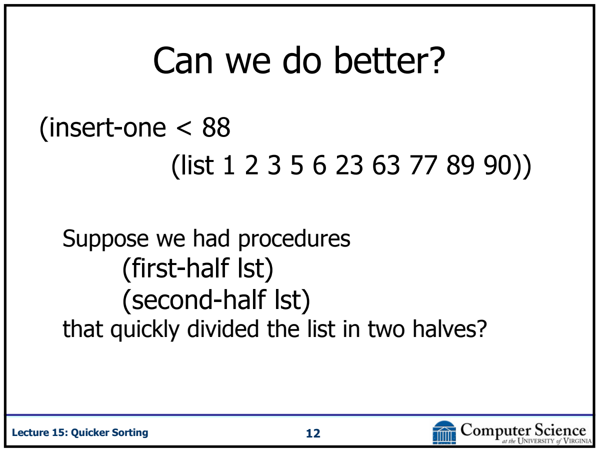
```
> (best-first-sort < (rand-int-list 20))
(4 4 16 18 19 20 23 32 36 51 53 59 67 69 73 75
82 82 88 89)
```

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?

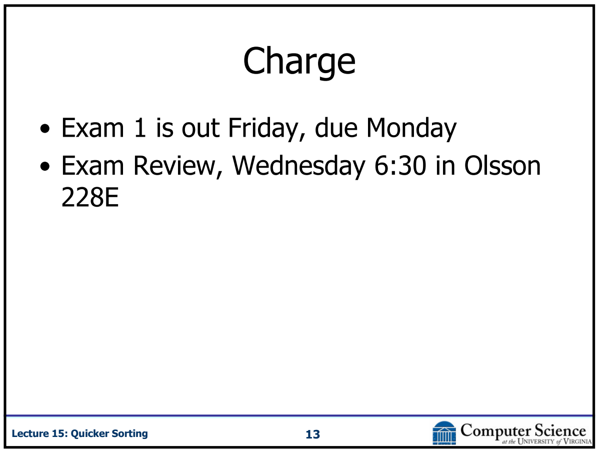
```
(insert-one < 88
            (list 1 2 3 5 6 23 63 77 89 90))
```

Suppose we had procedures

(first-half lst)

(second-half lst)

that quickly divided the list in two halves?



Charge

- Exam 1 is out Friday, due Monday
- Exam Review, Wednesday 6:30 in Olsson 228E