UNIVERSITY of WISCONSIN-MADISON
Computer Sciences Department

CS 202
Introduction to Computation

Professor Andrea Arpaci-Dusseau
Fall 2010

Lecture 36: How can computation… sort data in order for you?

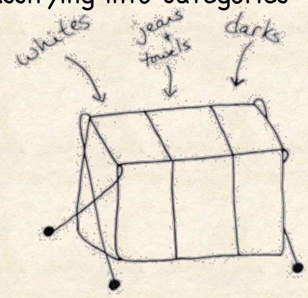
"Now remember while we're in 'sort order', which one of you is supposed to choose ASCENDING and DESCENDING?"

What is sorting?

Sorting in common language has different meaning

This is not what we mean by sorting:

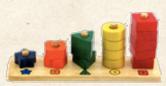
Not organizing or classifying into categories



What is sorting?

Put keys (and associated data) in specified order

- Ascending or descending
- Numerical or alphabetical



Sort keys and keep data with it

- Key: High score
  - Data: name
- Key: Employee id
  - Data: Name, Position, Phone, Salary
- Key: Search terms for web page
  - Data: URL, cached version, similar pages

Why is sorting important?

General:

Can find what you want faster given sorted data

Sorted data is easier to search through

- Can apply binary search instead of exhaustive search

Trivial to find minimum and maximum elements

- First and last in list

Easy to find duplicate values

- Adjacent to each other in list

Easy to find patterns, anomalies, gap

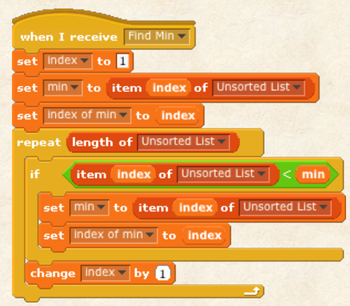
- TCP can find missing packets

Many Different Sorting Algorithms

Today: Slow algorithms, but easy to understand
- Selection sort
- Insertion sort

Next Lecture: Faster algorithms
- Mergesort
- Quicksort

Review: How to find Min in List?



Loop through List using **index** **variable**

Input:
- List : Unsorted List

Output:
- Min
- Min index

Local variable:
- index

Robust to length of List

How can you sort using Minimum?

Unsorted list

8
5
2
6
9
3
1
4
0
7

Sorted list

How can you sort list of numbers if you can find the minimum?
- Move numbers into "sorted list"

Algorithm 1: Selection Sort

Unsorted list

8
5
2
6
9
3
1
4
0
7

Sorted list

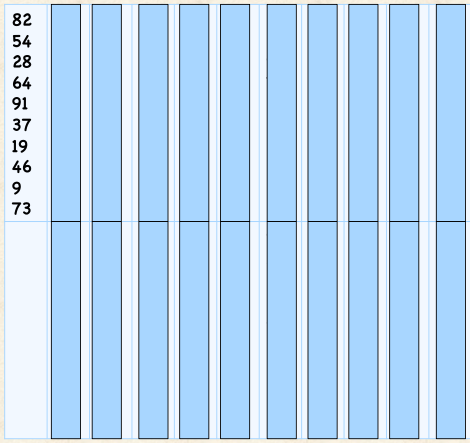
To sort data…

Repeat until nothing in unsorted list:
- Find minimum element
- Add element to sorted list
- Delete from unsorted list

Selection Sort in Action

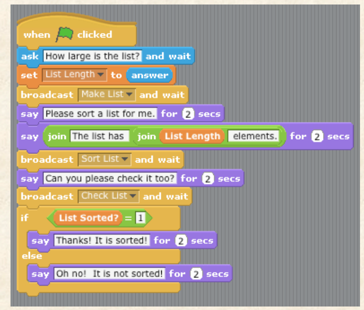
Unsorted list

Sorted list



How to Implement Selection Sort in Scratch?

Control code | **Asks Sorter Sprite to do work**



Create unsorted list

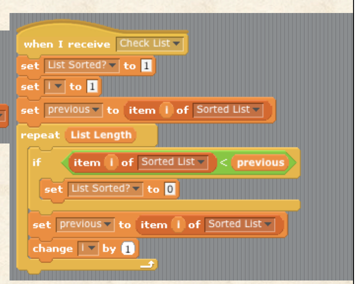
Get the list sorted

Check that the list really is sorted

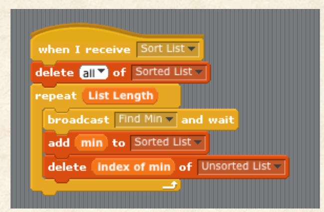
Sorter Sprite: Helper Functions

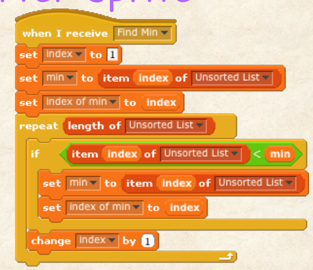
Make List

Check List



Selection Sorter Sprite





Selection Sort: Two Lists

Finds minimum remaining element in unsorted

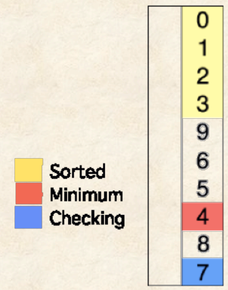
Adds to Sorted (in order)

Deletes minimum from unsorted

Selection Sort: Do we need two Lists?

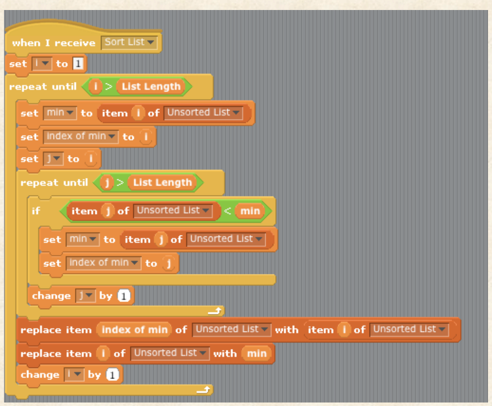
Unsorted list	8 5 2 6 9 3 1 4 0 7	8 5 2 6 9 3 1 4 7	8 5 2 6 9 3 4 7	8 5 6 9 3 4 7	8 5 6 9 4 7	8 5 6 9 7	8 6 9 7	8 9 7	8 9	9	
Sorted list		0	0 1	0 1 2	0 1 2 3	0 1 2 3 4	0 1 2 3 4 5	0 1 2 3 4 5 6	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7 8	0 1 2 3 4 5 6 7 8 9

Selection Sort: One List Demo



Swaps minimum with key at desired location

Selection Sort in Scratch: One List



Variable i:
Number of sorted elements

Variable j:
Looks for min of remaining unsorted elements (start at variable "I" in each iteration)

Algorithm 2: Insertion Sort

What algorithm do you use to sort cards?

Insertion Sort

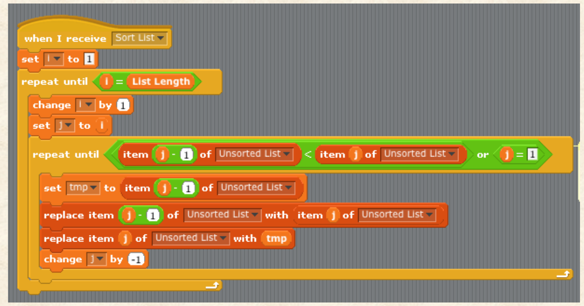
Divide cards into two groups: sorted and unsorted

Initial state: 1 sorted card, N-1 unsorted

Repeat for all cards

- Remove 1st card from unsorted portion
- Insert into correct location in sorted list
  - Repeat loop
  - Keep moving down list until card to left < new card
    - Could change algorithm to start at bottom and move up…
- Change definition of sorted vs. unsorted portions

Insertion Sort in Scratch



Repeat for all cards
  Take 1st unsorted card
  Insert into correct location in sorted list: Repeat loop
    Keep moving down list until card to left is smaller than new card
      OR at beginning of list

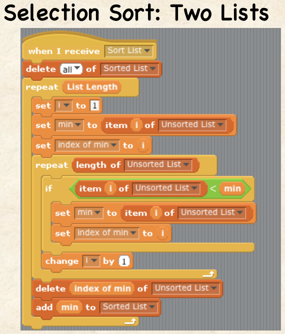
Which Sorting Algorithm is Best?

Compare number of loop iterations as function of N – size of input list

Previously analyzed searching algorithms
  Linear search: O(N) operations
  Binary search: $O(\log_2 N)$ operations

Selection Sort: How many loop iterations?

Selection Sort: Two Lists



Size of list: N

2 loops: Outer and inner

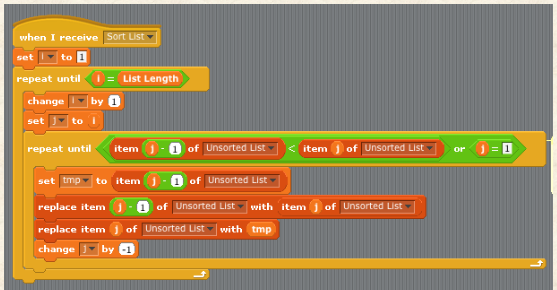
How many iterations of outer loop?
- N

How many iterations of inner loop?
- N, N-1, N-2, … 1
- Average: N/2

Total?
- N * N/2

Complexity?
- $O(N^2)$

Insertion Sort in Scratch



Outer loop?
    Always N
Inner loop – Worst case? Data in reverse order! Must move key to beginning
    1, 2, ….. N-3, N-2, N- → N/2
Best case? Data sorted already! Done immediately!
    0
Average case? Move to middle of list… ½ * Worst case
    $O(N^2)$

Today's Summary

Intuitive but Slow Sorting

- Selection sort: Select minimum and make next in list
- Insertion sort: Take next and insert in correct place
- Both require operations $O(N^2)$
- Tip: Always write check code (easier than work code)

Reading

- Section 3.3.3

Announcements

- Exam 2 returned (Median 86, Ave 80)
- Project 2: Trivia Lists
- Click "Love It" for Project 1 samples