X Y Z is for …

- **XOR**
  - $(A \ || \ B) \ && \ !(A \ && \ B)$ aka $A^B$

- **XML**
  - Extensible Markup Language
  - Yesterday's JSON

- **Y-Combinator**
  - [https://www.ycombinator.com/](https://www.ycombinator.com/)

- **YouTube**
  - Scale made/makes it work

- **Zero**
  - There are two bits in the universe, or 10

- **Zip**
  - Magic number is 0x4b50, “PK”

---

**Announcements**

- **APT-8 due Tuesday, April 21**
- **Assignment P6 Huffman due April 22**
  - All late work turned in by April 22 (APTs and Asgns)
  - Except Huffman grace through April 23
- **Assignment P7 Optional out – Extra Credit!**
  - Can turn in through Sunday night, April 26
- **Final Exam will be on April 30 – any time on this day**
- **Fill out course evaluation by Saturday 4/25**
  - 75% - 1 extra point added to assignment total
  - 85% - 2 extra points added to assignment total

---

**PfLDOC**

- **Where do you go from here in CompSci?**
- **More on trees – general tree, balanced tree**
- **What can we do with computers**
  - So so many things, power of scale
  - Data, computers, and storage: oh my!
- **What can't we do with computers?**
  - Today or every day now and forever?
  - Chess and go: we'll never compete, but now?
- **Final Exam: details**
Beyond CompSci 201

More on trees

- General tree

General Tree Node

```java
public class GenTreeNode {
    int info;
    TreeNode leftChild;
    TreeNode sibling;

    GenTreeNode(int x, TreeNode lchild, TreeNode sib) {
        info = x;
        leftChild = lchild;
        sibling = sib;
    }
}
```
What else can you do with Trees?

- I invented a new tree data structure back in the day....

Example: Convex Hull problems

Problem: Dynamic Maintenance of Maximal Points in a Plane

- Points in the x-y plane
- We will calculate which points are maximal
- As the points come and go, we want to be able to quickly list out the maximal points (dynamically)

Graphics leads to careers in Animation, computer vision
Definition of Maximal Points in a Plane

A point \( p_i = (x_i, y_i) \) in the x-y plane is maximal

if there is no point \( p_j = (x_j, y_j) \)

such that \( x_j > x_i \) and \( y_j > y_i \)

Maximal Points – form a staircase or m-contour

Points in the (x,y) plane, which are maximal points?

Deleting point 9
Presenting the Dynamic Contour Search Tree

- Here is the data structure I invented to solve this problem

1

4
5

3
6

2

8

Insertions: $O(\log n)$
Deletions: $O((\log n)^2)$
List m maximal points: $O(m)$

Balanced Trees

- Splay trees
- AVL trees
- Red-black trees
- B-trees

Red-Black Tree

- Invented by Bayr (1972) – (though called them something else)
- Robert Tarjan (Turing Award Winner) – noticed the rotations were $O(1)$
- Type of balanced tree – uses color scheme, recoloring and rotations to balance
Red-Black Tree

• Is a Binary Search Tree
• Properties:
  – Every node is red or black
  – The root is black
  – If a node is red, then its children are black
  – Every leaf is a null node and black (external node)
  – Every simple path from a node to a descendant leaf contains the same number of black nodes.

Example red-black tree

• In the figure, black nodes are shaded and red nodes are non-shaded
• Check properties

Example

• The five properties ensure that no path is more than twice as long as any other path
• Def. The height \( h \) of a node is the length of the longest path from the node (downward) to a leaf (including external nodes).
• Def. The black height \( bh \) of a node \( x \) is the number of black nodes on any path from \( x \) (not including \( x \)) to a leaf

Example red-black tree

• In the figure, black nodes are shaded and red nodes are non-shaded

\[ h(19): \]
\[ bh(19): \]
\[ h(8) \]
\[ bh(8): \]
Example red-black tree

- In the figure, black nodes are shaded and red nodes are non-shaded

h(19): 3
bh(19): 2
h(8): 2
bh(8): 1

Height of Red-Black Tree

- Lemma: A red-black tree with \( n \) internal nodes has height at most \( 2 \log (n + 1) \)

- Operations:
  - Time for search for \( x \): \( O(\log n) \)
  - Time for min: \( O(\log n) \)
  - Time for list inorder: \( O(n) \)

Rotations

- We want to perform insertions and deletions in \( O(\log n) \) time. Adding or deleting a node may disrupt one of its properties, so in addition to some recolorings, we may also have to restructure the tree by performing a rotation (change some pointers).

  ```plaintext
  left_rotate x
  ```

- Note the inorder traversal in both is: abcde
Right Rotate

- Note the rotations change the pointer structure while preserving the inorder property.

```
right_rotate x
```

Example of rotation

Insertion

- Insert node as RED using a binary search tree `insert`.
  - Means insert as a Red leaf with two black NULL nodes
- Then fix-up so that properties still hold
  - Recoloring and/or 1-2 rotations
- Several cases to consider

Cases for Insert
Insertion – Case 1 – How to Fix

- Case 1 – sibling of parent of x (called y) is red

```
A   D
  y
```

- To fix: recolor three nodes, then fix up new “x”

Insertion – Case 2 How to Fix

- Sibling of parent of x (call y) is black, x right child

```
A   D
  y
```

- To fix: set x to parent of x and left rotate x, then it becomes a case 3

Insertion – Case 3 – How to Fix

- Case 3 – sibling of parent of x (call y) is black, x left child

```
B   D
  y
```

- To fix: two recolorings and one right rotate of grandparent of x

Example of Insert 4 w/ double rotation

```
11  14  15
  7   8   y
```

- case 1

```
11  14  15
  7   8  y
```

- case 2

```
11  14  15
  7   8  y
```

- case 3
Analysis – Red Black Tree

- Insert
- Deletion

Analysis – Red Black Tree

- Insert \(O(\log n)\)
- Deletion \(O(\log n)\)

WOTO


Admiral Grace Hopper

- One of the first programmers
  - Harvard Mark 1
- PhD Math at Yale
- Admiral in the Navy
- On Letterman show
- https://youtu.be/IGTEUtS5H7I
- Gave out nanoseconds
  - Wire 11.8 inches long

*It is often easier to ask for forgiveness than to ask for permission.*
What can computers do?

Susan Rodger
April 22, 2020

WhatsApp

Traditional Paired Exchange

Donor Chain

Cluster #1

Cluster #2
What can you do?

- Not everyone wants to be a software engineer
  - Diplomat, lawyer, physician, entrepreneur,
  - Musician, teacher, data scientist, …

- Problems with programs, problems with people
  - UI, UX, PM, SWE

- What you know and what you know how to do
  - More of the latter, perhaps

I guess I'm too dumb

Some Better Scenarios

I can't write this program because it's provably impossible

Some Better Scenarios

I can't write this program but neither can all these famous people
Solving Problems

- Some problems cannot be solved at all
  - One program detects all infinite loops
- Some problems cannot be solved efficiently
  - Listing all N-bit sequences of 0's and 1's
- Some problems have approximate solutions
  - Siri: not exact, but close or good enough

What's a Hard Problem?

- Efficient solutions? Some yes, some no, some …
  - We don't know, but if we found one?
  - We'd solve many, many unknown ones
- Clay Prize: does P == NP?
  - Efficient solutions versus guess and check
  - Theoretical aspects of computer science

Let's look at Math now

- How many Natural numbers are there? 1, 2, 3, …
  - What about Integers, …, -2, -1, 0, 1, 2, …
  - Both infinity, but the same?
- What about rational numbers? ½, ¾, and so on?
- What about real numbers: sqrt(2), pi
- There are degrees of infinity
  - More reals than the others which are the same
  - If you enjoy degrees of infinity, …

Good and Bad Websites

- How do we identify these?
  - Go to goodwebsites.com
  - Go to badwebsites.com
- Are both listed on goodwebsites.com?
Contradiction

• What about this website?
  • sites_that_do_not_list_themselves.com

• Does sites_that_do_not_list_themselves.com list itself?
  • If so, it doesn't
  • But, if it doesn't, it does
  • Oh no. We cannot have this website, no way.

Infinite Loop Testing

• Can we write doesHalt as specified? Suppose so!
  • Reads a program and an input
    • Like the Java Compiler: reads a program

```java
public class ProgramUtils {
    /**
     * Returns true if programe halts on input,
     * otherwise returns false (infinite loop)
     */
    public static boolean doesHalt(String progname, String input){
    }
}
```
Self-reference, ... please, no

- What about Confuse.java?
  - Specifies program and input, calls `doesHalt`

```java
public class Confuse {
    public static void main(String[] args){
        String prog = "Confuse.java";
        if (ProgramUtils.doesHalt(prog,prog)) {
            while (true) {
                // do nothing forever
            }
        }
    }
}
```

Two alternatives, none work

- What if `doesHalt` returns true? Meaning stops!
  - If Confuse/confuse stops, then infinite loop, oh no!
- What if `doesHalt` returns false?
  - What happens in main? Program ends. But ...

This was (or is) earth-shattering

- Alan Turing first showed this for programs: 1936
  - Had to formally specify what a program was
  - Also demonstrated by Alonzo Church
- Cantor showed Real Numbers > Rationals
  - So-called diagonalization, 1891
  - Ridiculed by establishment

Dijkstra's Shortest Path Algorithm

- Similar to BFS, Use PQ not Queue
  - Only works with positive edge-weights
  - Starting at one vertex, S, find shortest paths to every other vertex.
  - Work to find path, not just length of path
    - Add a map!
- Reminder: No efficient way to find longest path!
In computational complexity, problems that are in the complexity class \( P \) but are neither in the class \( P \) nor \( NP \)-complete are called \( NP \)-intermediate, and the class of such problems is called \( NPI \). Ladner’s theorem, shown in 1975 by Richard E. Ladner, is a result asserting that, if \( P \neq NP \), then \( NPI \) is not empty; that is, \( NP \) contains problems that are neither in \( P \) nor \( NP \)-complete. Since the other direction is trivial, it follows that \( P = NP \) if and only if \( NPI \) is empty.

Exam 2

- 93 points total
- Median 76.5
- Mean 73.55
- Std Dev 13.59
- Request regrades through Sunday, April 26
Think you did vs How you did

Was Exam 2 Fair?

How was it taking Exam2 on Gradescope (more difficult -> easier)

I read your comments

Computer Problems

YOU KNOW THIS METAL RECTANGLE FULL OF LITTLE LIGHTS?

YEAH.

I SPEND MOST OF MY LIFE PRESSING BUTTONS TO MAKE THE PATTERN OF LIGHTS CHANGE HOWEVER I WANT.

SOUNDS GOOD.

BUT TODAY, THE PATTERN OF LIGHTS IS ALL WRONG! OH GOD! TRY PRESSING MORE BUTTONS!

IT'S NOT HELPING!
Final Exam

- Exam is Thursday, April 30
  - You take it any time during this 24 hour period
  - Once you start, you have 3 hours plus one additional hour because you are taking it online – so you have 4 hours total.
- Format will be similar to Exam 2 format
- Encourage you to take the exam and give it your best shot
  - Great practice for later CompSci courses
  - Solidifies knowledge in CompSci 201

More Final Exam Details - GradeScope

- MC, short answer, and short code segments.
- You will type in, or click on answers
- Suggest: write code in simple text file, and then copy paste it
- Gradescope saves each answer as you go.
  - Lose internet, just connect back in
- Exam is about 2 hours long, but you have 3 hours
- Plus you get one extra hour for logistics
- Total time you get: 4 hours
- Those with accommodations, Kate will email

Final Exam – Honor Code

- The exam is your work only
- Use books, open notes, code you have written
- DO NOT write code and run it in IntelliJ, Jshell or other computer means
- DO NOT Search on the web for answers to problems
- DO NOT Talk to any humans about the exam during the exam period
- DO NOT Talk to any humans about the exam after you take the exam or before you take the exam.

How to Study

- Covers all topics we have covered
- How to study
  - Go over lecture notes
    - Consider 1 MC question from each lecture
  - Programming assignments, APTs
  - Discussion
  - Practice writing code
    - Do a problem again
    - Write code on paper or type in text editor
    - Do APTs you haven’t done yet. OR Redo old ones
HashSet<String> hs and TreeSet<String> ts each contain the same N strings. Which of the following best characterizes the runtimes of printing all N values from each set in alphabetical order?

A. For ts this is O(N) for hs this is O(N)
B. For ts this is O(N) for hs this is O(N log N)
C. For ts this is O(N log N) for hs this is O(N)
D. For ts this is O(N log N) for hs this is O(N log N)

Learning Objectives

- Design, develop, debug, test Java program using standard libraries to efficiently solve problems with real-world components
- Write programs that effectively implement and use arrays, maps, linked lists, stacks, queues, trees
- Evaluate time and space complexities of algorithms, especially those that scale
- Apply basic object-oriented design and programming principles in developing software

Who to Thank...

201 UTAs, TAs and Teaching Assoc Kate Discussion, Helper hours, Piazza, grading...
CompSci 201 is over
Time for a computer science story…
Recursion has saved the day!