Through Visualization and Interaction, Computer Science Concepts Come Alive

Susan H. Rodger
Duke University

March 29, 2022
Purdue University
Outline

• **My Path**

• **CS Concepts Come Alive**
  • Alice Programming Language
  • Algorithm Visualization
  • Automata Theory with JFLAP
  • Solving Problems with Seven Steps

• Diversity Efforts
A long time ago, back in 1979....
B.S. Computer Science and Mathematics

• My first semester, my first course in programming - PL/I

Hello2: proc options(main);
  put list ('Hello, world!');
end Hello2;

\[ X = X + 3; \quad /* \text{THIS IS AN ASSIGNMENT STATEMENT} */ \]
Decisions? Industry? Grad School?

• Systems Programmer
  • NCSU, University Systems Control Center
• Undergraduate Research
  • Cleanup data from buoys in the water
• Wasn’t thinking about grad school
• Be sure to encourage students to think about graduate school!
• Started in 1983
• Teaching Assistant for intro programming in Fortran
• Punch cards...
• In trouble with email...
Finished Graduate School!

- PhD Purdue University 1989
  - Computational Geometry
  - Parallel Scheduling Algorithms
- New Data Structure
  - Dynamic contour search tree
Assistant Professor

• Continued research in algorithms

• CAREER CHANGE....

• Got more interested in education
Started developing education tools
Changed area to Visualization Tools and CS Education

- Tool – NPDA - to experiment with pushdown automata
1994 – Moved to Duke University Professor of the Practice

• Position focuses on Education in the Discipline
About Me - Hobby – Baking Shape cakes

The Wiggles magazine
Issue No. 42
How do you make those cakes?
Outline

• My Path

• CS Concepts Come Alive
  • Alice Programming Language
  • Algorithm Visualization
  • Automata Theory with JFLAP
  • Solving Problems with Seven Steps

• Diversity Efforts
CS Concepts Coming Alive

• What data structure is this?
YARN, in the shape of a binary tree. Subtrees made with molecule kit. What is it?
2D-range tree

• Search in x-y plane
• Main tree organized by x-values
• Subtree organized by y values
Binary Search tree of points in the plane – sorted by X-value

In the x-range

Each subtree organized by y-value

Search each subtree by y-value
Outline

• Introduction

• CS Concepts Come Alive
  – Alice Programming Language
  – Algorithm Visualization
  – Automata Theory with JFLAP
  – Solving Problems with Seven Steps

• Diversity Efforts Sprinkled in...
Alice Programming Language

• Create interactive stories or games
• Learn programming in an easy way, drag-and-drop your code
• Problem solving with visual feedback
  • Objects are visual!
• Alice is free: www.alice.org
• Developed by Randy Pausch
More on … Alice Programming Language

• Has libraries of 3D objects

• Keeps Track of objects you select
Objects Have Multiple Parts that are moveable
Alice Code is Easy to Learn

Select Code, Drag-and-Drop code in program
Play Alice Animation

• Chicken rises, cow turns head and talks

Moo Moo Moo
Computer Science Concepts come alive with Alice - Examples

• Objects - visible
• Variables - see how they are changing
• Inheritance - visual
• Lists/Arrays - visual
Objects are visible
Variables – Timer and Score
Example - Inheritance

• Start with a chicken object

• Rename it to
  TalentedChicken
    • Change its color
    • Resize it larger
    • Add new methods (jump, fly, scurry)
    • Add events for this chicken

• Save this new class
  TalentedChicken that inherits from the Chicken class
Example list
Example – Arrays
Shuffle, then Selection Sort

Sort by height
Outline

• Introduction

• CS Concepts Come Alive
  • Alice Programming Language
  • Algorithm Visualization
  • Automata Theory with JFLAP
  • Additional Ways to Engage with CS

• Diversity Efforts
Algorithm Visualization/Animation Software/Aps/Videos

• Tango, Xtango, Samba, JSamba - Stasko (Georgia Tech)
• AnimalScript – Roessling (Darmstadt Univ of Tech, SIGCSE 2001)
• JHAVE – Naps (U. Wisc. Oshkosh, SIGCSE 2000)
• TRAKLA2 – Software Visualization Group – TKK Finland
• Lots of animations and systems on the web!
• Lots of videos of algorithm animations on the web!
Use of Algorithm Animation in CS 1/2

• Instructor
  • Make/Use animations for lecture
  • Stop/Pause – ask what will happen next
  • must be interactive

• Student
  • Create animations
  • Replay animations from lecture with same or new inputs
Lots of other software/programs for algorithm animation

• **Red Black Tree** – animation on web page

http://aleph0.clarku.edu/~achou/cs102/examples/bst_animation/RedBlackTree-Example.html

Student must have graduated. Link no longer works!
Python Tutor
Compute reverse of a list

```
Python 2.7
1  def reverse(numbers):
2      answer = []
3      for num in numbers:
4          answer.insert(0, num)
5      return answer
6
7  myList = [4, 7, 8, 3]
8  reversed = reverse(myList)
```

Edit code

Frames

Objects

Global frame

function reverse(numbers)

myList

list

```
0 1 2 3
```

4 7 8 3
Python Tutor
Compute reverse of a list

```python
def reverse(numbers):
    answer = []
    for num in numbers:
        answer.insert(0, num)
    return answer

myList = [4, 7, 8, 3]
reversed = reverse(myList)
```
Python Tutor

Compute reverse of a list

```python
def reverse(numbers):
    answer = []
    for num in numbers:
        answer.insert(0, num)
    return answer

myList = [4, 7, 8, 3]
reversed = reverse(myList)
```

Edit code
Compute reverse of a list

```python
def reverse(numbers):
    answer = []
    for num in numbers:
        answer.insert(0, num)
    return answer

myList = [4, 7, 8, 3]
reversed = reverse(myList)
```
Python 2.7

1 def reverse(numbers):
   2     answer = []
   3     for num in numbers:
   4         answer.insert(0, num)
   5     return answer
   6
   7 myList = [4, 7, 8, 3]
   8 reversed = reverse(myList)

Edit code
Python Tutor
Compute reverse of a list

```python
def reverse(numbers):
    answer = []
    for num in numbers:
        answer.insert(0, num)
    return answer

myList = [4, 7, 8, 3]
reversed = reverse(myList)
```

Frames
Objects

Global frame
- reverse
- myList

function reverse(numbers)
- list
  - 4 7 8 3
- list
  - 0 1
- num
  - 7
Python 2.7

1 def reverse(numbers):
   2     answer = []
  
 3     for num in numbers:
  4         answer.insert(0, num)
  5     return answer
  
6
7 myList = [4, 7, 8, 3]
8 reversed = reverse(myList)

Edit code
Python Tutor
Compute reverse of a list

```python
def reverse(numbers):
    answer = []
    for num in numbers:
        answer.insert(0, num)
    return answer

myList = [4, 7, 8, 3]
reversed = reverse(myList)
```

Frames
Global frame
reverse
myList

Objects
list
4 7 8 3

list
0 1 2 3

num
3
Electronic Textbooks (ebooks) engage students

- OpenDSA (Shaffer, Virginia Tech)
  - Algorithm animations built in
- runestoneinteractive.org (Brad Miller)
  - Several books (Python)
    - Python - try and run code built in
    - Quizzes
- ZyBooks – interactive textbooks
- Track student progress
- Requirements and design strategies for open source interactive computer science eBooks
  - ITiCSE 2013 Working Group (Korhonen, Naps, et al)
Index Operator: Working with the Characters of a String

The indexing operator (Python uses square brackets to enclose the index) selects a single character from a string. The characters are accessed by their position or index value. For example, in the string shown below, the 14 characters are indexed left to right from position 0 to position 13.

```
  0  1  2  3  4  5  6  7  8  9  10  11  12  13
L u t h e r  C o l l e g e
-14 -13 -12 -11 -10  -9  -8  -7  -6  -5  -4  -3  -2  -1
```

It is also the case that the positions are named from right to left using negative numbers where -1 is the rightmost index and so on. Note that the character at index 6 (or -8) is the blank character.

```
school = "Luther College"
m = school[2]
print(m)
lastchar = school[-1]
print(lastchar)
```
Run and edit code in the book

```python
school = "Luther College"
m = school[2]
print(m)
lastchar = school[-1]
print(lastchar)
```
Integrates in Python Tutor
Questions for feedback

Check your understanding

strings-4-1: What is printed by the following statements?

```python
s = "python rocks"
print(s[3])
```

- t
- h
- c
- Error, you cannot use the [ ] operator with a string.

Incorrect. Index locations do not start with 1, they start with 0.

Check Me  Compare me

strings-4-2: What is printed by the following statements?

```python
s = "python rocks"
print(s[2] + s[-5])
```

- tr
- ps
- nn
- Error, you cannot use the [ ] operator with the + operator.

Correct! Yes, indexing operator has precedence over concatenation.

Check Me  Compare me
Outline

• Introduction

• CS Concepts Come Alive
  • Alice Programming Language
  • Algorithm Visualization
  • Automata Theory with JFLAP
  • Solving Problems with Seven Steps
  • Additional Ways to Engage with CS

• Diversity Efforts
How does a compiler work?
Determining if a Java program is syntactically correct

- Finite state machine (or deterministic finite automaton - DFA) – to identify the words or tokens of the program
- Context-free grammar – to write the rules of the programming language
- LR Parsing determining if the program fits the rules – trying to derive the program. (modelled using a pushdown automaton)

- This area is known as Formal languages and Automata theory
Formal Languages and Automata Theory

• Traditionally taught
  • Pencil and paper exercises
  • No immediate Feedback!

• More mathematical than programming
• Less hands-on than most CS courses
Why Develop Tools for Automata?

| Textual | \( (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\}) \)  
|         | \( \delta = \{(q_0, b, q_0), (q_0, a, q_1), (q_1, a, q_0), (q_1, b, q_2), (q_2, a, q_1)\} \) |

| Tabular | | a | b |
|         | q_0 | q_1 | q_0 |
|         | q_1 |     | q_2 |
|         | q_2 |     |     |

| Visual |
|        |

| Interactive |
Overview of JFLAP

• **Java Formal Languages and Automata Package**

• Instructional tool to learn concepts of Formal Languages and Automata Theory

• Topics:
  - Regular Languages
  - Context-Free Languages
  - Recursively Enumerable Languages
  - Lsystems

• With JFLAP your creations come to life!
Thanks to Students - Worked on JFLAP and Automata Theory Tools

- NPDA - 1990, C++, Dan Caugherty
- JFLAP - 1996-1999, Java version
  Eric Gramond, Ted Hung, Magda and Octavian Procopiuc
- Pâté, JeLLRap, Lsys
  Anna Bilska, Jason Salemme, Lenore Ramm, Alex Karweit, Robyn Geer
- JFLAP 4.0 – 2003, Thomas Finley, Ryan Cavalcante
- JFLAP 6.0 – 2005-2008 Stephen Reading, Bart Bressler, Jinghui Lim, Chris Morgan, Jason Lee
- JFLAP 7.0 - 2009 Henry Qin, Jonathan Su
- JFLAP 8.0Beta – 2011-14 Julian Genkins, Ian McMahon, Peggy Li, Lawrence Lin, John Godbey
- JFLAP in OpenDSA – 2015 Sung-Hoon Kim and Martin Tamayo

Over 30 years!
DFA Example

• Build a deterministic finite automaton (DFA) to recognize even binary numbers with an even number of 1s.
• Only use symbols 0 and 1
• Binary numbers: 0, 1, 10, 11, 100, 101, 110, 111, ...
• When is a binary number an even number?
  • Ends in 0
• Which strings should be accepted?
• 11010, 10010, 1111, 10100

No, odd no. of 1’s       Yes       No, ends In 1       Yes
Build with JFLAP

START

q0

1

q1

FINISH

q2

0

q3

1

q4

0

FINISH
Simulation on 1101010
Simulation on 1101010
Simulation on 1101010
Simulation on 1101010
Simulation on
1101010
Simulation on 1101010
Simulation on 1101010
Accepts Input!

1101010
Add meaning to states!
The diagram represents a deterministic finite automaton (DFA) for determining whether a binary number has an odd number of 1's.

- **States:**
  - q0: Start state, only one 0.
  - q1: Only one 0.
  - q2: Odd number of 1's.
  - q3: Even number of 1's, ends in 1.
  - q4: Accept state.

- **Transitions:**
  - From q0, on input 1, go to q2.
  - From q2, on input 0, go to q3.
  - From q3, on input 1, go to q4.
  - From q3, on input 0, go to q1.
  - From q4, on input 0, loop back to q4.

The automaton accepts a binary string if it ends in state q4, indicating an odd number of 1's.
The diagram illustrates a finite automaton that checks if a binary number has an odd or even number of 1's, with the following states:

- **q0**: Start state. Transition on 0 to q1 and on 1 to q2.
- **q1**: State with label "only one 0". Transition on 0 to q1 and on 1 to q3.
- **q2**: State with label "odd number of 1's". Transition on 0 to q2 and on 1 to q4.
- **q3**: State with label "even number of 1's, ends in 1". Transition on 0 to q3 and on 1 to q4.
- **q4**: Accepting state with label "even number of 1's, ends in 0". Transition on 0 to q4 and on 1 to q4.
Test Multiple Inputs
Example: Build an NFA for valid integers

• Example:
  • Valid integers {-3, 8, 0, 456, 13, 500, ...}
  • Not valid: {006, 3-6, 4.5, ...}
Example: NFA for all valid integers
NFA annotated and shortcut

• Shortcut: [1-9] on labels
Another Example: Grammar

• Grammar – set of replacement rules to define a language
• Grammar for $a^n b^n c^n$
• Why look at such a grammar?
• Consider representing underlined words in a text file (to be interpreted later):
  • cookie&&&&&______                   cookie
  & = go back one
Grammar for $a^n b^n c^n$

- Unrestricted grammar
- Generates strings with an equal number of a’s, b’s, c’s
- a’s first, then b’s, then c’s
- Example strings can derive:
  - abc
  - aabbcc
  - aabbbcccc
  - aaaaabbbbbcccccc
  - ...
Example Derivation for aabbcc

\[ S \rightarrow AX \]

rule: \[ S \rightarrow AX \]
Example Derivation for aabbcc

S $\rightarrow$ AX  
   $\rightarrow$ aAbcX

rule: S $\rightarrow$ AX
rule: A $\rightarrow$ aAbc
Example Derivation for aabbcc

S → AX  
→ aAbcX  
→ aABbcbcX

rule: S -> AX 
rule: A -> aAbc 
rule: A -> aBbc

NOTE: We have generated the correct symbols, aabcbc, but they are in the wrong order!
Example Derivation for aabbcc

S → AX  
→ aAbcX  rule: S -> AX
→ aAbcX  rule: A -> aAbc
→ aaBbcbcX  rule: A -> aBbc
→ aaBbcbcX  rule: Bb -> bB
→ aaBbcbcX
Example Derivation for aabbcc

S \rightarrow AX

\rightarrow aAbcX

\rightarrow aaBbcX

\rightarrow aaBBbcX

\rightarrow aaBdbbcX

Note: the D absorbed the c!
Example Derivation for aabbcc

S → AX
→ aAbcX
→ aaBbcbcX
→ aabBbcbcX
→ aabDbcX
→ aabDbcX

rule: S -> AX
rule: A -> aAbc
rule: A -> aBbc
rule: Bb -> bB
rule: Bc -> D
rule: Db -> bD
Example Derivation for aabbccc

S \rightarrow AX
    \rightarrow aAbcX (rule: S \rightarrow AX)
    \rightarrow aaBbcbcX (rule: A \rightarrow aAbc)
    \rightarrow aabBbcbcX (rule: A \rightarrow aBbc)
    \rightarrow aabBcbcX (rule: Bb \rightarrow bB)
    \rightarrow aabDbcX (rule: Bc \rightarrow D)
    \rightarrow aabbbDcX (rule: Db \rightarrow bD)
    \rightarrow aabbDcX (rule: Dc \rightarrow cD)
Example Derivation for aabbcc

S → AX
  → aAbcX
  → aaBbcX
  → aabBbcX
  → aabDBbcX
  → aabbDbcX
  → aabbcDX
  → aabbcEXc

Eventually ... → aabbcc

Note the c spit out on right end!
We could have done this derivation of aabbcc with JFLAP.

Now let’s see how JFLAP visualizes this derivation with a “parse tree”

Parse DAG
<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AX</td>
</tr>
<tr>
<td>A</td>
<td>aAbc</td>
</tr>
<tr>
<td>A</td>
<td>aBbc</td>
</tr>
<tr>
<td>Bb</td>
<td>bB</td>
</tr>
<tr>
<td>Bc</td>
<td>D</td>
</tr>
<tr>
<td>Dc</td>
<td>cD</td>
</tr>
<tr>
<td>Db</td>
<td>bD</td>
</tr>
<tr>
<td>DX</td>
<td>EXc</td>
</tr>
<tr>
<td>BX</td>
<td>\lambda</td>
</tr>
<tr>
<td>cE</td>
<td>Ec</td>
</tr>
<tr>
<td>bE</td>
<td>Eb</td>
</tr>
<tr>
<td>aE</td>
<td>aB</td>
</tr>
</tbody>
</table>

Input: aabbcc
String accepted! 51 nodes generated.
Input: aabbcc
String accepted! 51 nodes generated.

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AX</td>
</tr>
<tr>
<td>A</td>
<td>aAbc</td>
</tr>
<tr>
<td>A</td>
<td>aBbc</td>
</tr>
<tr>
<td>Bb</td>
<td>bB</td>
</tr>
<tr>
<td>Bc</td>
<td>D</td>
</tr>
<tr>
<td>Dc</td>
<td>cD</td>
</tr>
<tr>
<td>Db</td>
<td>bD</td>
</tr>
<tr>
<td>DX</td>
<td>E xc</td>
</tr>
<tr>
<td>BX</td>
<td>λ</td>
</tr>
<tr>
<td>cE</td>
<td>Ec</td>
</tr>
<tr>
<td>bE</td>
<td>Eb</td>
</tr>
<tr>
<td>aE</td>
<td>aB</td>
</tr>
</tbody>
</table>

Derived AX from S.
Input: aabbcc
String accepted! 51 nodes generated.

LHS | RHS
---|---
S   | AX
A   | aBbc
A   | aBbc
Bb  | bB
Bc  | D
Dc  | cD
Db  | bD
DX  | EXc
BX  | λ
cE  | Ec
bE  | Eb
aE  | aB

Derived aBbc from A.
Note all letters there, but wrong order: aabcbbc
Input: aabbcc
String accepted! 51 nodes generated.

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AX</td>
</tr>
<tr>
<td>A</td>
<td>aA</td>
</tr>
<tr>
<td>A</td>
<td>aB</td>
</tr>
<tr>
<td>Bb</td>
<td>bB</td>
</tr>
<tr>
<td>Bc</td>
<td>D</td>
</tr>
<tr>
<td>Dc</td>
<td>cD</td>
</tr>
<tr>
<td>Db</td>
<td>bD</td>
</tr>
<tr>
<td>DX</td>
<td>EXc</td>
</tr>
<tr>
<td>BX</td>
<td>λ</td>
</tr>
<tr>
<td>cE</td>
<td>Ec</td>
</tr>
<tr>
<td>bE</td>
<td>Eb</td>
</tr>
<tr>
<td>aE</td>
<td>aB</td>
</tr>
</tbody>
</table>

Derived bB from Bb.
What’s happening?
Bb → bB
Absorb the “c”
Input: aabbcc
String accepted! 51 nodes generated.

LHS       RHS
S → AX
A → aAbc
A → aBbc
Bb → bB
Bc → D
Dc → cD
Db → bD
DX → EXc
BX → λ
cE → Ec
bE → Eb
aE → aB

Derived bD from Db.
String accepted! 51 nodes generated.

Derived cD from Dc.
Spit out the “c” at the right end
String accepted! 51 nodes generated.

Derived Ec from cE.
Input: aabbcc
String accepted! 51 nodes generated.

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AX</td>
</tr>
<tr>
<td>A</td>
<td>aAbc</td>
</tr>
<tr>
<td>A</td>
<td>aBbc</td>
</tr>
<tr>
<td>Bb</td>
<td>bB</td>
</tr>
<tr>
<td>Bc</td>
<td>D</td>
</tr>
<tr>
<td>Dc</td>
<td>cD</td>
</tr>
<tr>
<td>Db</td>
<td>bD</td>
</tr>
<tr>
<td>DX</td>
<td>EXc</td>
</tr>
<tr>
<td>BX</td>
<td>λ</td>
</tr>
<tr>
<td>cE</td>
<td>Ec</td>
</tr>
<tr>
<td>bE</td>
<td>Eb</td>
</tr>
<tr>
<td>aE</td>
<td>aB</td>
</tr>
</tbody>
</table>

Derived Eb from bE.
Input: aabbcc
String accepted! 51 nodes generated.

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AX</td>
</tr>
<tr>
<td>A</td>
<td>aAbc</td>
</tr>
<tr>
<td>A</td>
<td>aBbc</td>
</tr>
<tr>
<td>Bb</td>
<td>bB</td>
</tr>
<tr>
<td>Bc</td>
<td>D</td>
</tr>
<tr>
<td>Dc</td>
<td>cD</td>
</tr>
<tr>
<td>Db</td>
<td>bD</td>
</tr>
<tr>
<td>DX</td>
<td>EXc</td>
</tr>
<tr>
<td>BX</td>
<td>λ</td>
</tr>
<tr>
<td>cE</td>
<td>Ec</td>
</tr>
<tr>
<td>bE</td>
<td>Eb</td>
</tr>
<tr>
<td>aE</td>
<td>aB</td>
</tr>
</tbody>
</table>

Derived Eb from bE.
Input: aabbcc

String accepted! 51 nodes generated.

Derived aB from aE.
Input: aabbcc
String accepted! 51 nodes generated.

LHS  | RHS
---|---
S    | AX
A    | aA
A    | aAb
Bb   | bB
Bc   | D
Dc   | cD
Db   | bD
DX   | EXc
BX   | \(\lambda\)
cE   | Ec
bE   | Eb
aE   | aA

Derived bB from Bb.
Input: aabbcc
String accepted! 51 nodes generated.

LHS | RHS
---|---
S   | AX
A   | aAbc
A   | aBbc
Bb  | bB
Bc  | D
Dc  | cD
Db  | bD
DX  | EXc
BX  | λ
cE  | Ec
bE  | Eb
aE  | aB

Derived bB from Bb.
Absorb second “c”
Spit the “c” out at right end
Input: aabbcc
String accepted! 51 nodes generated.

LHS | RHS
---|---
S  | AX
A  | aAbcc
A  | aBbc
Bb | bB
Bc | D
Dc | cD
Db | bD
DX | EXc
BX | \(\lambda\)
cE | Ec
bE | Eb
aE | aB

Derived Eb from bE.
Input: aabbcc
String accepted! 51 nodes generated.

LHS | RHS
--- | ---
S   | AX
A   | aAbc
A   | aBbc
Bb  | bB
Bc  | D
Dc  | cD
Db  | bD
DX  | EXc
BX  | λ
cE  | Ec
bE  | Eb
aE  | aB

Derived Eb from bE.
Input: aabbc

String accepted! 51 nodes generated.

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AX</td>
</tr>
<tr>
<td>A</td>
<td>aA</td>
</tr>
<tr>
<td>A</td>
<td>aB</td>
</tr>
<tr>
<td>Bb</td>
<td>bB</td>
</tr>
<tr>
<td>Bc</td>
<td>cD</td>
</tr>
<tr>
<td>Db</td>
<td>bD</td>
</tr>
<tr>
<td>DX</td>
<td>DXc</td>
</tr>
<tr>
<td>BX</td>
<td>λ</td>
</tr>
<tr>
<td>cE</td>
<td>Ec</td>
</tr>
<tr>
<td>bE</td>
<td>Eb</td>
</tr>
<tr>
<td>aE</td>
<td>aB</td>
</tr>
</tbody>
</table>

Derived aB from aE.
String accepted! 51 nodes generated.
Input: aabbcc
String accepted! 51 nodes generated.

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AX</td>
</tr>
<tr>
<td>A</td>
<td>aAbc</td>
</tr>
<tr>
<td>A</td>
<td>aBbc</td>
</tr>
<tr>
<td>Bb</td>
<td>bB</td>
</tr>
<tr>
<td>Bc</td>
<td>D</td>
</tr>
<tr>
<td>Dc</td>
<td>cD</td>
</tr>
<tr>
<td>Db</td>
<td>bD</td>
</tr>
<tr>
<td>DX</td>
<td>EXc</td>
</tr>
<tr>
<td>BX</td>
<td>λ</td>
</tr>
<tr>
<td>cE</td>
<td>Ec</td>
</tr>
<tr>
<td>bE</td>
<td>Eb</td>
</tr>
<tr>
<td>aE</td>
<td>aB</td>
</tr>
</tbody>
</table>

Derived bB from Bb.
Input: aabbcc
String accepted! 51 nodes generated.
What else can JFLAP do?

• Create other machines
  • Moore and Mealy
  • Pushdown Automaton
  • Turing machine

• Parsing of grammars
  • regular, context-free grammars
  • Unrestricted grammar

• Conversions for proofs
  • NFA to DFA to minimal DFA
  • NFA $\leftrightarrow$ regular expression
  • NFA $\leftrightarrow$ regular grammar
  • CFG $\leftrightarrow$ NPDA
L-Systems may be used to model biological systems and create fractals.

Similar to Chomsky grammars, except all variables are replaced in each derivation step, not just one!

Commonly, strings from successive derivations are interpreted as strings of render commands and are displayed graphically.
**Axiom:** \( R \sim \#\# B \)

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>( [\sim ## TL - B + + B] )</td>
</tr>
<tr>
<td>L</td>
<td>( [\text{angle}=15 { -g + + g % - - g }] )</td>
</tr>
<tr>
<td>R</td>
<td>!@@@ R</td>
</tr>
<tr>
<td>T</td>
<td>( T g )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>15</td>
</tr>
<tr>
<td>color</td>
<td>brown</td>
</tr>
<tr>
<td>polygonColor</td>
<td>forestGreen</td>
</tr>
</tbody>
</table>

\( \text{L-System} = (A, \Sigma, R) \)
Add second T rule
Expansion contains 532 Symbols!
L-Systems

The same stochastic L-system, rendered 3 different times all at the 9th derivation.
Students like L-systems
Two-year JFLAP Study
2005-2007

Fourteen Faculty Adopter Participants

- small, large
- public, private
- includes minority institutions

• Duke
• UNC-Chapel Hill
• Emory
• Winston-Salem State University
• United States Naval Academy
• Rensselaer Polytechnic Institute
• UC Davis
• Virginia State University
• Norfolk State University
• University of Houston
• Fayetteville State University
• University of Richmond
• San Jose State University
• Rochester Institute of Technology
Conclusions From Study

• Results of Study showed
  • All the faculty used JFLAP in their courses, mostly for homework, some in lecture
  • Students had a high opinion of JFLAP
  • Majority of students felt access to JFLAP
    • Made learning course concepts easier
    • Made them feel more engaged
    • Made the course more enjoyable
  • Over half the students used JFLAP to study for exams
  • Over half the students thought time and effort using JFLAP helped them get a better grade.
Now a few tips if you ever write educational software...
Make your tool as interactive as possible – but not too tedious!

• User shouldn’t type everything
• Sometimes select
• Example: DFA to regular expression in JFLAP
Allow user to proceed on if they got it

- Complete the rest for them

- Complete parts for them
Avoid Too Many Pop up windows

• OLD JFLAP LR PARSE TOOL
Add Pause/Checkpoint questions

• Allow for pause to think about what comes next
• Undo/go back

• Pop up a quiz question to see if the user understands what he/she just did
  • JHAVE tool does this
  • Can integrate into ebooks
What can make the tool more useable?

• Annotations on states
• Multiple run window
  • Develop test data
  • Easier for grading
• General definitions
  • FA – recognize one or more symbols
  • NPDA – pop or push 0 or more symbols
• Batch processing
Naming your software

What is a “good” name for your tool?
Jawaa

- Algorithm Animation tool

**Rectangle**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>a name uniquely identifying this rectangle</td>
</tr>
<tr>
<td>x</td>
<td>x-coordinate</td>
</tr>
<tr>
<td>y</td>
<td>y-coordinate</td>
</tr>
<tr>
<td>width</td>
<td>width of the rectangle</td>
</tr>
<tr>
<td>height</td>
<td>height of the rectangle</td>
</tr>
<tr>
<td>color</td>
<td>color of the rectangle outline</td>
</tr>
<tr>
<td>bgclr</td>
<td>color of the rectangle's background</td>
</tr>
</tbody>
</table>

Example:

rectangle r1 10 20 100 120 black red
rectangle r2 150 20 180 60 cyan yellow

The first example will create a rectangle with its upper left corner at (10,20) and rectangle will be red with a black outline, as shown in the figure below on the left. The second example will create a rectangle with its upper corner at (150,20) and rectangle will be yellow with a cyan outline. This is shown in the figure below on the right.
JAWAA name is not unique

How popular is JAWAA?

Results 241 - 250 of about 16,100 for JAWAA. (0.18 seconds)
FLAP

- **Formal Languages and Automata Package**
- 1996 – converted to Java
- FLAP -> JFLAP
JFLAP name is unique
Much more than Google Analytics
Forums, Blogs, Course websites

Newest 'jflap' Questions - Stack Overflow
stackoverflow.com/questions/tagged/jflap
We can use small letters for terminals and caps for Non-terminals in JFLAP while entering grammar. But this restricts to only 26 options. Can we have more ...

Blog:Recent posts - JFLAP
jflap.wikia.com/wiki/Blog:Recent_posts

CS 301: Using JFLAP
www.cs.colostate.edu/~massey/Teaching/.../JFLAP/gettingstarted.html
This course uses the JFLAP package. According to the JFLAP website, JFLAP is a package of graphical tools which can be used as an aid in learning the basic ...

[PDF] JFLAP Startup
Download JFLAP and the files referenced in this book from www.jflap.org to get started. JFLAP is written in Java to allow it to run on a range of platforms.
JFLAP is free

www.jflap.org

JFLAP tutorial
Outline

• Introduction
• CS Concepts Come Alive
  • Alice Programming Language
  • Algorithm Visualization
  • Automata Theory with JFLAP
  • Solving Problems with Seven Steps
• Diversity Efforts
Stuck on solving a problem? Don’t know where to start?

• Use the 7 step process!

• CompEd 2019, Translation from Problem to Code in Seven Steps, Hilton, Lipp and Rodger
1. Work small examples by hand
2. Write down what you did in words (algorithm)
3. Find Patterns (generalize algorithm)
4. Work another example by hand (algorithm work? If not, go back to 3, or 1)
Problem Solving to Code – Steps 5-7

5. Translate to code
6. Test several cases
7. Debug **failed** test cases
Problem Statement

Strange abbreviations are often used to write text messages on uncomfortable mobile devices. One particular strategy for encoding texts composed of alphabetic characters and spaces is the following:

- Spaces are maintained, and each word is encoded individually. A word is a consecutive string of alphabetic characters.

- If the word is composed only of vowels, it is written exactly as in the original message.

- If the word has at least one consonant, write only the consonants that do not have another consonant immediately before them. Do not write any vowels.

- The letters considered vowels in these rules are 'a', 'e', 'i', 'o' and 'u'. All other letters are considered consonants.

For instance, "ps i love u" would be abbreviated as "p i lv u" while "please please me" would be abbreviated as "ps ps m". You will be given the original message in the string parameter original. Return a string with the message abbreviated using the described strategy.
Examples

1. "text message"
   Returns "tx msg"

5. "aeiou bcdfghjklmnpqrstvwxyz"
   Returns: "aeiou b"
Focus on transforming one word
Write helper function \textit{transform}

• How?
• Use seven steps
• Work an example by hand
Transform word - Step 1: work small example by hand

• Word is “please”
• Letter is ‘p’, YES
• answer is “p”
• Letter is ‘l’, NO
• Letter is ‘e’, NO
• Letter is ‘a’, NO
• Letter is ‘s’, YES
• answer is “ps”
• Letter is ‘e’, NO
Step 2: Describe what you did

• Word is “please”, create an empty answer
• Letter is ‘p’, consonant, no letter before, YES
• Add ‘p’ to answer
• Letter is ‘l’, consonant, letter before “p”, NO
• Letter is ‘e’, vowel, letter before ‘l’, NO
• Letter is ‘a’, vowel, letter before ‘e’, NO
• Letter is ‘s’, consonant, letter before ‘a’, YES
• Add ‘s’ to answer
• Letter is ‘e’, vowel, letter before ‘s’, NO
• Answer is “ps”
Step 3: Find Pattern and generalize

Need to initialize letter before, pick “a”
answer is empty
for each letter in word
  If it is a consonant, and the letter before is a vowel, then add the letter to the answer
This letter is now the letter before
return answer
Step 4 – Work another example

• Word is message
• Letter is ‘m’, before is ‘a’, add ‘m’ to answer
• Letter is ‘e’, before is ‘m’, NO
• Letter is ‘s’, before is ‘e’, add ‘s’ to answer
• Letter is ‘s’, before is ‘s’, NO
• Letter is ‘a’, before is ‘s’, NO
• Letter is ‘g’, before is ‘a’, add ‘g’ to answer
• Letter is ‘e’, before is ‘g’, NO
• Answer is “msg” WORKS!!
Step 5: Translate to Code

# Letter before is “a”       # start with a vowel

# answer is empty

# for each letter in word
Step 5: Translate to Code

# Letter before is “a”       # start with a vowel
before = ‘a’
# answer is empty
answer = []          # or this could be an empty string
# for each letter in word
for ch in word:
Step 5: Translate to Code (code)

#If it is a consonant, and the letter before is a vowel, then add the letter to the answer

#This letter is now the letter before

# return answer
Step 5: Translate to Code (code)

```python
# If it is a consonant, and the letter before is a vowel, then add the letter to the answer
if !(isVowel(ch)) and isVowel(before):
    answer += ch

# This letter is now the letter before
before = ch

# return answer
return answer
```

Compsci 101, Spring 2022
Student Anecdotes

• From CompSci 101
  • “I just want to tell you that I tried the seven step method, and I worked on all of my code for one or two hours before I even looked at the computer. AND IT WORKED! I got all my code right on the first try! For the first time ever, I don’t have to go to the help lab ...”
Student Anecdotes

• From Coursera course
  • “I have been programming for a couple of years. Learned from so many resources but none said how to write the algorithm, they just say you should write your algorithm first. The steps illustrated here are beautiful and definitely help to understand how to decompose a problem.”
Outline

• Introduction

• CS Concepts Come Alive
  • Alice Programming Language
  • Algorithm Visualization
  • Automata Theory with JFLAP
  • Solving Problems with Seven Steps

• Diversity Efforts
Success - Alice attracts diverse group

- At Duke
  - CompSci 4 Spring 2005
    - 22 preregister, 30 enroll (12 female + 3 African Amer.)
  - CompSci 4 Fall 2005
    - 20 preregister, 31 enroll (17 female + 1 African Amer.)
  - CompSci 4 Fall 2006 – 2 sections
    - 64 students, 33 female, 7 African Amer.
  - CompSci 4 Fall 2007 – 2 sections
    - 84 students -> 50% female
  - CompSci 4 Fall 2008 – 2 sections
    - 100 students -> 50% female
  - Same for Spring 2009, Fall 2009...
  - Advertised in school paper
    - picture of ice skater
    - Web site of animations
  - This course is now CompSci 94
Success - Alice Excites 4th-6th Grade Girls

• Duke Femmes Event, April 07
• 60 girls – 4 groups of 15
• Taught them Alice for an hour
• Handout to take home
• Event again in 2008, almost every year since
Adventures in Alice Programming
www.cs.duke.edu/csed/alice/aliceInSchools

• 2-week Teacher workshops
  • Over 500 teachers, middle school, high school, some elementary
  • First week Teach Alice, Practice
  • Second week - Develop Lesson Plans
  • All disciplines: math, science, history, language arts, foreign language, art, music, business
  • Summers 2008-2017

• Main Sites:
  • Duke University, Durham, NC
  • Charleston/Columbia, SC
  • San Jose, CA
  • Lincoln, Nebraska

• THANKS IBM and NSF
CRA-WP Board

• Organize Career Mentoring Workshops for Women and underrepresented groups
  • Early Career Workshop
    • Asst Prof, PhD students, PostDocs, Industry
  • Mid-Career Workshop
    • Assoc Prof, Industry Equiv

• Grad Cohort for Women
  • For Graduate students in first 3 years
How Visible are Notable Women in Computer Science?

• Pondered this question in early 2012
• Looked at Wikipedia
  • The internet encyclopedia
  • Who writes those pages?
  • Why did some notables have pages and others not?
• Turing Award Winners
  • Only two women at that time
Fran Allen

- School teacher – got a job at IBM
- Compilers and Optimization Technology
- IBM Fellow – First Women
- Turing Award (2006) – First Woman
- The Turing Award was announced on Feb. 21, 2007
- Her Wikipedia page was created on...
  - Feb. 6, 2007
- On Feb 21, 2007 the Turing Award was added to her Wikipedia page.
Fran Allen has made outstanding contributions to the field of programming languages for more than forty-five years, and her work has significantly influenced the wider computer science community.

Ms. Allen is a pioneer in the field of optimizing compilers. Her achievements include seminal work in compilers, code optimization, and parallelization. In the early 1980s, she formed the Parallel TRANslation (PTRAN) group to study the issues involved in compiling for parallel machines. The group was considered one of the top research groups in the world working with parallelization issues. Her work on these projects culminated in algorithms and technologies that form the basis for the theory of program optimization and are widely used in today's commercial compilers throughout the industry.

Ms. Allen's influence on the IBM community was recognized by her appointment as an IBM fellow, the first woman to receive this recognition. She was also president of the IBM Academy of Technology. The Academy plays an important role in the corporation by providing technical leadership, advancing the understanding of key technical areas and fostering communications among technical professionals.

In 1997, Ms. Allen was inducted into the WITI Hall of Fame. Ms. Allen retired from IBM in 2002.
Three days later...

Frances E. Allen

From Wikipedia, the free encyclopedia

Fran Allen is a pioneer in the field of optimizing compilers. Her achievements include seminal work in compilers, code optimization, and parallelization.

In the early 1980s, she formed the Parallel TRANslation (PTRAN) group to study the issues involved in compiling for parallel machines. The group was considered one of the top research groups in the world working with parallelization issues. Her work on these projects culminated in algorithms and technologies that form the basis for the theory of program optimization and are widely used in today's commercial compilers throughout the industry.

Ms. Allen's influence on the IBM community was recognized by her appointment as an IBM Fellow, the first woman to receive this recognition. She was also president of the IBM Academy of Technology. The Academy plays an important role in the corporation by providing technical leadership, advancing the understanding of key technical areas and fostering communications among technical professionals.

In 1997, Ms. Allen was inducted into the WITI Hall of Fame. Ms. Allen retired from IBM in 2002.

This article has not been added to any categories. Please help out by adding categories to it so that it can be listed with similar articles.
In 1997, Ms. Allen was inducted into the WITI Hall of Fame. Ms. Allen retired from IBM in 2002.

Early 2007, she became the first woman to win the A.M. Turing Award.

A. M. Turing Award laureates

<table>
<thead>
<tr>
<th>Year</th>
<th>Name(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1966</td>
<td>Alan Perlis</td>
</tr>
<tr>
<td>1967</td>
<td>Maurice Vincent Wilkes</td>
</tr>
<tr>
<td>1968</td>
<td>Richard Hamming</td>
</tr>
<tr>
<td>1969</td>
<td>Marvin Minsky</td>
</tr>
<tr>
<td>1970</td>
<td>James H. Wilkinson</td>
</tr>
<tr>
<td>1971</td>
<td>John McCarthy</td>
</tr>
<tr>
<td>1972</td>
<td>Edsger W. Dijkstra</td>
</tr>
<tr>
<td>1973</td>
<td>Charles Bachman</td>
</tr>
<tr>
<td>1974</td>
<td>Donald Knuth</td>
</tr>
<tr>
<td>1975</td>
<td>Allen Newell / Herbert A. Simon</td>
</tr>
<tr>
<td>1976</td>
<td>Michael O. Rabin / Dana Scott</td>
</tr>
<tr>
<td>1977</td>
<td>John Backus</td>
</tr>
<tr>
<td>1978</td>
<td>Robert W. Floyd</td>
</tr>
<tr>
<td>1979</td>
<td>Kenneth E. Iverson</td>
</tr>
<tr>
<td>1980</td>
<td>Tony Hoare</td>
</tr>
<tr>
<td>1981</td>
<td>Edgar F. Codd</td>
</tr>
<tr>
<td>1982</td>
<td>Stephen Cook</td>
</tr>
<tr>
<td>1983</td>
<td>Ken Thompson / Dennis Ritchie</td>
</tr>
<tr>
<td>1984</td>
<td>Niklaus Wirth</td>
</tr>
<tr>
<td>1985</td>
<td>Richard Karp</td>
</tr>
<tr>
<td>1986</td>
<td>John Hopcroft / Robert Tarjan</td>
</tr>
<tr>
<td>1987</td>
<td>John Cocke</td>
</tr>
<tr>
<td>1988</td>
<td>Ivan Sutherland</td>
</tr>
<tr>
<td>1989</td>
<td>William Kahan</td>
</tr>
<tr>
<td>1990</td>
<td>Fernando J. Corbató</td>
</tr>
<tr>
<td>1991</td>
<td>Robin Milner</td>
</tr>
<tr>
<td>1992</td>
<td>Butler Lampson</td>
</tr>
<tr>
<td>1993</td>
<td>Juris Hartmanis / Richard E. Stearns</td>
</tr>
<tr>
<td>1994</td>
<td>Edward Feigenbaum / Raj Reddy</td>
</tr>
<tr>
<td>1995</td>
<td>Manuel Blum</td>
</tr>
<tr>
<td>1996</td>
<td>Amir Pnueli</td>
</tr>
<tr>
<td>1997</td>
<td>Douglas Engelbart</td>
</tr>
<tr>
<td>1998</td>
<td>Jim Gray</td>
</tr>
<tr>
<td>1999</td>
<td>Fred Brooks</td>
</tr>
<tr>
<td>2000</td>
<td>Andrew Yao</td>
</tr>
<tr>
<td>2001</td>
<td>Ole-Johan Dahl / Kristen Nygaard</td>
</tr>
<tr>
<td>2001</td>
<td>Ron Rivest / Adi Shamir / Leonard Adleman</td>
</tr>
<tr>
<td>2002</td>
<td>Alan Kay</td>
</tr>
<tr>
<td>2003</td>
<td>Vint Cerf / Bob Kahn</td>
</tr>
<tr>
<td>2004</td>
<td>Peter Naur</td>
</tr>
<tr>
<td>2005</td>
<td>Frances E. Allen</td>
</tr>
<tr>
<td>2006</td>
<td>Frances E. Allen</td>
</tr>
</tbody>
</table>

Categories: Turing Award laureates
In the next three days

• Over 30 edits, added awards, boards

Awards and honors

Allen is a member of the National Academy of Engineering, a fellow of the IEEE, the Association for Computing Machinery (ACM) and the American Academy of Arts and Sciences. She is currently on the Computer Science and Telecommunications Board, the Computer Research Associates (CRA) board and National Science Foundation's CISE Advisory Board.

In 1997, Allen was inducted into the WITI Hall of Fame. She retired from IBM in 2002 and won the Augusta Ada Lovelace Award that year from the Association for Women in Computing. In 2007, she became the first woman to win the A.M. Turing Award.
Frances E. Allen

From Wikipedia, the free encyclopedia

For the early American nun, see Frances Allen (nun).

Frances Elizabeth "Fran" Allen (born August 4, 1932) is an American computer scientist and pioneer in the field of optimizing compilers. Her achievements include seminal work in compilers, code optimization, and parallelization. She also had a role in intelligence work on programming languages and security codes for the National Security Agency.[2][3]

Allen was the first female IBM Fellow and in 2006 became the first woman to win the Turing Award.[4]

Contents [hide]
1 Career
2 Awards and honors
3 See also
4 References
5 External links

Career [edit]

Allen grew up on a farm in Peru, New York and graduated from The New York State College for Teachers (now State University of New York at Albany) with a B.Sc. degree in mathematics in 1954.[8] She earned an M.Sc. degree in mathematics at the University of Michigan in 1957 and began teaching school in Peru, New York.[8] Deeply in debt, she joined IBM on July 15, 1957 and planned to stay only until her school loans were paid, but ended up staying for her entire 45-year career.

Fran Allen's work has had an enormous impact on compiler research and
What about other Notable Women in Computer Science?

• ACM Fellows
  • Few women
    • 1994 first year over 130 Fellows
      • 9-12 were women? Less than 10%
  • About 20-50 Fellows per year
  • 2014 – 47 fellows, 6-8 women
  • Noticed few of Women had Wikipedia pages
Write Wikipedia pages for Notable women in Computing

• How hard is it to write a Wikipedia page?
  • Lots of rules you have to follow

• Another area with few women
  • 2013 study – 16% of Wikipedia writers are female
Some Rules in Writing Wikipedia Biography pages

• You cannot write your own page!
• Neutral point of view
• Person must be notable
• Be careful!
  • Must write only facts and reference them
  • Must be verifiable
  • Do not plagiarize – write in your own words
• Regard for subject’s privacy
  • NOT A TABLOID!
CRA-W and Anita Borg Institute Wikipedia Project Writing Wikipedia Pages for Notable Women in Computing

About this project

This project started when it was recognized that there are very few notable women (or famous women or leading women) computer scientists who have Wikipedia pages. For example, a large number of women with notable awards such as ACM Fellow, IEEE Fellow, ACM Distinguished Educator, Scientist or Engineer, or other awards are not yet in Wikipedia. As a result, in April 2012, a few members...
Our Database of Notable Women in CS

- Over 300 women
- Why notable
- Status of their Wikipedia page
- Forms for adding women and updating status

<table>
<thead>
<tr>
<th>Title/Position</th>
<th>Web page</th>
<th>Prestigious Award or why notable</th>
<th>Wikipedia page?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Professor of Human-Computer Interaction, CS</td>
<td><a href="http://www.daimi.au.dk/~bodke">http://www.daimi.au.dk/~bodke</a></td>
<td>Member, CHI Academy</td>
<td>no page</td>
</tr>
<tr>
<td>Founder</td>
<td><a href="http://anitaborg.org/about/history">http://anitaborg.org/about/history</a></td>
<td>WITI Hall of Fame, Fellow ACM, EFF Pioneer</td>
<td>has a page</td>
</tr>
<tr>
<td>rat</td>
<td><a href="http://polaris.gseis.ucla.edu/cb">http://polaris.gseis.ucla.edu/cb</a></td>
<td>ACM Fellow</td>
<td>has a page, needs work</td>
</tr>
</tbody>
</table>
To Share These Achievements....

- August 2014, with Katy Dickinson and Jessica Dickinson Goodman....
- Created Notable Women in Computing cards
Vicki Hanson
Had no Wikipedia page, now does
What happens when your hobby and your career collide?

It is now time for engaging students with edible CS
Automata Theory
Interaction in Class – Props
Edible Turing Machine

- TM for \( f(x) = 2x \) where \( x \) is unary

- TM is not correct, can you fix it? Then eat it!

- States are blueberry muffins
Students building DFA with cookies and icing
CS 1
Sorting
Cookies
CS 1 had around 300 students
Thank You

• Questions?