

CS-257L

Nonimperative Programming: Scheme!

Instructor:

Joel Castellanos

e-mail: joel@unm.edu

Web: <http://cs.unm.edu/~joel/>

Office: Farris Engineering
Center (FEC) room 321

Context-Free Languages

$S \rightarrow B$

$B \rightarrow bS \mid b$

CS-257 Lecture Cancelled,

There will not be lecture on Monday, Feb 18.
Monday's Recitation is still scheduled to meet.

UNM Graduate School Info BBQ

- When: Wed., Feb. 20th at 4 pm
- Where: FEC 141 – and the outside portch
- What:
 - Learn more about graduate school in computer science.
 - Score some free barbeque.
 - Great music – Professor Forrest vote: "2001: A Space Odyssey"

Job Opening

- Sandia National Labs
- Looking for UNM Undergrad
- Very strong Excel skills
- Visual Basic Programming

Quiz #2 Graded

```
(define Fibonacci
  (lambda (n)
    (display "x ")
    (if (or (< n 1) (< (truncate n) n)) "Bad Input"
        (case n
          ((1 2) 1)
          (else (+ (Fibonacci (- n 1))
                  (Fibonacci (- n 2)))
                )
        )
    )
  )
)
```

The output of (Fibonacci 3) is: x x x 2.

What is the outout of (Fibonacci 6)?

x x x x x x x x x x x x x x 8

5 pts: Handed in a paper with your name and work.

7 pts: Final value of 8, but incorrect number of x's.

8 pts: Correct number of x's

10 pts: Correct answer.

HW5 – DCFL-checker – Input & output

(DCFL-checker

`axiom ; a list of atoms`

`rules ; a list of lists of atoms.`

`initialAngle ;real number [0, 360]`

`turnAngle ;real number [0, 180]`

`shrinkage ;real number [0.01, 100.0]`

)

- If the arguments do not define a DCFL, then display an appropriate error message.
- If the arguments do define a DCFL, then display:
 1. An affirmation,
 2. A list of the language's variables, and
 3. A list of the language's terminals.

What should DCFL-checker Output?

```
(DCFL-checker
  (a b b a) (b b a b) (c a c) (d c b c)
  90 90 0.5
)
```

Error: Bad input: Expected 5 arguments. Given 7.

```
(DCFL-checker
  (a b b a) ((b b a b) (c a c) (d c b c))
  90 90 0.5
)
```

What should DCFL-checker Output?

```
(DCFL-checker
  (a b b a) ((b b a b) (c a c) (d c b c))
  90 90 0.5
)
```

Yes, This defines a fine DCFL
variables: (b c d)
terminals: (a)

In this grammar, c and d will never be generated. Their rules are unused and can be deleted without changing the grammar.

Find the first 3 Generations

(DCFL-checker

(a b b a) ((b c d) (c c a c) (d c b c))

90 90 0.5

)

0	a			b				b				a			
1	a		c			d			c		d	a			
2	a		c	a	c	c	b	c	c	a	c	c	b	c	a
3															

Simple Context Free Grammers

$S \rightarrow abS \mid ab$

$S \rightarrow SS \mid ab$

$S \rightarrow aB$

$B \rightarrow bS \mid a$

$S \rightarrow aB$

$B \rightarrow bS \mid ab$

$S \rightarrow aB \mid ab$

$B \rightarrow bS$

$S \rightarrow aSa \mid bSb \mid cSc \mid dSd \mid ()$

example: $aSa \rightarrow abSba \rightarrow abcScba \rightarrow abccba$

Blue-Green Algae

Can this pattern be generated by a Context-Free Grammar?

Can it be generated by a Deterministic Context-Free Grammar?

