

# Something for everyone:

**A.I. lab assignments that span  
learning styles and aptitudes**

Christopher League

CCSC/NE

12 April 2008





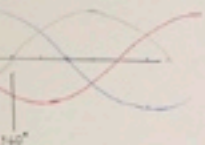
**Something for  
everyone:**



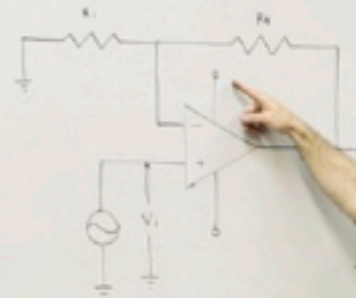








$$\cos \theta = PF$$



$$A_v = \frac{R_1 + R_2}{R_1}$$





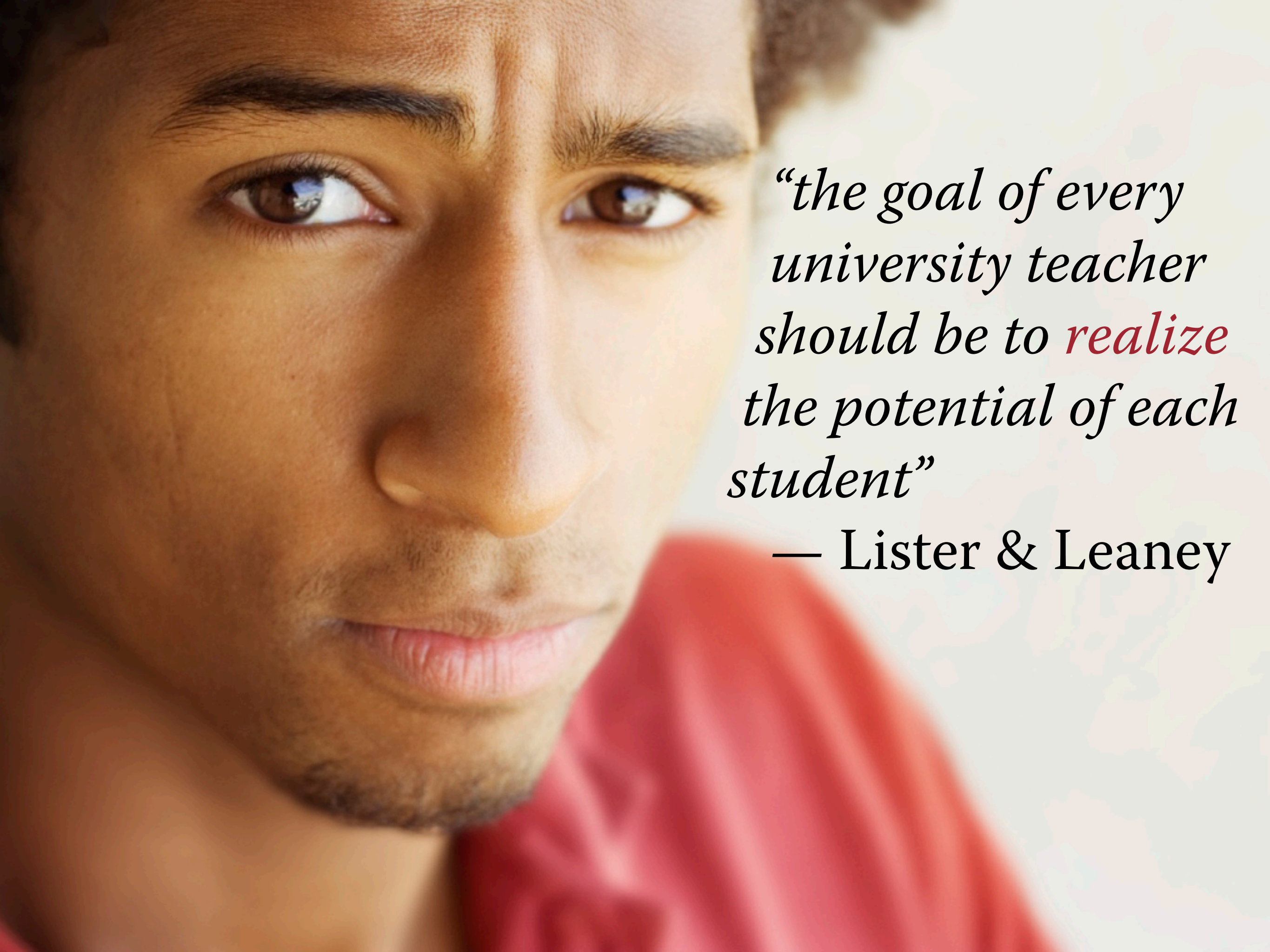
wide range of educational backgrounds,  
learning styles, aptitudes, and  
time/energy constraints











*“the goal of every  
university teacher  
should be to **realize**  
the potential of each  
student”*

— Lister & Leaney

Bloom, *Taxonomy of educational objectives*, 1956



# Bloom, *Taxonomy of educational objectives*, 1956

## Evaluation

appraise · conclude · criticize  
critique · defend · justify · support

## Synthesis

arrange · compile · compose · create · devise  
design · extend · generate · modify · plan · write

## Analysis

compare · contrast · deconstruct · differentiate  
distinguish · illustrate · infer · relate · separate

## Application

change · compute · demonstrate · discover  
operate · predict · prepare · show · solve · use

## Comprehension

convert · estimate · explain · generalize · exemplify  
infer · interpret · paraphrase · summarize · translate

## Knowledge

define · describe · identify · label · list · match · name  
outline · recall · recognize · reproduce · state







*“IT academics place  
premature  
emphasis on  
the higher levels  
of the taxonomy”  
— Lister & Leaney*





# Common LISPcraft

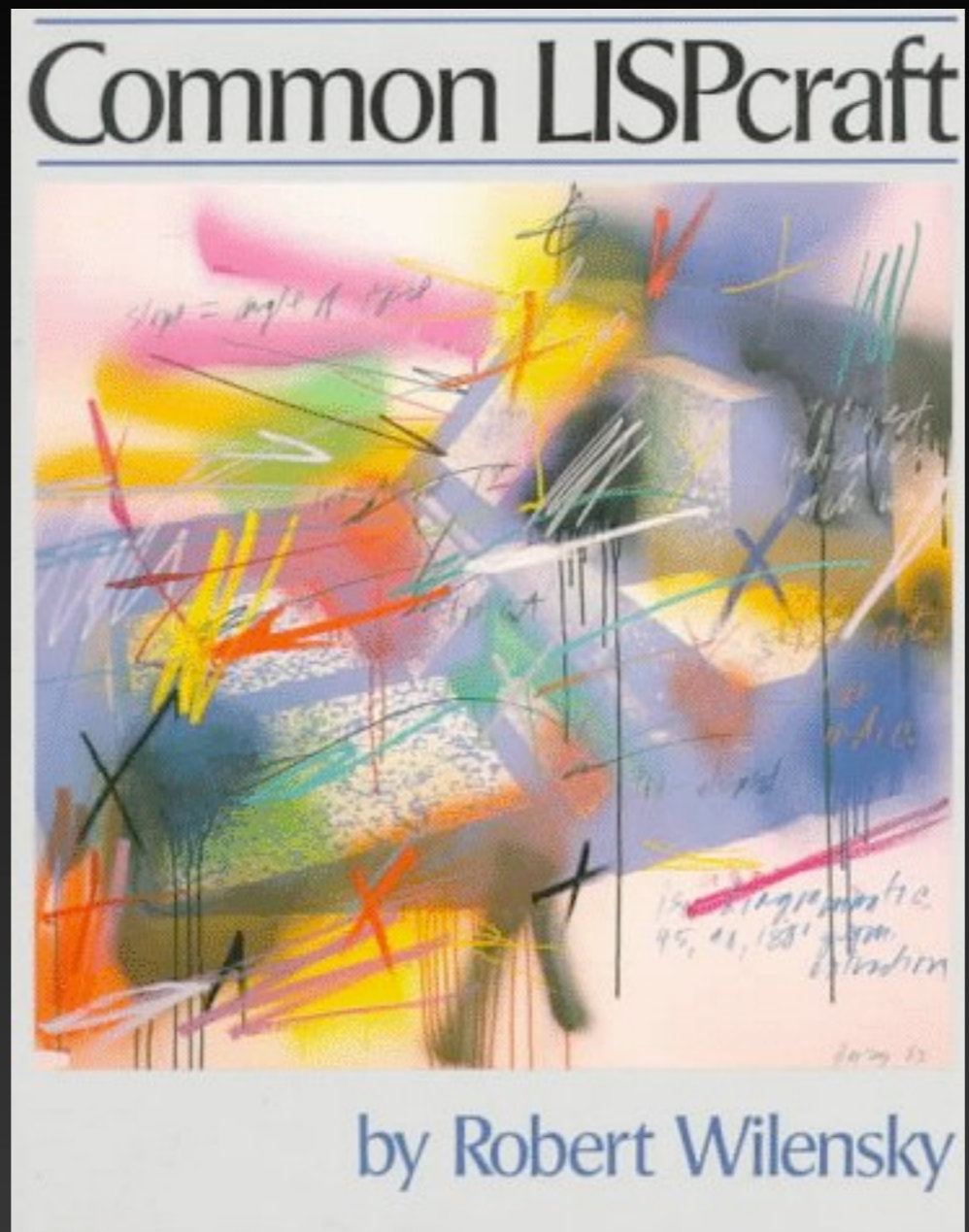


by Robert Wilensky

by Robert Wilensky



“Implement a  
constraint  
solver...



due on  
Tuesday”





**Evaluation**

**Synthesis**

**Analysis**

**Application**

**Comprehension**

**Knowledge**





**Workbook-style lab assignments that interleave lecture notes and software demos with a series of questions, tasks, and projects at multiple levels of Bloom's taxonomy**

# Topic Outline



# Topic Outline

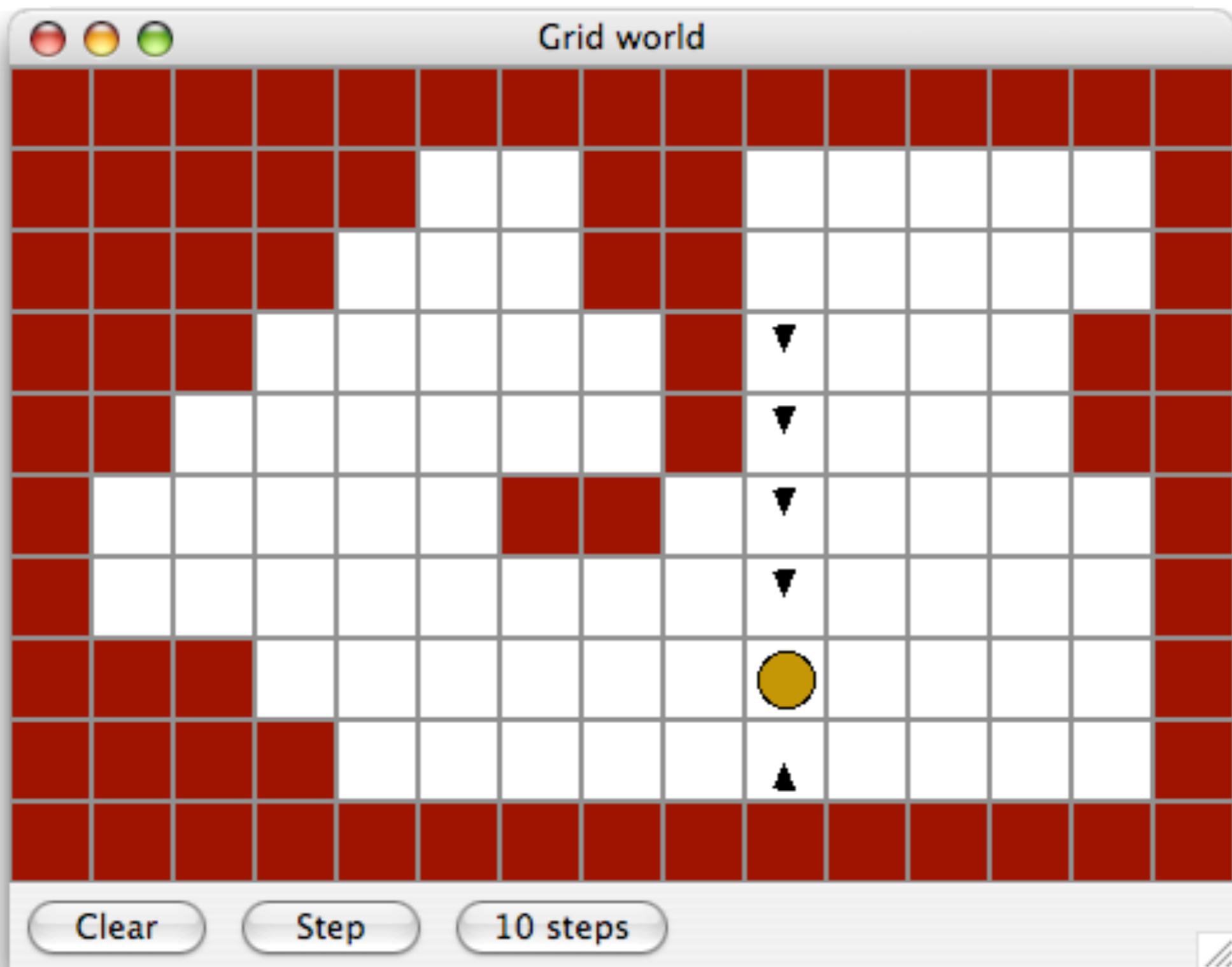
1. **Philosophical background**, strong vs. weak AI, Turing test, chat-bots
2. **Machine learning by example**: classification problems, decision trees, entropy, ID3
3. **Machine learning by evolution**: optimization problems and genetic algorithms
4. Planning using **uninformed and heuristic search**: breadth-first, depth-first, and A\* algorithm
5. **Constraint propagation** and satisfaction with AC3
6. **Adversarial search** with minimax & heuristics
7. **Knowledge representation**, logic, expert systems, common sense

# ARTIFICIAL INTELLIGENCE: *A New Synthesis*



Nils J. Nilsson











# Dr Scheme

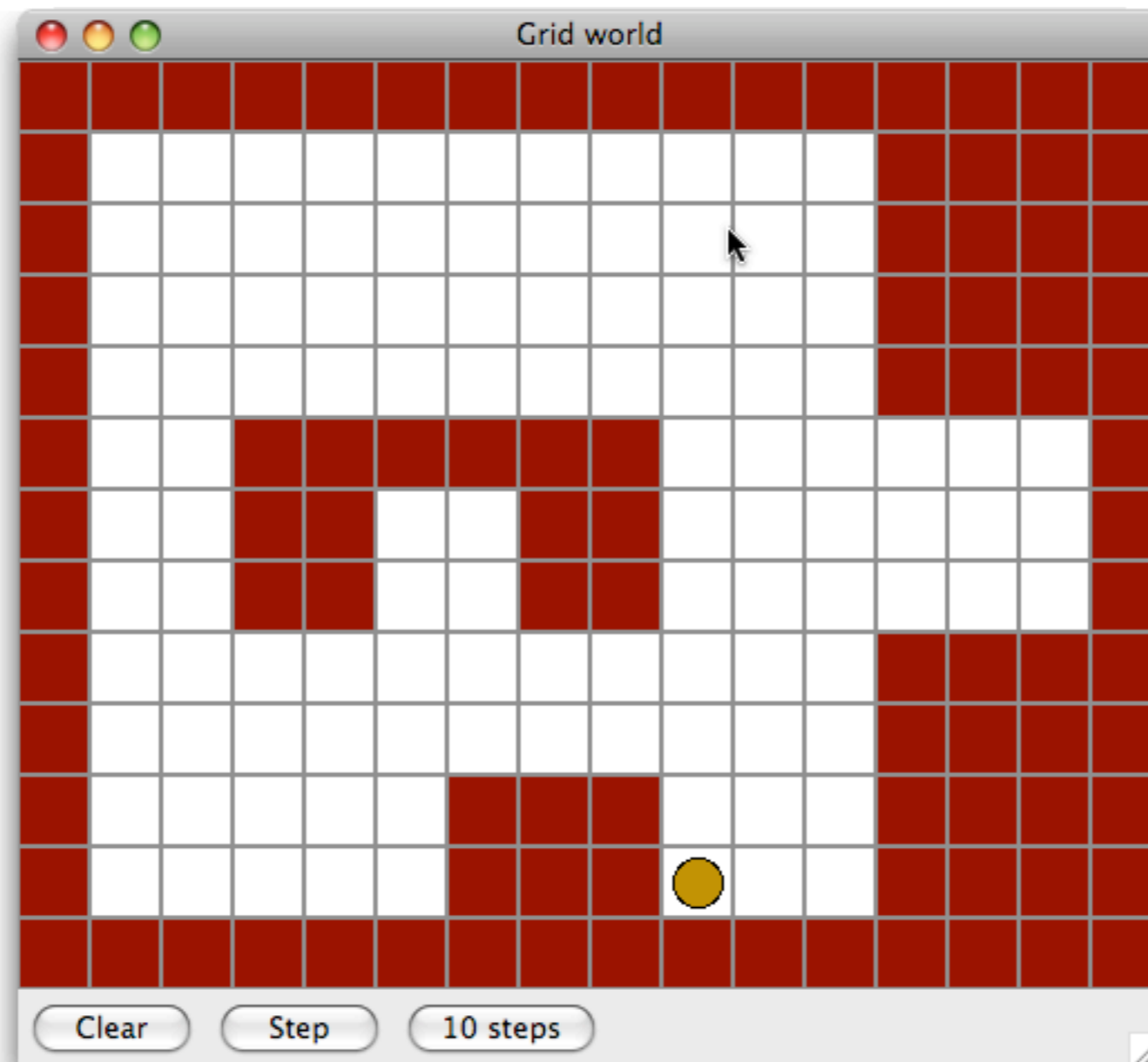






# Knowledge Comprehension

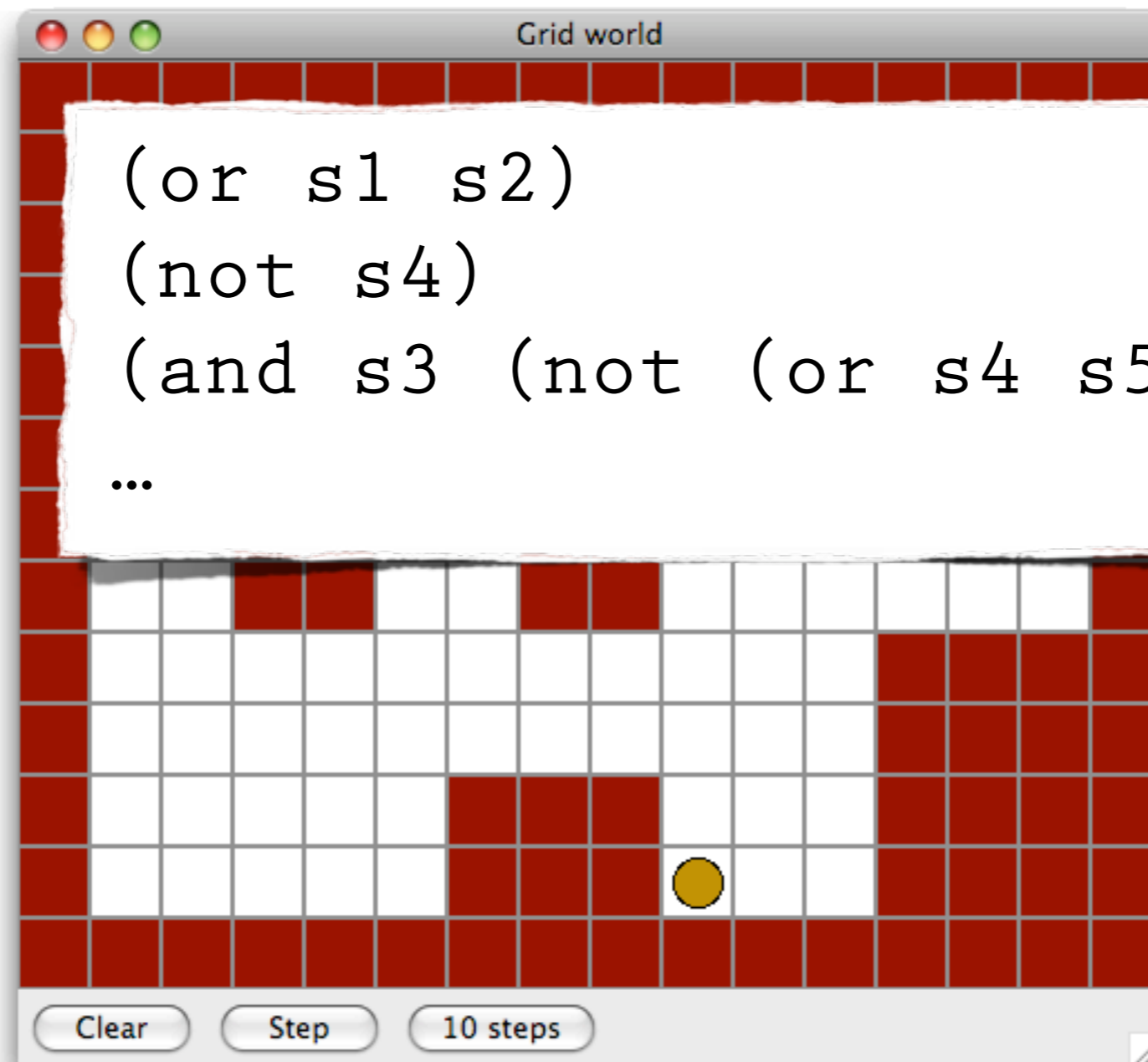
- List the values of the robot's sensors at its current location on this map:





# Comprehension Application

- Carefully compute the truth values of the following conditional expressions:



```
(or s1 s2)
(not s4)
(and s3 (not (or s4 s5)))
...
```





# Application

- Which direction will the robot attempt to go, if using this controller?

The image shows a window titled "Grid world" containing a 10x10 grid. The grid has red cells representing obstacles and white cells representing free space. A yellow circle representing a robot is located at the intersection of the 8th column and 7th row (assuming the top-left is (1,1)). A white box with a torn edge contains the following code:

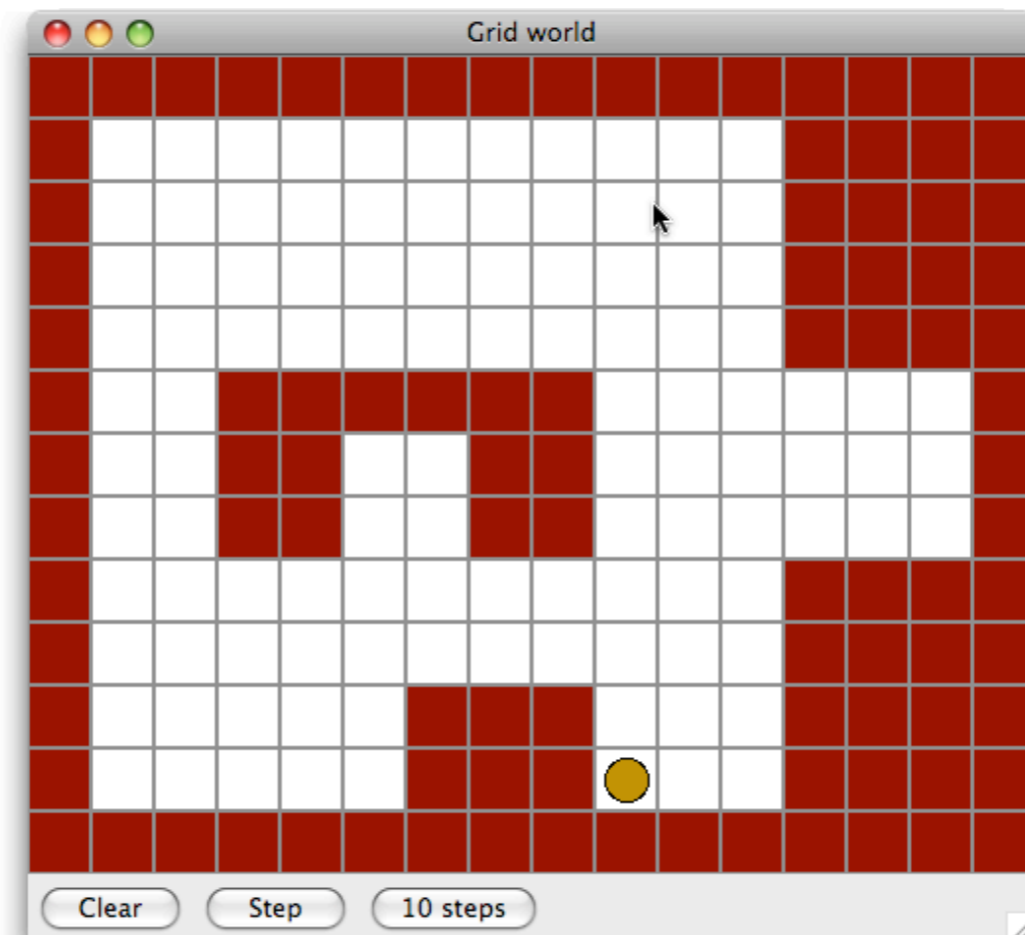
```
(define my-robot  
  '(if s6 'north 'south))
```

At the bottom of the window, there are three buttons: "Clear", "Step", and "10 steps".



# Analysis

- Mark all the squares from which your robot should move north.
  - What features distinguish those squares from all the others?

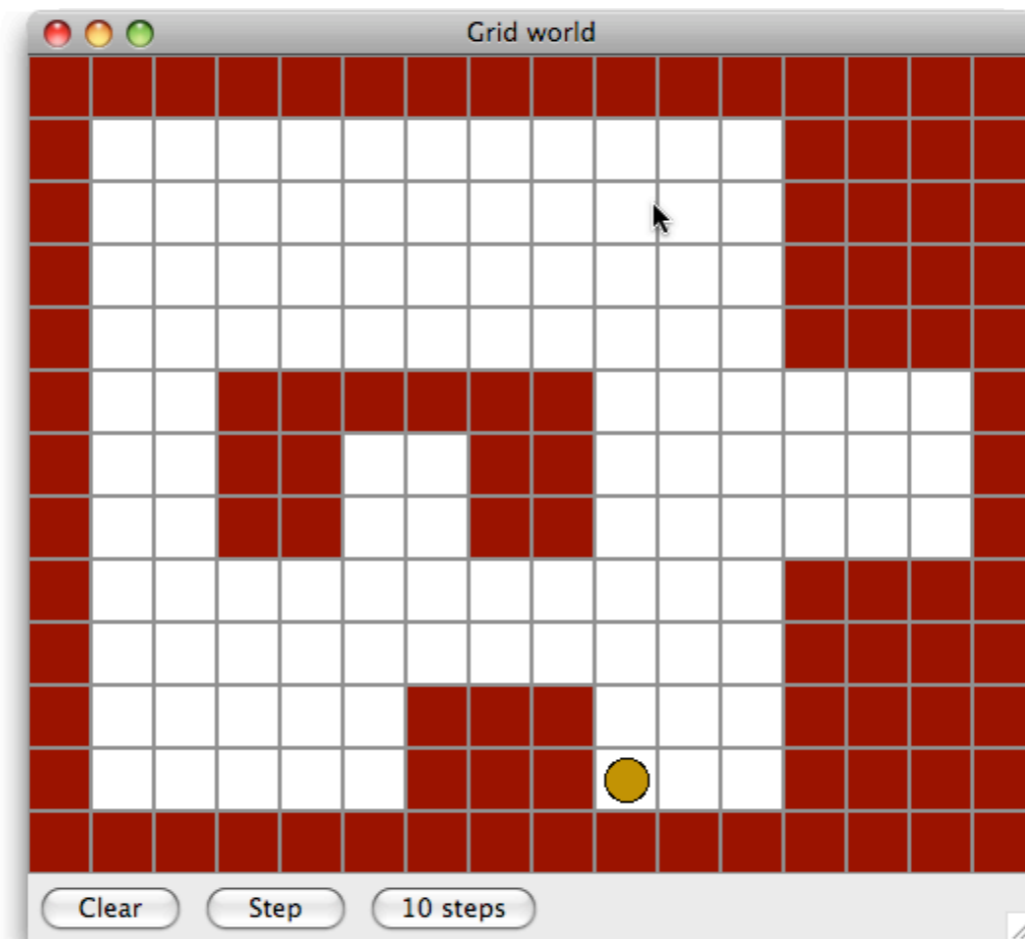


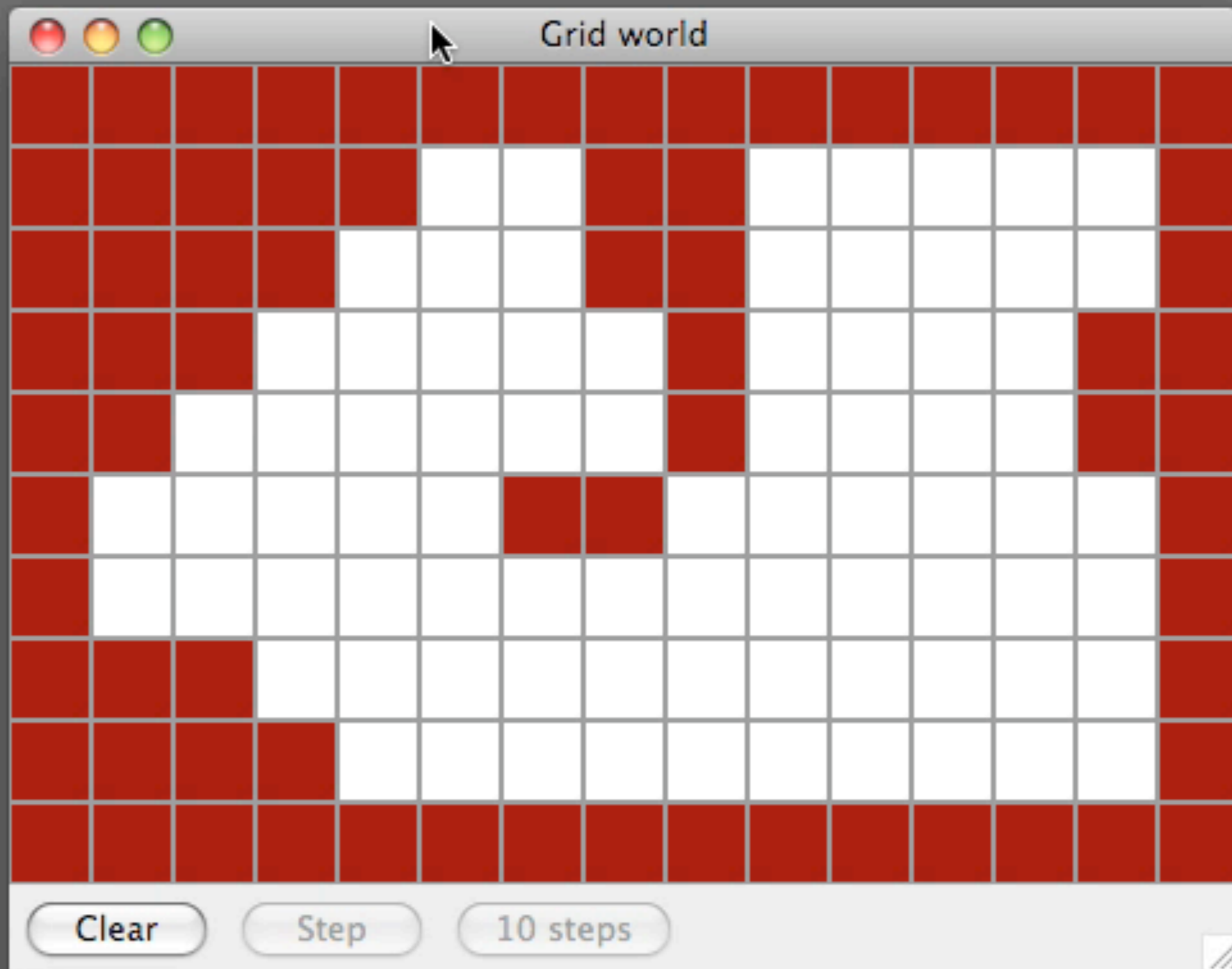




# Synthesis

- Compose and test your own robot controller





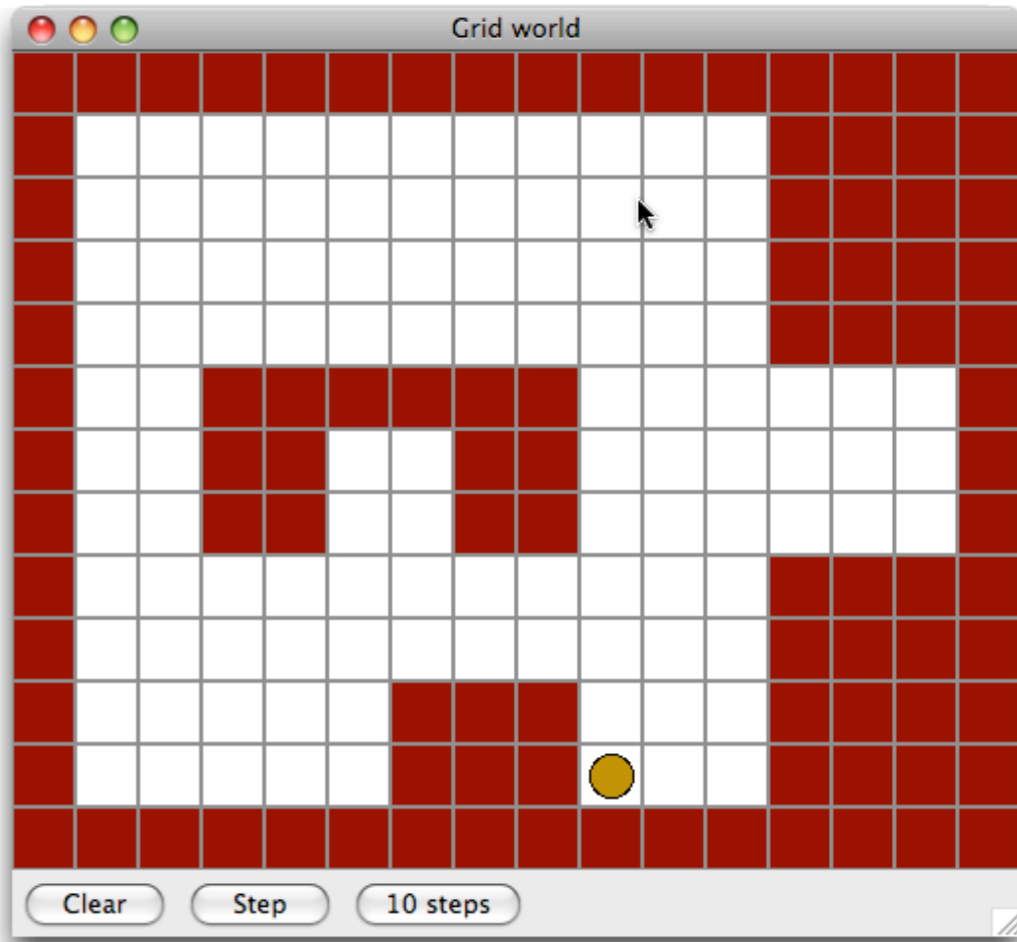




# Analysis Synthesis

- ◆ Determine how the map is specified, then design and test your own room

```
(define room-1
  ' (" xxx"
    " xxx"
    " xxx"
    " xxx"
    " xxxxxx"
    " xx  xx"
    " xx  xx"
    " xxx"
    " xxx"
    "   xxx"
    "   xxx" ) )
```





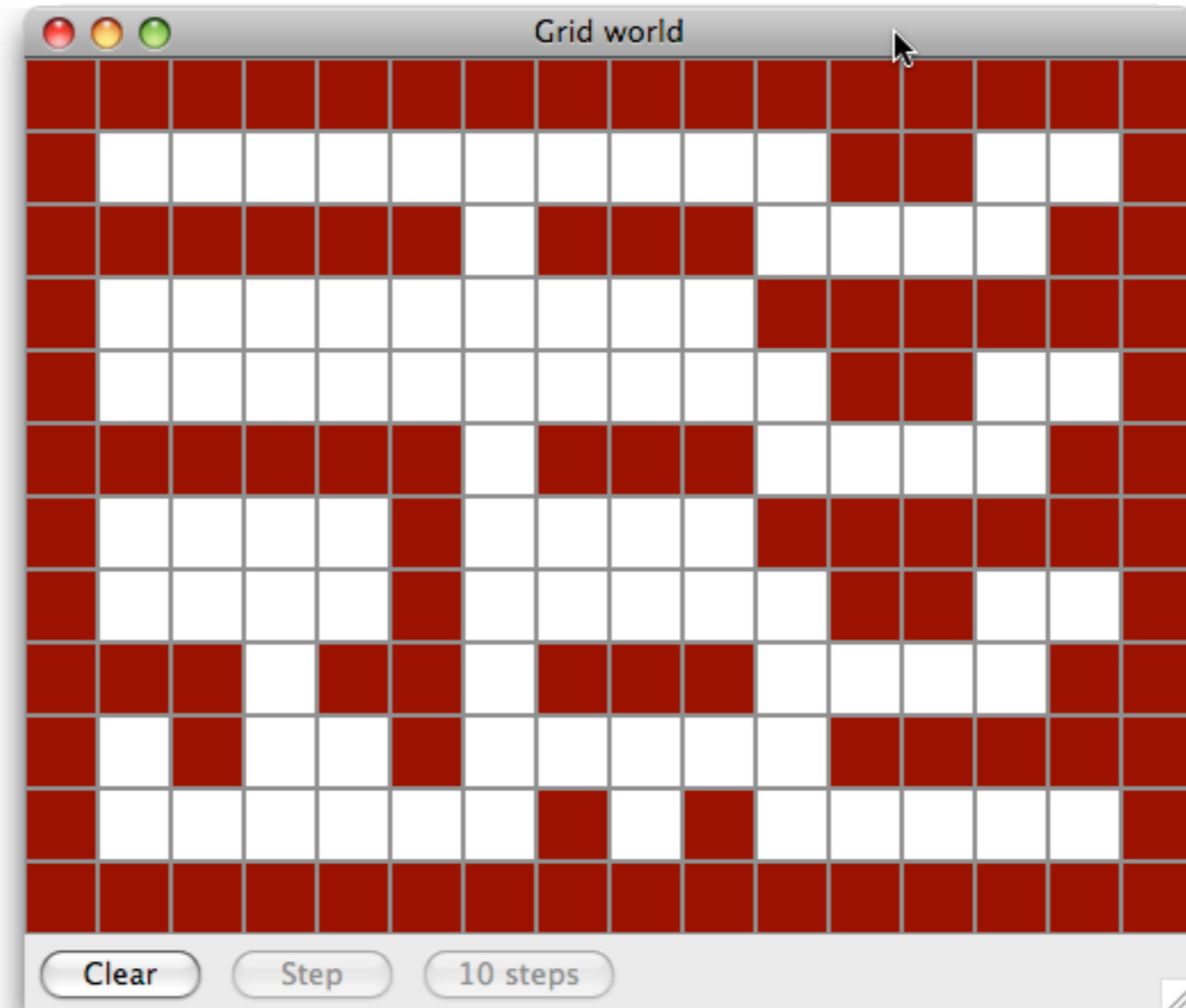
# Evaluation

- What are some limitations of a stateless stimulus/response system?



# Evaluation

- What are some limitations of a stateless stimulus/response system?





search-bot.scm - DrScheme

search-bot.scm ▾

Debug Macro Stepper Check Syntax Run Stop

```
(define ...)  
  
;; Here is the planning engine  
(define visitation-plan%  
  (class searchable%  
    (override start goal? state-  
      (init-field (state #f))  
      (define (start) (cdr (ass  
        (define (goal? p) (elemen  
        (define (state-eq? p q) (  
        (define (successors-of p)  
          (let ((r (car p))
```

Welcome to [DrScheme](#), version 371 [3m].  
Language: [Graphical \(MrEd, includes MzS](#)  
>

### Grid world

█	█	█	█	█	█	█	█	█	█	█	█	█	█	█
█	█	█	█	█	□	□	█	█	□	□	□	□	█	█
█	█	█	█	□	□	□	█	█	□	□	□	□	█	█
█	█	█	□	□	□	□	□	█	□	□	□	█	█	█
█	█	□	□	□	□	□	█	█	□	□	□	█	█	█
█	█	□	□	□	□	█	█	□	□	□	□	█	█	█
█	█	█	█	□	□	□	□	□	□	□	□	□	█	█
█	█	█	█	█	□	□	□	□	□	□	□	□	█	█
█	█	█	█	█	█	█	█	█	█	█	█	█	█	█

Clear Step 10 steps

Programming language: ▾  
Graphical (MrEd, includes MzScheme)

3:2 Read/Write



arch-bot.scm - DrScheme

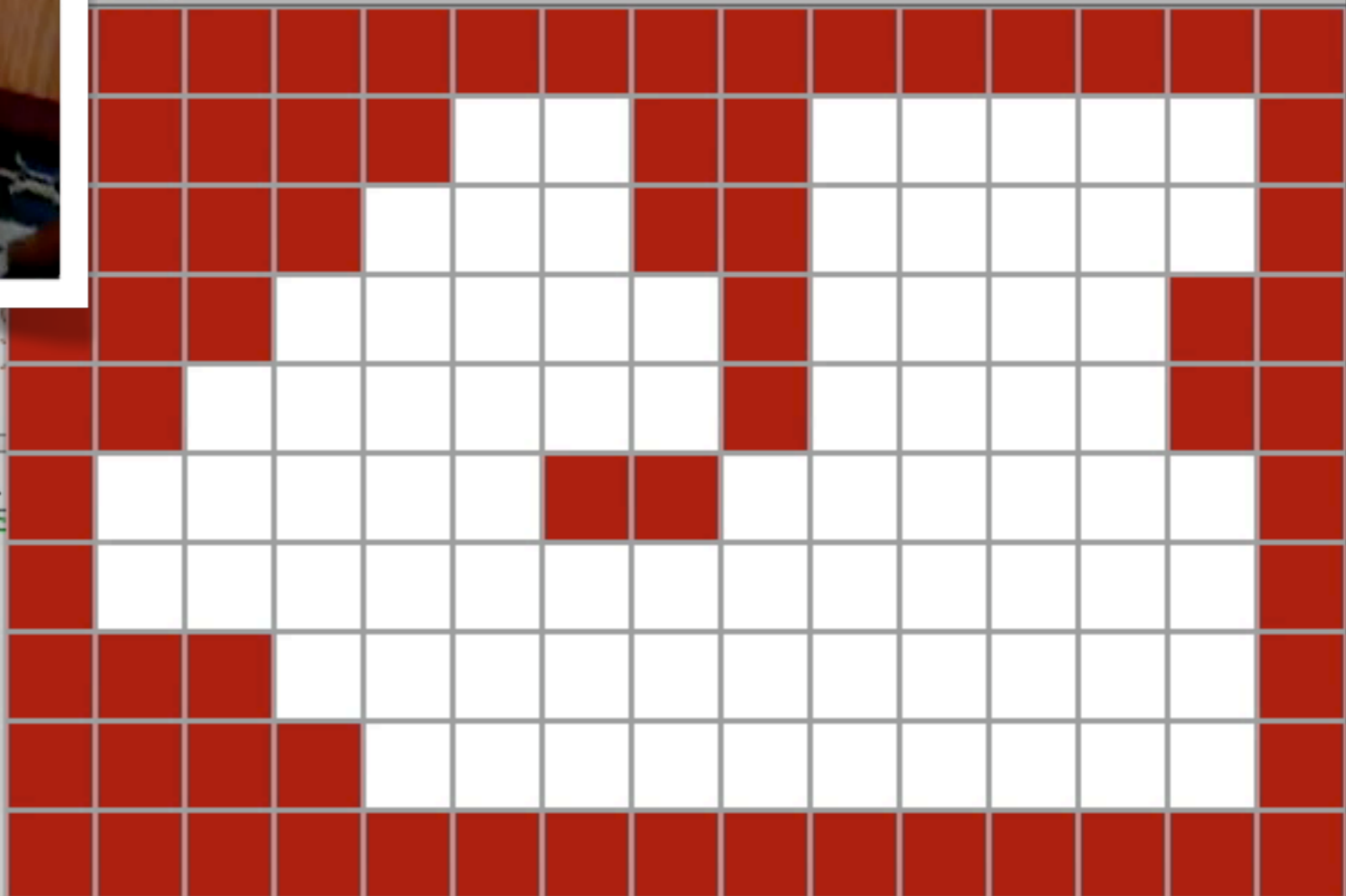
Macro Stepper

Check Syntax

Run

Stop

Grid world



Clear

Step

10 steps

```
(define (state-eq? p q)  
(define (successors-of p)  
  (let ((r (car p))
```

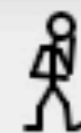
Welcome to [DrScheme](#), version 371 [3m].  
Language: [Graphical \(MrEd, includes MzS](#)  
>

Programming language: ▼

Graphical (MrEd, includes MzScheme)

3:2

Read/  
Write



# Constraint satisfaction



# Constraint satisfaction

Sudoku

1	4	•••	•••	5	•••	•••	•••	2	•••
2	8	•••	•••	7	•••	•••	4	•••	•••
3	•••	5	2	•••	9	•••	7	1	•••
4	5	•••	•••	8	7	1	•••	•••	•••
5	9	•••	7	6	•••	5	3	•••	1
6	•••	•••	•••	4	3	9	•••	•••	6
7	•••	8	6	•••	5	•••	1	4	•••
8	•••	•••	4	•••	•••	3	•••	•••	2
9	•••	1	•••	•••	•••	6	•••	•••	7

Reset

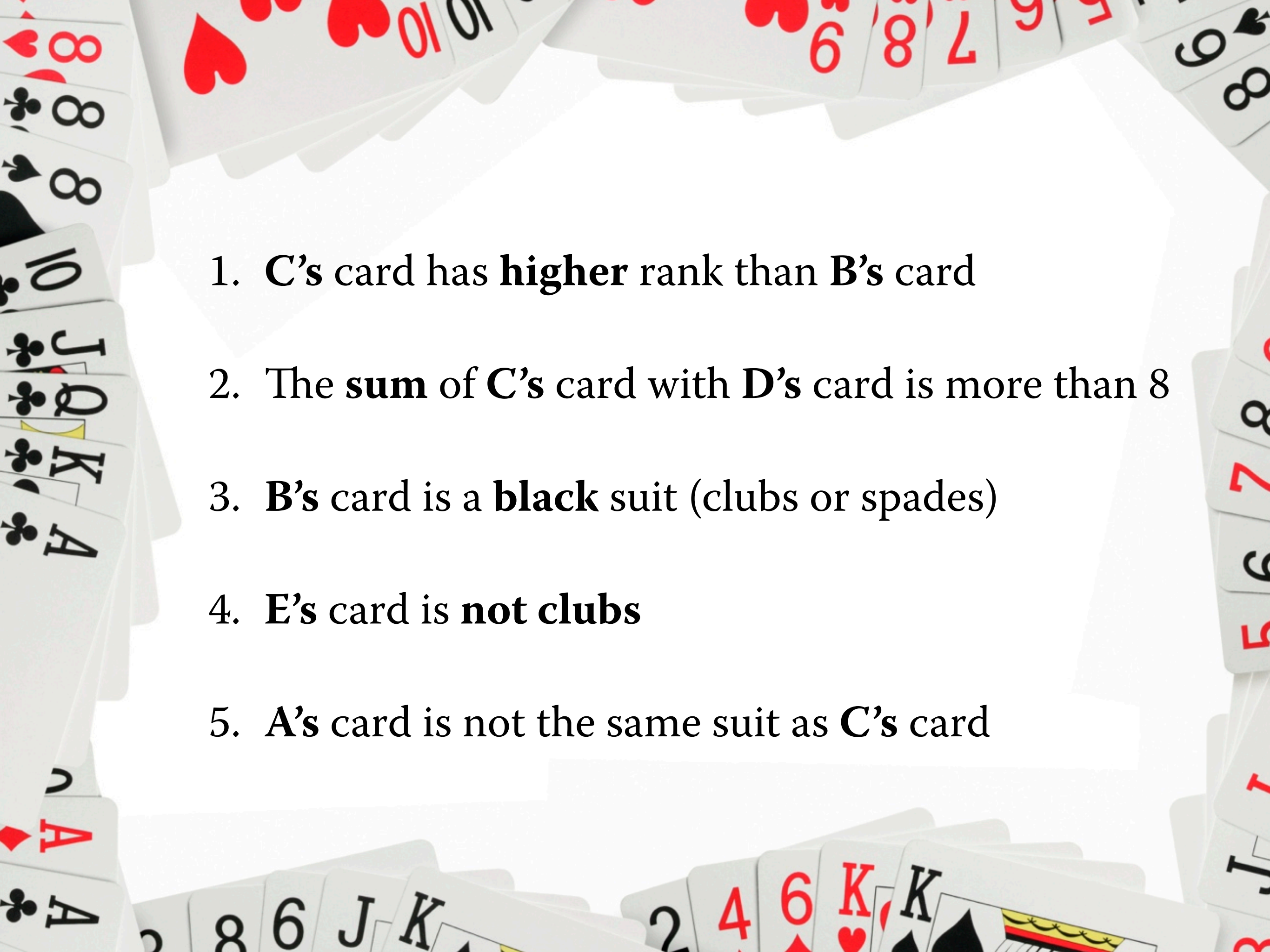
Propagate 1

Propagate all

Search step

Search

((1 . 2) 1 . 1)

- 
1. **C's** card has **higher** rank than **B's** card
  2. The **sum** of **C's** card with **D's** card is more than 8
  3. **B's** card is a **black** suit (clubs or spades)
  4. **E's** card is **not clubs**
  5. **A's** card is not the same suit as **C's** card



# Knowledge Comprehension

- Identify the unary constraints
- Identify the binary constraints





# Application

- Apply the unary constraints to the hand you were dealt
- Draw a graph showing the binary relationships



# Analysis

- How many arcs are in the graph?
- When your hand changes, which arcs are added to the work list?



# Synthesis

- Same process for 8-queens, but we follow through to implementation

The image shows a software window titled "queens" with a standard macOS-style title bar (red, yellow, and green buttons). The window contains an 8x8 grid representing an 8-queens problem. The columns are labeled 1 through 8, and the rows are labeled 1 through 8. The grid shows the following 'X' marks:

	1	2	3	4	5	6	7	8
1	X							
2			X	X	X	X	X	X
3		X		X	X	X	X	X
4		X	X		X	X	X	X
5		X	X	X		X	X	X
6		X	X	X	X		X	X
7		X	X	X	X	X		X
8		X	X	X	X	X	X	

To the right of the grid are five buttons: "Reset", "Propagate 1", "Propagate all" (with a mouse cursor over it), "Search step", and "Search". At the bottom right of the window, the text "satisfiable" is displayed.



# Topic Outline

# Topic Outline

1. **Philosophical background**, strong vs. weak AI, Turing test, chat-bots
2. **Machine learning by example**: classification problems, decision trees, entropy, ID3
3. **Machine learning by evolution**: optimization problems and genetic algorithms
4. Planning using **uninformed and heuristic search**: breadth-first, depth-first, and A\* algorithm
5. **Constraint propagation** and satisfaction with AC3
6. **Adversarial search** with minimax & heuristics
7. **Knowledge representation**, logic, expert systems, common sense

# Connect 4

The image shows a DrScheme window titled "c4minimax.scm - DrScheme" with a "Connect 4" sub-window overlaid. The sub-window displays a 7x7 Connect 4 board with red and blue pieces and numbered circles (1-17). The board state is as follows:

9						
7						
6						
5	17	15				
3	16	13		8		14
1	11	4		2	12	10

The sub-window also features "Clear" and "Undo" buttons, a score of "999", and a player selection dropdown showing "H2" vs. "human".

The background DrScheme window shows the following Scheme code:

```
(define ...)
;; Is there space for another piece at the
(define (space-in-vec? v)
  (= 0 (vector-last v)))
;; Drop a piece into the end of this vect
(define (drop-in-vec player v)
  (do ((i (- (vector-length v) 1) (- i 1)
          ((or (< i 0)
              (not (= 0 (vector-ref v i))))
        (vector-replace v (+ i 1) player))))))
;; Is there space for another piece in co
(define (space-in-column? board c)
  (space-in-vec? (vector-ref board c)))
;; Drop a piece into the top of column #c
(define (drop-in-column player board c)
```

The bottom of the DrScheme window shows the programming language set to "Graphical (MrEd, includes MzScheme)".







- ◆ Results: seems to work, more students submitting than usual

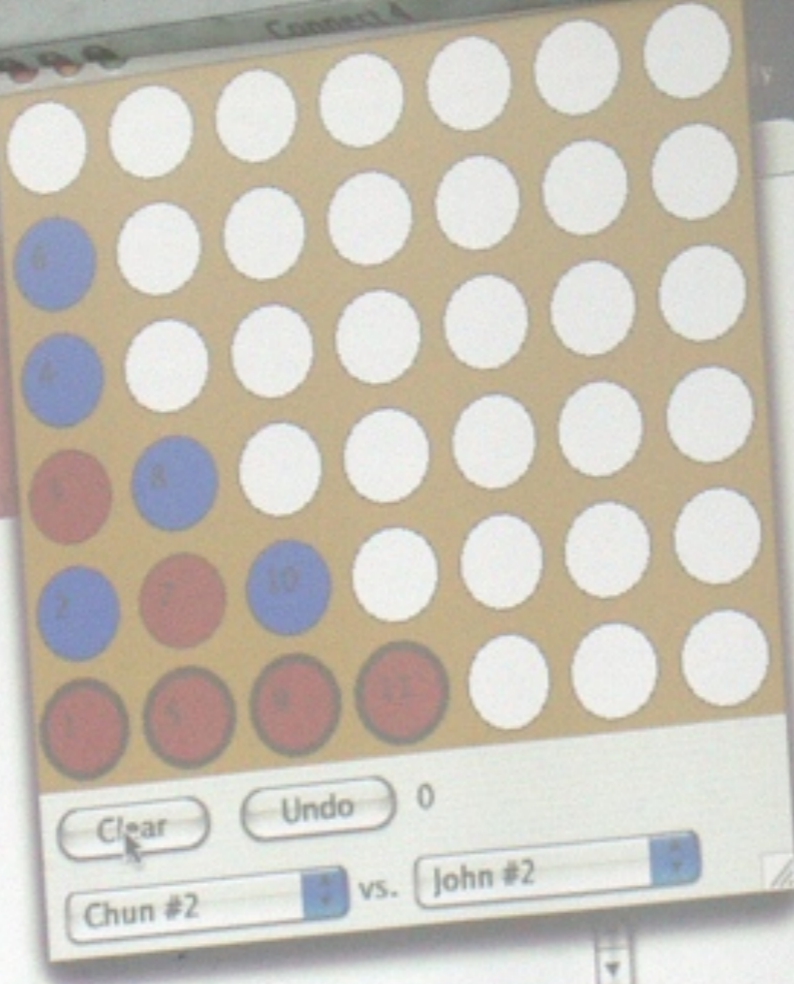
- ◆ “☆☆☆☆☆”



```

((and (= v z) (= x 0) (= y 0) (= z 2)) -9999)
((and (= v 2) (= x 2) (= y 2) (= z 2)) -9999)
(define (cl-column-score c)
  (+ (cl-quad-score (vr c 0) (vr c 1) (vr c 2) (vr c 3))
     (cl-quad-score (vr c 1) (vr c 2) (vr c 3) (vr c 4))
     (cl-quad-score (vr c 2) (vr c 3) (vr c 4) (vr c 5))))
(define (cl-board-score b)
  (+ (cl-column-score (vr b 0)) (cl-column-score (vr b 1))
     (cl-column-score (vr b 2)) (cl-column-score (vr b 3))
     (cl-column-score (vr b 4)) (cl-column-score (vr b 5))))
  (cl-row-score (b 0) (cl-row-score (b 1)
  (cl-row-score (b 2) (cl-row-score (b 3)
  (cl-row-score (b 4) (cl-row-score (b 5))))))
(define (cl-horiz-from b c r)
  (cl-horiz-quad-score (get b c r)
    (get b (+ c 1) r)
    (get b (+ c 2) r)
    (get b (+ c 3) r)))
(define (cl-row-score b r)
  (+ (cl-horiz-from b 0 r)
     (cl-horiz-from b 1 r)
     (cl-horiz-from b 2 r)
     (cl-horiz-from b 3 r)))
(define (cl-horizontal-quad-score)
  (cond
    ((and (= v 1) (= x 0) (= y 0) (= z 0)) 1)

```



0

vs.

Welcome to DrScheme, version 352.  
 Language: Graphical (MzEd, includes MzScheme) custom.

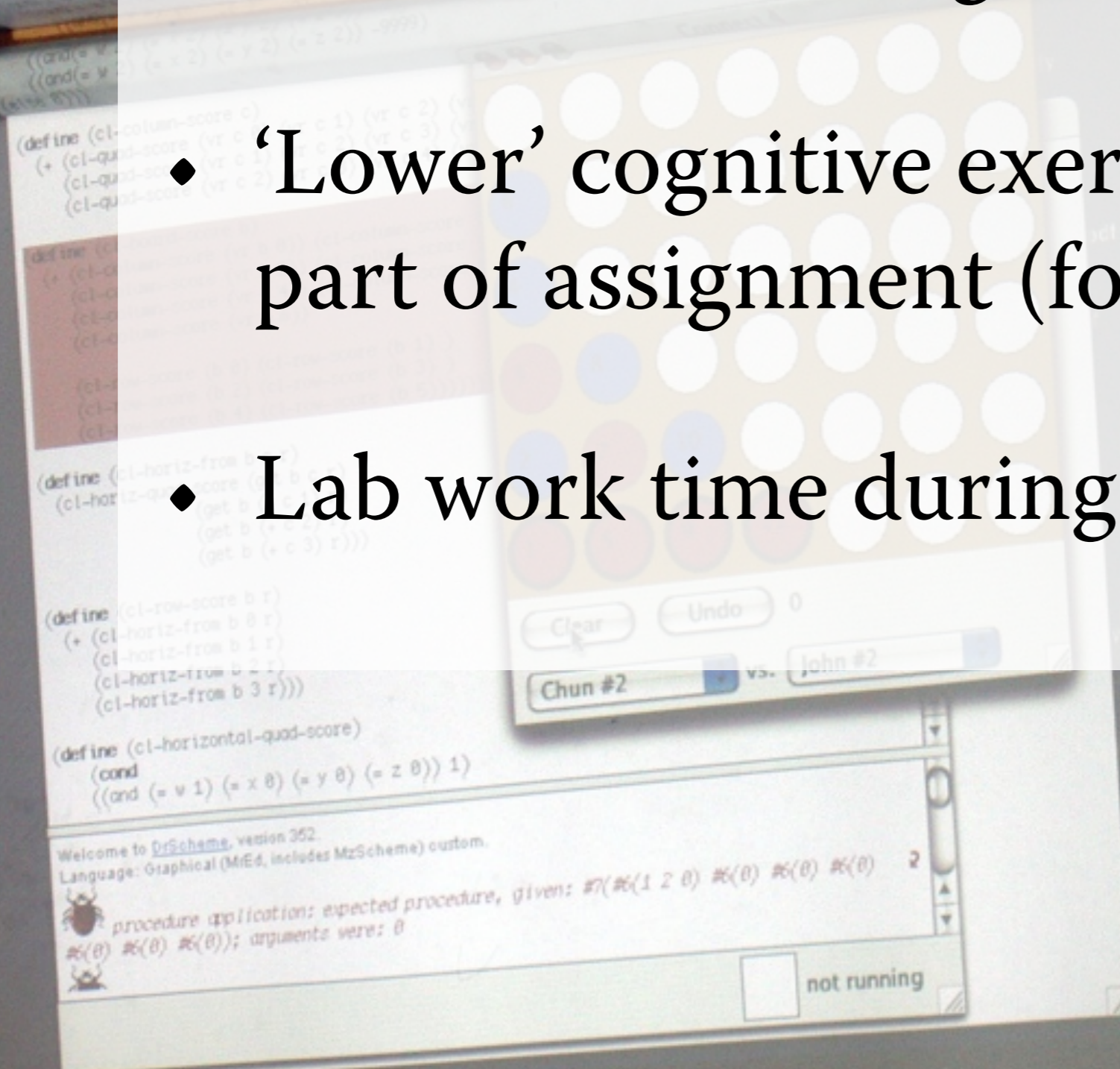
procedure application: expected procedure, given: #f(#(1 2 0) #(0) #(0) #(0)) 2  
 #(0) #(0) #(0)); arguments were: 0

not running

Adam V  
 | Chun Vs  
 Paul Vs  
 Imtiaz



- ◆ Start with working software
- ◆ ‘Lower’ cognitive exercises explicitly part of assignment (for credit)
- ◆ Lab work time during class hours



Adam V  
Chun Vs  
Paul Vs  
Imtiaz



http://contrapunctus.net/sail/

 Search

logged in as league | [Logout](#) | [Preferences](#) | [Help/Guide](#) | [About Trac](#)

	<a href="#">Wiki</a>	<a href="#">Timeline</a>	<a href="#">Roadmap</a>	<b><a href="#">Browse Source</a></b>	<a href="#">View Tickets</a>	<a href="#">New Ticket</a>	<a href="#">Search</a>	<a href="#">Admin</a>
--	----------------------	--------------------------	-------------------------	--------------------------------------	------------------------------	----------------------------	------------------------	-----------------------

[Last Change](#) | [Revision Log](#)

root / **trunk**

Visit:  View revision:

Name ▲	Size	Rev	Age	Last Change
<a href="#">../</a>				
<a href="#">ac3algo.scm</a>	7.4 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">adventure.scm</a>	7.0 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">connect4-brain.scm</a>	9.2 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">connect4-model.scm</a>	6.6 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">connect4-view.scm</a>	5.3 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">connect4.scm</a>	9.7 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">gene-algo.scm</a>	5.1 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">gene-bot.scm</a>	2.0 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">gene-knapsack.scm</a>	2.1 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">gene-prog.scm</a>	6.7 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">gene-tree.scm</a>	4.0 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">gene-vec.scm</a>	1.5 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">grid-main.scm</a>	122 bytes	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point
<a href="#">grid-model.scm</a>	8.5 kB	4	<a href="#">10 minutes</a>	league: copy code from Fall 2007 tag into trunk, as starting point

<http://contrapunctus.net/sail/>



Christopher League

CCSC/NE

12 April 2008

