

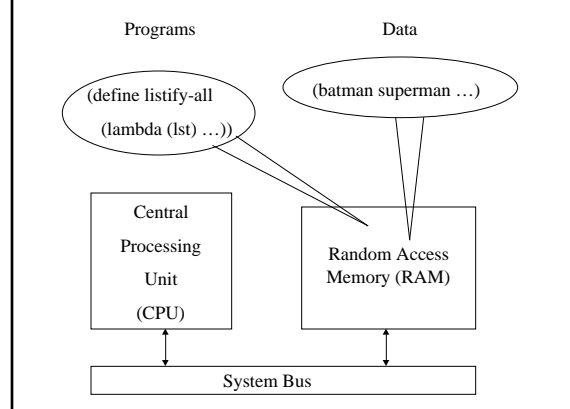
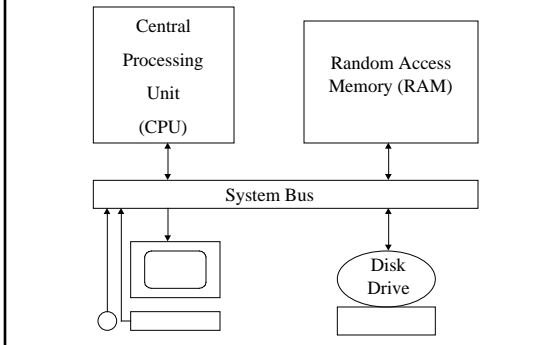
# Computer Science I

Professor Tom Ellman  
Lecture 8

## You may have been wondering...

- You've heard of bits and bytes.
- I have been talking about symbols and lists.
- What's the connection?
- How are lists actually stored in the memory of a computer?

## Von Neumann Architecture



## Organization of RAM

Address	Contents
0	
1	
2	
3	
	⋮
$2^n-1$	

- Memory is divided into storage locations.
- Each location holds some data called its "contents".
- Each location has a label called its "address".
- Given the address, one can use it to find the location and get the contents.

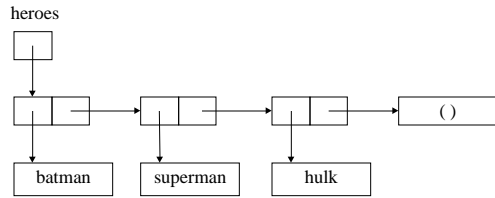
## Organization of RAM

Address	Contents
0	
1	5
2	
3	
4	2
5	
	⋮
$2^n-1$	

- The address of one location may be stored as the contents of another.
- The first location is said to "point" to the second.
- We represent a pointer visually with an arrow from one location to another.

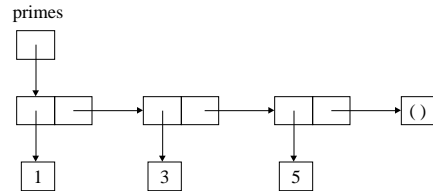
## Representation of Lists

(define heroes '(batman superman hulk))



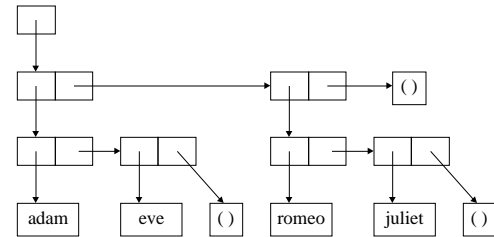
## Representation of Lists

(define primes '(1 3 5))



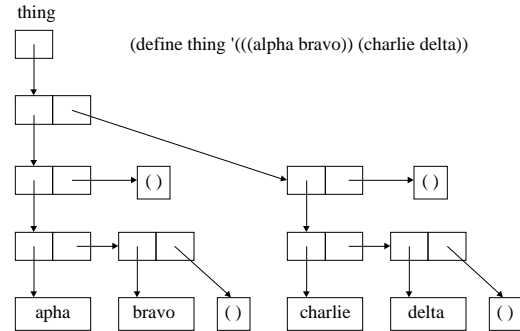
## Representation of Nested Lists

(define couples '((adam eve) (romeo juliet)))



## Representation of Nested Lists

(define thing '((alpha bravo) (charlie delta)))



## CONS, CAR & CDR

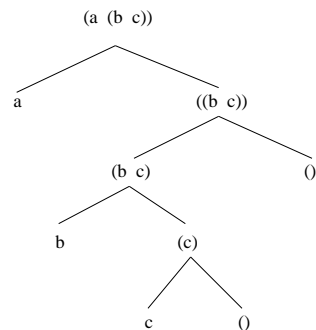
A "CONS" cell: 

CAR	CDR
-----	-----

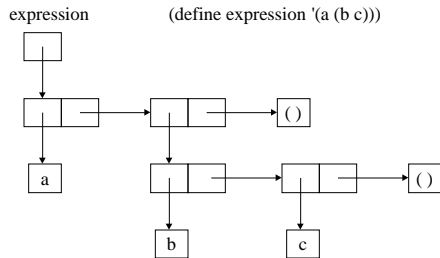
A block of RAM divided into two parts:

- One part is the "CAR".
- The other part is the "CDR".

## Splitting (a (b c)) into CARS & CDRS



## CONS Cell Representation of (a (b c))



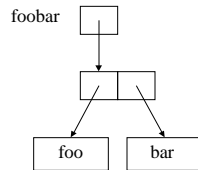
## CONS Cells and Deep Recursion

- Compare the CONS cell representation of (a (b c)) to the tree we used to analyze the ITEM-COUNT procedure on (a (b c)).
- They are both trees, and the trees have the same structure.
- In deep recursion, the recursive procedure is called once on each CONS cell.
- Recursion stops at the data that are not CONS cells.
- I.e., data that does not satisfy the PAIR? predicate.

## What is this?

```
>>> (define foobar (cons 'foo 'bar))
foobar
>>> foobar
(foo . bar)
```

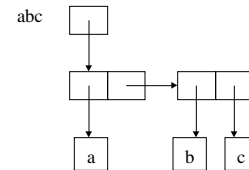
A "dotted pair".



## What is this?

```
>>> (define abc (cons 'a (cons 'b 'c)))
abc
>>> abc
(a b . c)
```

An "improper list".



## Scheme's Symbol Table

