CMSC 430 - 15 June 2026

Array data; function definitions and calls

- Exam review?
- Hoax - Vectors and strings
- Iniquity - Function definitions and calls

CMSC 430 - 15 June 2026

Announcements

- Exam 1 not yet graded, but will be soon
- Assignment 3 grades released later today
- Issue with Assignment 5 autograder, will be fixed after class
- Quiz released after class, due by start of class tomorrow
- Reminder: Assignment 4 is due tonight
- Reminder: Friday is a holiday

Hints for Assignment 5

Some thoughts on the assignment

- Determining if something is a list requires traversing it
- Computing the length of a list requires traversing it
- Temporal and spatial ordering of memory reads/writes can be arbitrary
  - You can write the elements starting from the end
  - You can write the length last, etc.
- Working examples out on paper will get you there faster
  - Draw the memory you expect
- Using static data to make examples that you interactively develop can help
- All of the primitives can be implemented in a single pass over input and without allocating temporary memory (i.e. only allocating the output value)
- Reference solution < 90 LOC
- Programming in assembly sucks, someone should develop a high-level… oh

Hoax

Hoax: Adding vectors

Concrete examples

```
(make-vector 3 #t)                  ;=> '#(#t #t #t)
(let ((v (make-vector 3 #t)))
  (vector-ref v 0))                 ;=> #t
(let ((v (make-vector 3 #t)))
  (vector-ref v 3))                 ;=> err
(let ((v (make-vector 3 #t)))
  (vector-length v))                ;=> 3
(let ((v (make-vector 3 #t)))
  (begin (vector-set! v 0 #f)
         v))                        ;=> '#(#f #t #t)
```

Hoax: Adding vectors

AST additions

```
;; type Op1 = ...
;;          | 'vector? | 'vector-length
;; type Op2 = ...
;;          | 'make-vector | 'vector-ref
;; type Op3 = 'vector-set!
```

Hoax: Adding vectors

Semantic additions

```
;; type Value = ...
;; | (vector Value ...)
```

Hoax: Adding vectors

Semantic additions

```
;; Op2 Value Value -> Value { raises 'err }
(define (interp-prim2 op v1 v2)
  (match (list op v1 v2)
    #;...
    [(list 'make-vector (? integer?) _)
     (if (<= 0 v1)
         (make-vector v1 v2)
         (raise 'err))]
    [(list 'vector-ref (? vector?) (? integer?))
     (if (<= 0 v2 (sub1 (vector-length v1)))
         (vector-ref v1 v2)
         (raise 'err))]))
```

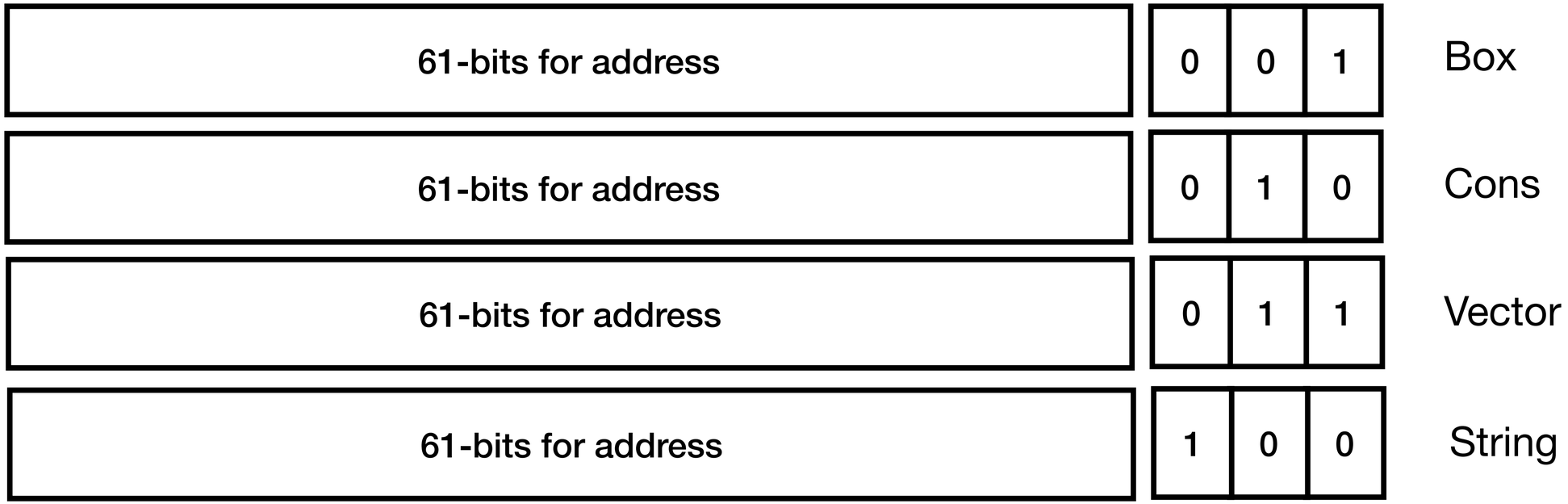
Hoax: Adding vectors

Semantic additions

```
;; Op3 Value Value Value -> Value { raises 'err }
(define (interp-prim3 p v1 v2 v3)
  (match (list p v1 v2 v3)
    [(list 'vector-set! (? vector?) (? integer?) _)
     (if (<= 0 v2 (sub1 (vector-length v1)))
         (vector-set! v1 v2 v3)
         (raise 'err))]
    [_ (raise 'err)]))
```

Encoding pointer values (Hoax)

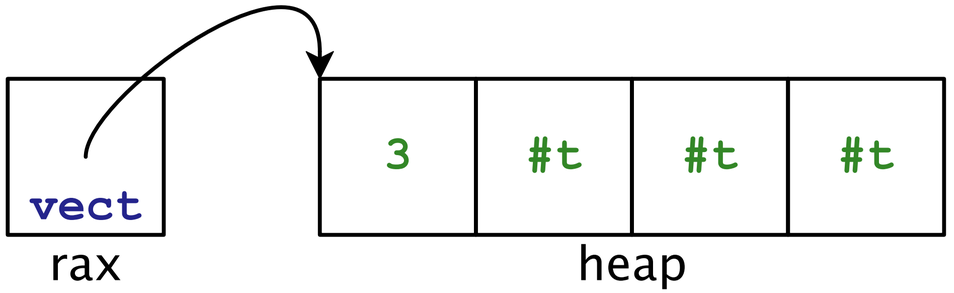
Type tag in least significant bits



Creating a vector

A vector is a tagged pointer to a sized array of values in memory

```
> (exec (parse '(make-vector 3 #t)))
'#(#t #t #t)
```



For next time

- Read Hoax notes
- Take ELMS quiz
- Submit assignment 4 tonight