

## Representing Sets as Lists

We represent the set  $\{e_1, e_2, \dots, e_n\}$   
as the list  $(e_1\ e_2\ \dots\ e_n)$ .

The elements are arbitrary expressions.

```
(define member (lambda (e s)
  (cond ((null? s) #f)
        ((equal? e (car s)) #t)
        (#t (member e (cdr s))) ) ))
```

```
(define subset (lambda (s1 s2)
  (cond ((null? s1) #t)
        (#t (and (member (car s1) s2)
                  (subset (cdr s1) s2) ) ) ))
```

```
(define diff (lambda (a b)
  (cond ((null? a) ())
        ((member (car a) b) (diff (cdr a) b))
        (#t (cons (car a) (diff (cdr a) b))) ) ))
```