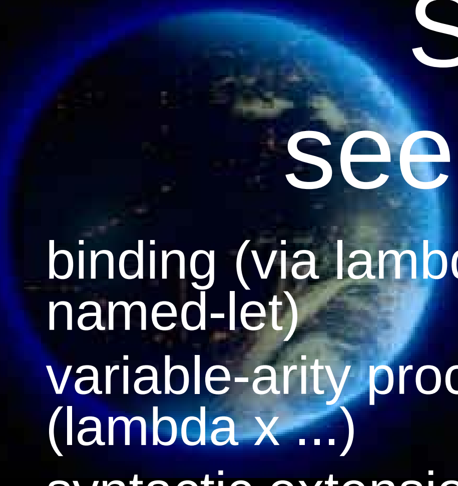
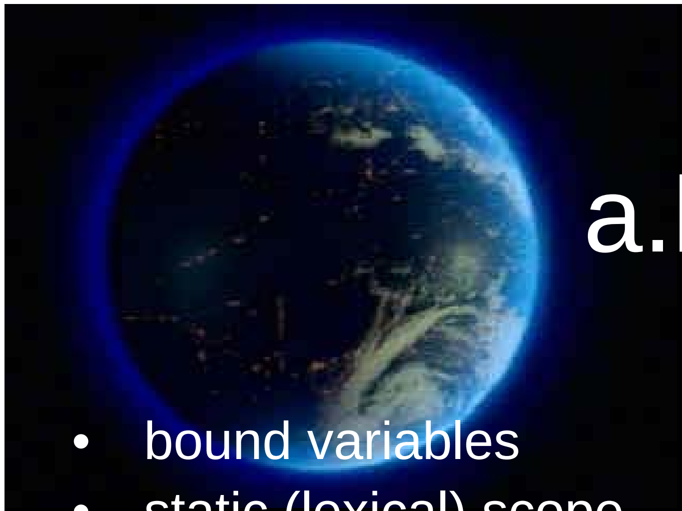
Some PL Concepts seen in the course so far



- binding (via lambda, let, letrec, named-let)
- variable-arity procedure interface (lambda x ...)
- syntactic extension (e.g., let can be written in terms of lambda)
- procedure vs syntactic form
- predicate
- application of a procedure
- higher-order procedure
- read-eval-print-loop
- lambda-calculus expressions
- free variables
- syntactic extension
- abstract datatypes
- representation-independent code
- mapping a procedure over a list
- dynamic vs static typing of variables.
- vectors *vs* lists
- sublist-sharing
- anonymous procedure
- first-class procedure
- currying
- short-circuit evaluation
- Backus-Naur Form (BNF), a.k.a. context-free gramars
- Kleene star and plus
- syntactic derivation from a BNF grammar
- multi-value return
- receivers



Some PL Concepts a.k.a. final exam review list

- bound variables
- static (lexical) scope
- lexical distance (a.k.a.lexical address)
- environment
- closure
- abstract syntax vs. concrete syntax
- parsing
- functional programming (values of variables never change)
- mutation
- tail-recursion
- variant records
- memoization
- procedural abstraction (list-recur, bt-recur, snlist-recur, etc.)
- parser
- interpreter
- recursive environment extension
- escape procedure
- continuation
- continuation: DS representation
- continuation-passing style
- call/cc
- iterator
- imperative form
- engine
- coroutine
- abstraction
- representation-independent