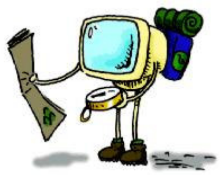
Programming Languages

Tevfik Koşar

Lecture - VII
February 7th, 2006

Roadmap

- Names
- Scopes
- Binding
  - Binding Times
  - Static vs Dynamic Binding
  - Object Lifetime & Storage Management



2

Name, Scope, and Binding

- A name is exactly what you think it is
  - Most names are identifiers
    - Constants, variables, functions
  - symbols (like '+') can also be names
- A binding is an association between two things, such as a name and the thing it names
- The scope of a binding is the part of the program (textually)in which the binding is active

3

Binding Time

- Binding Time is the point at which a binding is created or, more generally, the point at which any implementation decision is made
  - language design time
    - program structure, control flow, possible types
  - language implementation time
    - Coupling of I/O to OS
    - arithmetic overflow
    - Maximum sizes of stack and heap
    - Precision of the (number of bits) of fundamental types

4

Binding Time

- Implementation decisions (continued):
  - program writing time
    - algorithms, names
  - compile time
    - Mapping of high level code to machine language
  - link time
    - layout of whole program in memory
      - A name in one module refers to an object in another module
  - load time
    - OS loads the program into memory before running
    - choice of physical addresses

5

Binding Time

- Implementation decisions (continued):
  - run time
    - value/variable bindings, sizes of strings
    - subsumes
      - program start-up time
      - module entry time
      - elaboration time (point at which a declaration is first "seen")
      - procedure entry time
      - block entry time
      - statement execution time

6

Static vs Dynamic Binding

- The terms STATIC and DYNAMIC are generally used to refer to things bound before run time and at run time, respectively
  - "static" → binding before run time
  - "dynamic" → binding at run time

7

Binding

- In general, early binding times are associated with greater efficiency
- Later binding times are associated with greater flexibility
- Compiled languages tend to have early binding times
- Interpreted languages tend to have later binding times

8

Object Lifetime

- Key events
  - creation of objects
  - creation of bindings
  - references to variables (which use bindings)
  - (temporary) deactivation of bindings
  - reactivation of bindings
  - destruction of bindings
  - destruction of objects

Object Lifetime

- The period of time from creation to destruction is called the LIFETIME of a binding
  - If object outlives binding it's garbage
  - If binding outlives object it's a dangling reference
- The textual region of the program in which the binding is *active* is its scope
- In addition to talking about the *scope of a binding*, we sometimes use the word *scope* as a noun all by itself, without an indirect object

Storage Management

- Object lifetimes correspond to one of three storage allocation mechanisms:
  - Static:
    - absolute address retained throughout program's execution
  - Stack:
    - Allocated & deallocated in last-in, first-out order, with subroutine calls and returns)
  - Heap:
    - Allocated and deallocated at arbitrary times.

- Stack and heap is used for dynamic allocation

11

Static Allocation

- Static allocation for
  - Machine language translation of the source code
  - Global variables
  - Local but static variables
  - explicit constants (including strings, sets, etc)
  - scalars may be stored in the instructions

12