FORTRAN, Part 3

CS4100
February 18, 2011

Reminders

- Jeopardy tournament with Watson ends
- Project proposals due Today
  - Please upload to submission system
  - By midnight

Activation Records

- What happens when a subprogram is called?
  - Transmit parameters
  - Save caller's status
  - Enter the subprogram
  - Restore caller's state
  - Return to caller

What happens exactly?

- Before subprogram invocation:
  - Place parameters into callee's activation record
  - Save caller's status
    - Save content of registers
    - Save instruction pointer (IP)
  - Save pointer to caller's activation record in callee's activation record
  - Enter the subprogram

What happens exactly?

- Returning from subprogram:
  - Restore instruction pointer to caller's
  - Return to caller
  - Caller needs to restore its state (registers)
  - If subprogram is a function, return value must be made accessible

Contents of Activation Record

- Parameters passed to subprogram
- P (resumption address)
- Dynamic link (address of caller's activation record)
- Temporary areas for storing registers

DESIGN: Data Structures

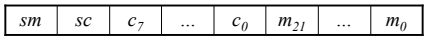
- First data structures
  - Suggested by mathematics
    - Primitives
    - Arrays

Primitives

- Primitives are scalars only
  - Integers
  - Floating point numbers
  - Double-precision floating point
  - Complex numbers
  - No text (string) processing

Representations

- Word-oriented
  - Most commonly 32 bits
- Integer
  - Represented on 31 bits + 1 sign bit
- Floating point
  - Using scientific notation: characteristic + mantissa



Arithmetic Operators

- 2 + 3.1 = ?
  - 2 is integer, 3.1 is floating point
- How do we handle this situation?
  - Explicit type-casting: FLOAT(2) + 3.1
    - Type-casting is also called "*coercion*"
  - FORTRAN: Operators are overloaded
  - Automatic type coercion
    - Always coerce to encompassing set
      - Integer + Float → float addition
      - Float * Double → double multiplication
      - Integer – Complex → complex subtraction
    - Types *dominate* their subsets

Example

- X**(1/3) = ?

  1/3 = 0

  1/3.0 = 0.33333

Hollerith Constants

- Early form of character string in FORTRAN
  - 6HCARMEL is a six character string 'CARMEL' (H is for Hollerith)
  - Second-class citizens
    - No operations allowed
    - Can be read into an integer variable, which cannot (should not) be altered
- Problems
  - Integer representing a Hollerith constant may be altered, which makes no sense
- Weak typing
  - No type checking is performed

Constructor: Array

- Constructor
  - Method to build complex data structures from primitive ones
- FORTRAN only has array constructors

```
DIMENSION DTA, COORD(10,10)
```

  - Initialization is not required
  - Maximum 3 dimensions

Representation

- Simple, intuitive representation
- Column-major order
  - Most languages do row-major order
  - Addressing equation:
    - $\alpha\{A(2)\} = \alpha\{A(1)\} + 1 = \alpha\{A(1)\} - 1 + 2$
    - $\alpha\{A(i)\} = \alpha\{A(1)\} - 1 + i$
    - $\alpha\{A(i,j)\} = \alpha\{A(1,1)\} + (j - 1)m + i - 1$
    - FORTRAN uses 1-based addressing
      - One addressable slot of each elt

Element	Address
A(1,1)	A
A(2,1)	A + 1
…	
A(m,1)	A + m - 1
A(1,2)	A + m
…	
A(m,2)	A + 2m - 1
…	
A(m,n)	A + nm - 1

Optimizations

- Arrays are mostly associated with loops
  - Most programmers initialize controlled variable to 1, and reference array A(i)
  - Optimization:
    - Initialize controlled variable to address of array element
    - Therefore, we'll increment address itself
    - Dereference controlled variable to get array element

Subscripts

- Subscripts can be expressions
  - A(i+m*c)
  - This defeats above optimization
  - Therefore, subscripts are limited to
    - c and c' are integers, v is an integer variable
    - c
    - v
    - v+c, v-c
    - c*v
    - c*v+c', c*v-c'
  - A(J - 1) ok; A(1+J) not ok
- Optimizations like this sold FORTRAN

DESIGN: Name Structures

- What do name structures structure?
  - Names, of course!
- Primitives bind names to objects
  - INTEGER I, J, K
    - Allocate integers I, J, and K, and bind the names to memory locations
    - Declare: name, type, storage

Declarations

- Declarations are non-executable statements
- Unlike IF, GOTO, etc., which are executable statements
- Static allocation
  - Allocated once, cannot be deallocated for reuse
  - FORTRAN does not do dynamic allocation

Optional Declaration

- FORTRAN does not require variables to be declared
  - First use will declare a variable
- What's wrong with this?
  - COUNT = COUMT + 1
  - What if first use is not assignment?
- Convention:
  - Variables starting with letters i, j, k, l, m, n are integers
  - Others are floating point
  - Bad practice: Encourages funny names (KOUNT, ISUM, XLENGTH…)

Now: Semantics (meaning)

- "They went to the bank of the Rio Grande."
- What does this mean?
- How do we know?
- CONTEXT, CONTEXT, CONTEXT

Programming Languages

- X = COUNT(I)
- What does this mean
  - X integer or real
  - COUNT array or function
- Again Context
  - Set of variables visible when statement is seen
- Context is called ENVIRONMENT