CS342: Organization of Prog. Languages

Topic 20: Evolution of Programming Languages

- How do languages evolve?
- What does this imply about how we use them?
- What does this imply about implementations?
Evolution of C

• B – like C with no types other than machine word.

• Early C (1969-1973): Used to implement early Unix. Like B, but with declarations. E.g. int, double, char *.

• K+R C (1978): Published book ”The C Programming Language”. Introduced struct, long and unsigned ints. Changed, e.g. a -= b to a -= b.

• Evolution... void, enum, const. struct and union assignments and function returns.

• Standard C (ANSI C89 == ISO C90): Superset of K+R. prototypes, void*, locales, expanded CPP.

• ANSI C99: inline functions, variables declared anywhere, long long, boolean, complex, variable length arrays, line end comments //, compound literals, n-ary macros, ...

• Feature exchange with C++
Evolution of Fortran

- Fortran (mid 1950s)
  Assignment, arrays (DIMENSION), aliasing (EQUIVALENCE)
  Arithmetic IF, GOTO, computed GOTO, assigned GOTO, DO loops
  PAUSE, STOP, CONTINUE
  Formatted and unformatted IO: tapes, punches, printers, drums.
  Fixed format source.

  SUBROUTINE FUNCTION END
  CALL RETURN
  COMMON


- Fortran IV (1962). Removed many machine dependecies.
  LOGICAL data type, logical IF. Popular. WATFIV.
- Fortran 66. ANSI Standard. Same level as Fortran IV.

- Fortran 77. ANSI Standard.
  CHARACTER type, IMPLICIT statement
  IF-THEN-ELSE, OPEN, CLOSE, PARAMETER, ...

- Fortran 90. ANSI Standard.
  Free-form source.
  Modules, recursive procedures, operator overloading, derived and abstract types, whole array operations, dynamic memory allocation, pointers, EXIT and CYCLE statements, in-line comments, portable numeric specification, new builtins.

- Fortran 95. ANSI Standard, minor cleanup of Fortran 90.
  Extensions for vectorization (FOR ALL, nested WHERE), PURE and ELEMENTAL procedures, automatic de-allocation of certain arrays.

• Fortran 2008. Minor upgrade of Fortran 2003. BIT (TYPELESS) data type, intelligent macros. SIMD parallel processing model, DO CONCURRENT, critical sections, synchronization. Oboletes ENTRY.
Evolution of Lisp

- Lisp 1.5 first widely distributed version
- Stanford Lisp
- MACLISP (MIT Project MAC, MACSYMA)
- InterLisp (BBN, Xerox Lisp Machines).
- FranzLisp (Berkeley, dialect of MACLISP)
- Zeta Lisp (Lisp machines, dialect of MACLISP)
- Scheme R5RS.
- Common Lisp
- Emacs Lisp, XLisp
• ObjectLisp, Loops, Flavors, CLOS.
• Scheme R6RS (in progress).
What does this mean?

- Popular languages do keep up to date.
- Issues in which version of the language a project shall assume.
- Language processors evolve over time (and get messier).
- Language processors have to deal with very different legacy code.
- Sometimes language status becomes fossilized in standards (CORBA IDL)
- The compiler middleware project.