Introduction, Computer Operations, Data, and Program Development

Meteorology 227
Fall 2015
Programming?

• Programming Language: An artificial language that can be used to control the behavior of a machine (often a computer). (Wikipedia)
  – A standard communication technique for expressing instructions to a computer (Wikipedia).

• Common (and not so common) languages:
  • Fortran, C, C++, C#, Python, Perl, COBOL, BASIC, Pascal, Java, PHP, Lisp, Ruby, AJAX, and so on, and so on.
Why Fortran?

• Built for scientific programming.

• First “High-Level” programming language.
  – Platform independent
  – Statements don’t look like machines language.
  – Portability, ease of use.

• Legacy codes
Programming for Meteorology and ISU

• NWS/Broadcast
  – AWIPS2 is primarily written in Java and plugins to AWIPS2 in Python.

• Research/Graduate School
  – Fortran, Python
  – Legacy codes, rapid processing of data.

• How does this impact ISU meteorology?
  – Programming requirement will accept either Mteor 227 or Comp Sci 207 (Java)
  – Mteor 227 will be offered every year during the fall semester.
  – Fortran and Python
History

• 1954-57
  – John Backus (IBM)
  – IBM Mathematical FORmula TRANslation system
  – Fortran 0 and Fortran I

• 1958
  – Fortran II
    • Separate compilation of modules.
  – Fortran III
    • Inlined assembly code.

• 1961
  – Fortran IV
  – Improved portability.
  – Implementation of new statements (common and equivalence).
• 1963
  – ~40 different compilers.
    • Compiler: translates the Fortran code to something that the machine will understand.
  – Standardization needed.
• 1966
  – Fortran 66
  – First ANSI version.
    • ANSI – American National Standard Institute
• 1978
  – Fortran 77
  – Second standard
  – Structured programming and other new features.
• 1991
  – Fortran 90
  – Third standard
  – New version promised in 10 years.
• 1997
  – Fortran 95
  – Largely a ‘Bug-Fix’ release of Fortran 90.
  – Some extensions, mainly HPF extensions (see below)
  – Fourth standard

• Late 2004
  – Fortran 2003
  – Object Oriented programming support.
  – Improved operability with C.

• Late 2010
  – Fortran 2008 (Find out more at http://j3-fortran.org/)
  – Co-Array FORTRAN (see below) extensions.
History cont.

• 2015
  – Fortran 2015
  – Planned minor revision
  – Further interoperability between FORTRAN and C.
  – More Parallel features

• Other types of Fortran
  – Co-Array Fortran (F--): Extension of 95/2003 for parallel processing.
Six Basic Computer Operations

1. Receive Information
   • Read TEMP
   • Get MAX_TEMP
   • Read TEMP, DEW_POINT

2. Put Out Information
   • Print ‘Tornado Warning’
   • Write METAR to file
   • Print TEMP, DEW_POINT

3. Perform Arithmetic
   • Add DAILY_RAIN to MONTHLY_RAIN
   • COUNT=COUNT+1
Six Basic Computer Operations

4. Assign a value to a variable or memory location.
   • Initialize MAX_TEMP, MIN_TEMP to zero.
   • Set counter to zero.
   • RAIN = RAIN + INCREMENT

5. Compare two variables and select one of two options
   • Board Example

6. Repeat a group of actions
   • Board Example
Data Types

• Integer
  – 32, -40, 212

• Real
  – 3.14, 2.5E6, 9.81

• Character
  – ‘F’, ‘C’, ‘%’

• Boolean
  – Two possible values: true or false
Stages in Program Development

• Programming: Development of a solution to an identified problem, and the setting up of a related series of instructions which, when directed through computer hardware, will produce the desired result.

• How do you do this?
  – Jumping straight to the code can be time consuming (error checking) and inefficient.
  – Seven Steps
Program Development

1. Define the problem
   – Inputs
   – Outputs
   – Processing steps to produce the required output
   – Defining diagram (later)

2. Outline the solution
   – Break into smaller tasks or steps
   – Establish an outline solution
     • Major processing steps, major subtasks (if any), user interface (if any), major control structures, major variables and record structures, mainline logic
Program Development cont.

3. Develop the outline into an algorithm
   – A set of precise steps that describe exactly the tasks to be performed and the order in which they are to be carried out.
   – Pseudocode, flow-charts, Nassi-Schneidermann diagrams.

4. Test the algorithm for correctness.
   – Use test data to check instructions
   – Keep track of all major variables
     • Desk check
Program Development cont.

5. Code the algorithm into a specific programming language.
   – Finally, you get to write code!

6. Run the program on the computer.

7. Document and maintain the program.
   – Document, document, document!
   – Comments, comments, comments!
Algorithm

• A set of detailed, unambiguous, and ordered instructions developed to describe the processes necessary to produce the desired output from a given input.
  – Lists the steps involved in accomplishing a task.

• Written in English and not a formal document.

• Pseudocode, flowcharts, Nassi-Schneiderman diagrams.
Pseudocode

• Essentially structured English
• Statements written in simple English
• Each instruction is written on a separate line.
• Keywords and indentation are used to signify particular control structures.
• Each set of instructions is written from top to bottom, with only one entry and one exit.
• Groups of statements may be formed into modules, and that group given a name.
Flowcharts

- Terminal symbol (starting and stopping points)
- Input/Output symbols
- Process symbols
- Predefined process symbol
- Decision symbol
- flow lines
Example Problem

• Take a temperature input from the user in either degrees F or C and output the same temperature converted to the other unit.
  – Follow 7 steps of program design.
  – Defining diagram.
  – Solution algorithm (flowchart)
  – Desk Check.