Different Forms of Software

Steven W. Oxman
NCCA
IT Division
Software

• Logic that is run on computing machinery
  – Data Logic
  – Computational Logic
  – Networking Logic
  – Presentation Logic

• Otherwise Known As:
  – Computer Programs
  – Firmware
  – Computing Logic
  – Code
Types of Software

- Device Automation (sensors and actuators)
- Platforms (ship control systems)
- Communications (software controlled terminals)
- Business Software (HR Systems)
- Enterprise Resource Planning (Navy ERP)
Software Language Levels

- Machine Language
- Assembler Language
- High Level Language
- 4th Gen Language
- 5th Gen Language (aka, Artificial Intelligence)
- Specialty Levels (e.g., RICE)
Software Languages

• Machine Language
• Assembler Language (ULTRA)
• Fortran, COBOL, PASCAL, CMS-2, C
• Focus
• ADA, Prolog, LISP, C++, VB
• VBA, VBS, C#, JAVA, JAVAScript, PERL
• ABAP (SAP), Cold Fusion
Software Structures

• Straight Code
• Data Driven Code
• Object Oriented Code
• RICE Object Code
Other Software Classifiers

• Real Time
• Batch
• Interactive
• On-Demand
Other Software Classifiers in DON

- Tactical Software
- Non-Tactical Software
- Human Safety Certified Software
- Nuclear Safety Certified Software
Machine Code

- 100000010 01 00101: Load Memory 5 > R1
- 100000010 10 00101: Load Memory 5 > R2
- 1010000100 00 01 10: R1 + R2 > R0
- 100000100 00 00110: Store R0 > Memory 6
# Assembler Code

<table>
<thead>
<tr>
<th>Programs</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>Identifies names of program</td>
</tr>
<tr>
<td>BALR 15,0</td>
<td>Start register 15 to address of the next instruction</td>
</tr>
<tr>
<td>USING</td>
<td>Pseudo-op indicating to assembler register 15 is base register and its content is address of next instruction</td>
</tr>
<tr>
<td>BEGIN+2,15</td>
<td></td>
</tr>
<tr>
<td>SR 4,4</td>
<td>Clear register 4 (set index=0)</td>
</tr>
<tr>
<td>L 3,TEN</td>
<td>Load the number 10 into register 3</td>
</tr>
<tr>
<td>L 2,DATA(4)</td>
<td>Load data (index) into register 2</td>
</tr>
<tr>
<td>A 2,FORTY9</td>
<td>Add 49</td>
</tr>
<tr>
<td>ST 2,DATA(4)</td>
<td>Store updated value of data (index)</td>
</tr>
<tr>
<td>A 4,FOUR</td>
<td>Add 4 to register 4 (set index – index+4)</td>
</tr>
<tr>
<td>BCT 3,LOOP</td>
<td>Decrement register 3 by 1, if result non zero, branch back to loop</td>
</tr>
<tr>
<td>TEN</td>
<td>branch back to caller</td>
</tr>
<tr>
<td>FOUR</td>
<td>Constant 10</td>
</tr>
<tr>
<td>FOURTY9</td>
<td>Constant 4</td>
</tr>
<tr>
<td>DATA</td>
<td>Constant 49</td>
</tr>
<tr>
<td></td>
<td>Words to be processed</td>
</tr>
</tbody>
</table>

END
Fortran Code

PROGRAM SQUARE
DO 15, I = 1,10
WRITE(*, *) I*I
15 CONTINUE
END
C# Code

using System;

class Squares1 {
    static void Main() {
        for (int i = 1; i <= 10; i++) {
            Console.WriteLine("{0} ", i * i);
            Console.WriteLine("{0} ", i * i);
        }
    }
}

<html>
<head>
  <title>Javascript Squares</title>
</head>
<body>

  <script>
    for (var i = 1; i <= 10; ++i) {
      document.write( Math.pow(i, i) + "<br>" );
    }
  </script>

</body>
</html>
LISP Code

(dotimes (i 10)
  (format t "~D " (expt i 2)))
PERL Code

print join(" ", map { $_[**2] } 1..10), "\n";
ABAP (SAP) Code

start-of-selection.
classname_select = p_class.
classstype_select = 'OT'.
objkey_select = p_objct.
PERFORM f_bds_call_navigator.

at line-selection.
PERFORM f_line_selection.

FORM f_bds_call_navigator.
* Function : BDS_CALL_NAVIGATOR
* -- data declaration -- *
DATA: i_connections LIKE bdn_con OCCURS 0 WITH HEADER LINE,
calling_signa LIKE bapisignat OCCURS 0 WITH HEADER LINE,
i_count LIKE sy-index,
i_gui_type LIKE bapibds01-type,
web_excluding LIKE bdn_fkt OCCURS 1 WITH HEADER LINE.
REPORT demo_mod_tech_example_2.
DATA: num TYPE i VALUE 5,
       fac TYPE i VALUE 0.
PERFORM fact USING num CHANGING fac.
WRITE: / 'Factorial of', num, 'is', fac.
FORM fact
   USING value(f_num) TYPE i
   CHANGING f_fact TYPE i.
   f_fact = 1.
   WHILE f_num GE 1.
       f_fact = f_fact * f_num.
       f_num = f_num - 1.
   ENDWHILE.
ENDFORM.
Software Issues for Costers

• Software Use (real time versus office use)
• Software Testing (path testing vs sim/stim)
• Software Language (generation?)
• Special Requirements (e.g., nuc cert)
• Code Counting (whose counting rules)
• Straight code versus object oriented code
• In-code documentation
• DBMS vs Data Tables
• ...

18