INTRODUCTION TO COMPUTING
TOPIC 8: SOFTWARE

PAUL L. BAILEY

1. What is software?

Software consists of the instructions which tell the computer's hardware how to behave. When manufactured, the computer has virtually no software. The software is installed on the computer.

The instructions, at their lowest level, consist of a sequence of words, which are in turn a sequence of bits. These bits are stored in the computer's memory or disk drives; they are changeable.

2. Types of software

We may classify software by its general function, and by its level.

2.1. Functional Categories.

- System software: coordinates and manages hardware operations
  - Drivers: manage particular peripheral devices
    - Video driver
    - Printer driver
    - Disk driver
  - File system: knows where files are kept on the disk, and how to access them
- Programming languages: enables programmers to create new software
  - Compilers - take the "source code" and convert it into machine language, which is then executed by the computer
  - Interpreters - execute the program one command at a time; the interpreter itself is the program being executed, what it does is interpret the program
- Applications: programs written for the "end-user"
  - Games, entertainment
  - Horizontal market applications: word-processing, spreadsheets
  - Vertical market applications: information systems for hospitals, banks, etc.

2.2. Generation Categories.

- First Generation - Machine language: a sequence of single word instructions
- Second Generation - Assembler: a sequence of mnemonics corresponding to machine instructions
- Third Generation - Compilers: higher level languages such as Pascal and C
- Fourth Generation - Code generators, whose output is the source code for a compiler, typically with a specific built-in database management tool

Date: April 24, 2008.

To see the variety of how programs look, we give three examples; each example computes the factorial of a positive integer. Recall that n factorial, written $n!$, is define the be the product of the distinct integers between 1 and n:

$$n! = 1 \times 2 \times \cdots \times (n-1) \times n.$$

For example,

$$5! = 1 \times 2 \times 3 \times 4 \times 5 = 120.$$

Assembly Language for the Intel series:

```
; Assembly language subroutine to compute the factorial of n

        mov  ecx,n                  ;copy parameter n to ECX register
        push ecx                    ;push parameter onto the stack
        call factor                 ;call the local factorial procedure
        jmp  outtahere              ;we have an answer, so jump to exit
  factor:                           ;We are forced to use the simplest form of
                                    ;an assembly procedure since inline code does
                                    ;not support the PROC assembler directive.
        push ebp                    ;Establish the "stack frame"
        mov  ebp,esp                ;
        push ecx                    ;place a copy of "n" on the stack
        ;if n = 1                   ;if "n" is not equal to 1
        cmp  dword ptr [ebp+8],1    ;  "n" is located on the stack
        jne  else01                 ;  jump to the else condition
        mov  eax,1                  ;  otherwise n==1, so set EAX to 1
        jmp  endif01                ;   and exit the if
  else01:                           ;
        mov  ecx,dword ptr [ebp+8] ;else retrieve n from stack
        dec  ecx                    ;  decrement n
        push ecx                    ;  place n back on the stack
        call factor                 ;  and call factor again
        add  esp, 4                 ;reduce the stack pointer by one dword
        mul  dword ptr [ebp+8]      ;multiply EAX by n
        push ecx                    ;place n parameter back on the stack
  endif01:                          ;end of if statement
        pop  ecx                    ;remove left over paramter from the stack
        pop  ebp                    ;restore the original base pointer
        ret  4                      ;discard the original value of n
  outtahere:                        ;the answer is in EAX register
        mov  n,eax                  ;so copy it to the parameter n
```

We see that the programs for higher level languages are much shorter. They are also slower.

C Compiler:

```
// factorial.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"

int factorial(int n)
{ int f=1,a=2;
  while (a<=n)
  { f=f*a;
    a=a+1; }
  return f; }

int main(int argc, char* argv[])
{ int n=5,f=factorial(n);
  printf("%d! = %d\n",n,f);
return 0; }
```

BASIC Interpreter:

```
10 LET N=5
20 GOSUB 100
30 PRINT N,F
40 END
100 F=1
110 A=2
120 IF A<=N THEN LET F=F*A: LET A=A+1: GOTO 120
130 RETURN
```

Department of Mathematics & Computer Science, Southern Arkansas University
E-mail address: plbailey@saumag.edu