# ECE199 Exam 1, Fall 2012

Tuesday, 18 September

Name and UIUC netid:

- **Be sure that your exam booklet has 9 pages.**
- **Write your name at the top of each page.**
- **The exam is meant to be taken apart.**
- **This is a closed book exam.**
- **You are allowed one $8.5 \times 11$" sheet of notes.**
- **We have provided a scratch sheet and an ASCII table at the back.**
- **Absolutely no interaction between students is allowed.**
- **Show all of your work.**
- **Don't panic, and good luck!**

Now...if you trust in yourself...and believe in your dreams...and follow your star...you'll still get beaten by people who spent their time working hard and learning things and weren't so lazy.
— from The Wee Free Men, by Terry Pratchett

| | | |
|---|---|---|
| Problem 1 | 20 points | _____ |
| Problem 2 | 15 points | _____ |
| Problem 3 | 20 points | _____ |
| Problem 4 | 25 points | _____ |
| Problem 5 | 20 points | _____ |
| Total | 100 points | _____ |

**Problem 1** (20 points): Representations

**Part A** (3 points): Express the 32-bit binary sequence "0110 1100 0111 0101 0110 0011 0110 1011" in hexadecimal.

**Part B** (4 points): Interpret the four successive 8-bit bytes making up the binary sequence in Part A as a four-character ASCII sequence. As your answer, give the equivalent ASCII sequence.

**Part C** (6 points): For the two eight-bit binary numbers, $A = 01101101$ and $B = 10110111$, give the result of the following bitwise logical operations.

$A$ AND $B =$ _____

$A$ OR $B =$ _____

$A$ XOR $B =$ _____

**Part B** (7 points): Express the decimal number 14.5 in IEEE 32-bit floating point representation in bits.

**Problem 2** (15 points): 2's-Complement Arithmetic

Please compute the following arithmetic operations in 8-bit 2's complement. Express your answer as an 8-bit 2's complement number. Indicate if it has an overflow by circling the corresponding YES or NO.

**Part A** (3 points): $00110110 + 00000100 =$ _____ Overflow? YES NO

**Part B** (3 points): $01101001 + 10111010 =$ _____ Overflow? YES NO

**Part C** (3 points): $10101101 + 10110110 =$ _____ Overflow? YES NO

**Part D** (3 points): $10011011 - 11001100 =$ _____ Overflow? YES NO

**Part E** (3 points): $01010101 + 00101011 =$ _____ Overflow? YES NO

**Problem 3** (20 points): Boolean Expressions and Truth Tables

**Part A** (10 points): Create the truth table for the following Boolean expression.

$$F(x, y, z) = (\bar{x} + yz) + \bar{y}$$

| $x$ | $y$ | $z$ | $F(x, y, z)$ |
|---|---|---|---|
| 0 | 0 | 0 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | |

**Part B** (10 points): Create a Boolean expression from the following truth table.

| $a$ | $b$ | $c$ | $G(a, b, c)$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**Problem 4** (25 points): C Program Analysis

Consider the following "mystery" C program, to which the inputs 5, 44, -2, 13, 50, 60, 55 will be given until the program terminates. (Note that the program may not scan all of those values.) For this problem, analyze and execute the program in your head (you can make notes on this page or on the scratch pages if needed) to find the results of the computation.

```
/* mystery.c */
#include <stdio.h>

#define A_VAL 5
#define MIN_VAL -9999

int main()
{
   int ii;
   int value;
   int value1 = MIN_VAL;
   int value2 = MIN_VAL;

   for ( ii = 0; ii < A_VAL; ii = ii + 1 )
   {
      scanf("%d", &value);
      if ( value > value1 )
      {
         value2 = value1;
         value1 = value;
      }
      else
      if ( value > value2 )
      {
          value2 = value;
      }

      /* CHECKPOINT FOR PART A */
   }

   printf("The output value is %d\nGoodbye!", value2);

   return 0;
}
```

**Part A** (15 points): At the location in the program marked "CHECKPOINT FOR PART A", determine and list the current values of the variables for each time that the program reaches that checkpoint. Fill in only as many rows as needed below.

ii = _____ value = _____ value1 = _____ value2 = _____

ii = _____ value = _____ value1 = _____ value2 = _____

ii = _____ value = _____ value1 = _____ value2 = _____

ii = _____ value = _____ value1 = _____ value2 = _____

ii = _____ value = _____ value1 = _____ value2 = _____

ii = _____ value = _____ value1 = _____ value2 = _____

ii = _____ value = _____ value1 = _____ value2 = _____

**Part B** (5 points): Write down EXACTLY the formatted text that will be printed on the terminal screen by the final `printf` statement in the program.

**Part C** (5 points): Complete the following sentence to describe the computational task performed by this "mystery" program.

The "mystery.c" program finds the _____

of a series of _____ [tell how many] integer input values."

**Problem 5** (20 points): Short Answers

Answer the following questions in **TWENTY-FIVE WORDS OR LESS**. We do not promise to read more, so be concise in your answers.

**Part A** (5 points): Think about the operations provided by the water faucet abstraction. Even faucets that have operations for both hot and cold water almost never provide an operation of the form, "Turn on water at $X$ degrees Fahrenheit." Twenty years ago, no faucets supported such an operation. Explain why such an operation is not common.

**Part B** (5 points): Recall the layers of abstraction in a computer system as discussed in the textbook and in class. Which layer specifies the operations that a specific computer, such as one based on an x86 processor or an ARM processor, is capable of executing? **(Clearly draw an arrow to or circle one of the layers in the figure to the right.)**

| Problems |
|---|
| Algorithms |
| Programming Language |
| Machine/Instruction Set Architecture |
| Microarchitecture |
| Circuits |
| Devices |

**Part C** (5 points): Write the complete sequence of numbers printed by the following code:

```
int i;

for (i = 0; 10 >= i; i = i + 3) {
    printf ("%d\n", i);
}
```

**Problem 5, continued:**

**Part D** (5 points): Consider the program below:

```
int main () {
    int i;            /* 32-bit 2's complement */
    unsigned int j;   /* 32-bit unsigned       */

    printf ("Type a number:  ");
    scanf ("%d", &i);


    j = i;            /* copies all 32 bits    */

    if ( ____________________________________ ) {
        printf ("Negative!\n");
    }
    return 0;
}
```

Fill in the blank by writing an expression based on variable `j` that checks whether the number entered is negative. **Your expression may not use variable `i`.**

Adapted from a LaTex ASCII table by (c) 2009 Michael Goerz

| Dec | Hex | Char | Dec | Hex | Char | Dec | Hex | Char | Dec | Hex | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 000 | 00*h* | (nul) | 001 | 01*h* | (soh) | 002 | 02*h* | (stx) | 003 | 03*h* | (etx) |
| 004 | 04*h* | (eot) | 005 | 05*h* | (enq) | 006 | 06*h* | (ack) | 007 | 07*h* | (bel) |
| 008 | 08*h* | (bs) | 009 | 09*h* | (tab) | 010 | 0A*h* | (lf) | 011 | 0B*h* | (vt) |
| 012 | 0C*h* | (np) | 013 | 0D*h* | (cr) | 014 | 0E*h* | (so) | 015 | 0F*h* | (si) |
| 016 | 10*h* | (dle) | 017 | 11*h* | (dc1) | 018 | 12*h* | (dc2) | 019 | 13*h* | (dc3) |
| 020 | 14*h* | (dc4) | 021 | 15*h* | (nak) | 022 | 16*h* | (syn) | 023 | 17*h* | (etb) |
| 024 | 18*h* | (can) | 025 | 19*h* | (em) | 026 | 1A*h* | (eof) | 027 | 1B*h* | (esc) |
| 028 | 1C*h* | (fs) | 029 | 1D*h* | (gs) | 030 | 1E*h* | (rs) | 031 | 1F*h* | (us) |
| 032 | 20*h* | (space) | 033 | 21*h* | ! | 034 | 22*h* | " | 035 | 23*h* | # |
| 036 | 24*h* | $ | 037 | 25*h* | % | 038 | 26*h* | & | 039 | 27*h* | ‘ |
| 040 | 28*h* | ( | 041 | 29*h* | ) | 042 | 2A*h* | * | 043 | 2B*h* | + |
| 044 | 2C*h* | ’ | 045 | 2D*h* | - | 046 | 2E*h* | . | 047 | 2F*h* | / |
| 048 | 30*h* | 0 | 049 | 31*h* | 1 | 050 | 32*h* | 2 | 051 | 33*h* | 3 |
| 052 | 34*h* | 4 | 053 | 35*h* | 5 | 054 | 36*h* | 6 | 055 | 37*h* | 7 |
| 056 | 38*h* | 8 | 057 | 39*h* | 9 | 058 | 3A*h* | : | 059 | 3B*h* | ; |
| 060 | 3C*h* | < | 061 | 3D*h* | = | 062 | 3E*h* | > | 063 | 3F*h* | ? |
| 064 | 40*h* | @ | 065 | 41*h* | A | 066 | 42*h* | B | 067 | 43*h* | C |
| 068 | 44*h* | D | 069 | 45*h* | E | 070 | 46*h* | F | 071 | 47*h* | G |
| 072 | 48*h* | H | 073 | 49*h* | I | 074 | 4A*h* | J | 075 | 4B*h* | K |
| 076 | 4C*h* | L | 077 | 4D*h* | M | 078 | 4E*h* | N | 079 | 4F*h* | O |
| 080 | 50*h* | P | 081 | 51*h* | Q | 082 | 52*h* | R | 083 | 53*h* | S |
| 084 | 54*h* | T | 085 | 55*h* | U | 086 | 56*h* | V | 087 | 57*h* | W |
| 088 | 58*h* | X | 089 | 59*h* | Y | 090 | 5A*h* | Z | 091 | 5B*h* | [ |
| 092 | 5C*h* | \ | 093 | 5D*h* | ] | 094 | 5E*h* | ^ | 095 | 5F*h* | _ |
| 096 | 60*h* | ‘ | 097 | 61*h* | a | 098 | 62*h* | b | 099 | 63*h* | c |
| 100 | 64*h* | d | 101 | 65*h* | e | 102 | 66*h* | f | 103 | 67*h* | g |
| 104 | 68*h* | h | 105 | 69*h* | i | 106 | 6A*h* | j | 107 | 6B*h* | k |
| 108 | 6C*h* | l | 109 | 6D*h* | m | 110 | 6E*h* | n | 111 | 6F*h* | o |
| 112 | 70*h* | p | 113 | 71*h* | q | 114 | 72*h* | r | 115 | 73*h* | s |
| 116 | 74*h* | t | 117 | 75*h* | u | 118 | 76*h* | v | 119 | 77*h* | w |
| 120 | 78*h* | x | 121 | 79*h* | y | 122 | 7A*h* | z | 123 | 7B*h* | { |
| 124 | 7C*h* | \| | 125 | 7D*h* | } | 126 | 7E*h* | ~ | 127 | 7F*h* | DEL |