16.216: ECE Application Programming

Spring 2012

Exam 1
February 24, 2012

Name: _____ ID #: _____ Section: _____

For this exam, you may use only one 8.5" x 11" double-sided page of notes. All electronic devices (e.g., calculators, cellular phones, PDAs) are prohibited. If you have a cellular phone, please turn it off prior to the start of the exam to avoid distracting other students.

The exam contains 3 questions for a total of 100 points. Please answer the questions in the spaces provided. If you need additional space, use the back of the page on which the question is written and clearly indicate that you have done so.

Please read each question carefully before you answer. In particular, note that:

- Question 3 has three parts, but you are only required to complete two of the three parts.
  - You may complete all three parts for up to 10 points of extra credit. If you do so, **please clearly indicate which part is the extra one—I will assume it is part (c) if you mark none of them.**
- For each part of that problem, you must complete a short program by writing all input and output statements (except prompts). I have provided comments to describe what your input/output statements should do, as well as written the rest of the program for you.
- You can solve each problem using only the variables that have been declared, but you may declare and use other variables if you want.

You will have 50 minutes to complete this exam.

Q1: Multiple choice	/ 20
Q2: Expressions and operators	/ 40
Q3: C input/output	/ 40
TOTAL SCORE	/ 100
EXTRA CREDIT	/ 10

1. (20 points, 5 points per part) ***Multiple choice***
For each of the multiple choice questions below, clearly indicate your response by circling or underlining the choice you think best answers the question.

a. Say you have the following code snippet:

```
int x = 5;
int y = 10;
if ((x % 2) == 1)
   y = y + x;
x = x + 1;
```

What are the values of `x` and `y` at the end of the sequence?

i. `x == 5, y == 10`

ii. `x == 6, y == 10`

iii. `x == 5, y == 15`

iv. `x == 6, y == 15`

b. Say you have the following conditional statement:

```
if (a < 0) {
   x = x + 1;
}
else if (b >= 0) {
   x = x – 1;
}
else {
   x = 0;
}
```

Assume `x == 5` initially. Which values of `a` and `b` shown below would cause `x` to be set to 0 when the code above is executed?

i. `a == -2, b == -2`

ii. `a == 1, b == -1`

iii. `a == 2, b == 5`

iv. `a == 0, b == 0`

1 (cont.)

c. Given an integer `val`, which of the following `if` statements prints `val` only if `val` is not in the range 0 through 100 (including the endpoints)?

i.
```
if ((val < 0) || (val > 100))
   printf("Value = %d\n", val);
```

ii.
```
if ((val < 0) && (val > 100))
   printf("Value = %d\n", val);
```

iii.
```
if ((val > 0) || (val < 100))
   printf("Value = %d\n", val);
```

iv.
```
if ((val > 0) && (val < 100))
   printf("Value = %d\n", val);
```

d. Which of the following `if` statements are completely valid? Circle all that apply.

i.
```
if (time_left_in_exam <= 5)
     panic = 1;
```

ii.
```
if (time_left_in_class_during_normal_lecture <= 25)
     printf("Zzzzzzzzzzz ... \n");
```

iii.
```
if (I_submitted_a_regrade_request)
     I_am_wondering_where_my_new_grade_file_is = 1;
```

iv.
```
if (1) {
     printf("I just realized there's no wrong answer ");
     printf("to this question.\n");
}
```

2. (40 points) ***Expressions and operators***

For each short program shown below, list the output exactly as it will appear on the screen. Be sure to clearly indicate spaces between characters when necessary.

You may use the available space to show your work as well as the output; just be sure to clearly mark where you show the output so that I can easily recognize your final answer.

a. (15 points)

```
#define ConstVal 2.5

void main() {
     int ivarA, ivarB;
     double dvarA, dvarB;

     ivarA = ConstVal + 3;
     ivarB = 5 / (ivarA – 2) + 3;

     dvarA = 3 * ConstVal;
     dvarB = -ivarA / 2 + 2;

     printf("Integers: %d\n%d\n", ivarA, ivarB);
     printf("\nDoubles: %lf %lf\n", dvarA, dvarB);
}
```

2 (cont.)

b. (13 points)

```
void main() {
     float f1, f2, f3;
     int x;

     f1 = 2 + 24.0 / 100;
     f2 = (f1 + 1.76) / 4 * 0.125;
     f3 = f1 / 2 + 0.8 / 2;
     x = f1;

     printf("f1 = %f\n", f1);
     printf("f2 = %.2f\n", f2);
     printf("f3 = %.*f\n", x, f3);
}
```

c. (12 points)

```
void main() {
     unsigned int H1, H2, H3;

     H1 = 0x101 | 0x404;
     H2 = ~H1 >> 8;
     H3 = H1 ^ 0xF0F;

     printf("H1: %x\n", H1);
     printf("H2: %x\n", H2);
     printf("H3: %x\n", H3);
}
```

3. (40 points, 20 per part) ***C input/output***

For each part of this problem, you are given a short program to complete. **CHOOSE ANY TWO OF THE THREE PARTS** and fill in the space provided with appropriate code. **You may complete all three parts for up to 10 points of extra credit, but must clearly indicate which part is the extra one—I will assume it is part (c) if you mark none of them.**

a. Complete the program below so that it prompts for and reads three unsigned decimal integers, then reprints them on three lines in hexadecimal using the following formatting:

- First value: show the value with a leading 0x but no other formatting.
- Second value: show exactly the lowest 16 bits of the value with a leading 0x. All 16 bits must be shown, including extra zeroes if necessary.
- Third value: show all 32 bits of the value with a leading 0x and extra zeroes if necessary.

Examples (with user input underlined; in the second case, note that 65792_{10} = 0x10100, but only the lowest 16 bits are shown):

```
Enter three values: 10 32 500
0xA
0x0020
0x000001F4
```

```
Enter three values: 500 65792 3
0x1F4
0x0100
0x00000003
```

```
void main() {
     unsigned int in1, in2, in3;   // Hexadecimal inputs

     // Prompt for and read three unsigned decimal values
     printf("Enter three values: ");




     // Print the hexadecimal form of the first value




     // Print the lowest 16 bits of the second value




     // Print all 32 bits of the third value



}
```

3 (cont.)

b. Complete the program below so that it prompts the user to enter four input values—the initial balance in a bank account, followed by the month, day, and amount of the most recent transaction—then prints two lines of output:

- The initial account balance, which is between 0.00 and 10000.00
- The transaction date and amount, as well as the account balance after the transaction
  - The date should be printed in MM/DD format (examples: 02/24, 12/03, 10/15)
  - The transaction amount is between -9999.99 and +9999.99, and its sign should always be shown.

Both the balance and transaction amount should be shown with two places after the decimal point. Each should always line up in the same way when printed, regardless of the number of digits. Examples are shown below, with user input underlined:

```
Input balance, month, day, and amount: 500 2 24 250
Balance:   500.00
02/24  +250.00   750.00

Input balance, month, day, and amount: 10.50 12 5 -50.25
Balance:    10.50
12/05   -50.25   -39.75
```

```
void main() {
     double bal, amt;    // Balance and transaction amount
     int mth, day;       // Month and day of transaction

     // Prompt for and read balance, month, day, and amount
     printf("Input balance, month, day, and amount: ");




     // Print initial balance




     // Print date (MM/DD format), amount, and balance




}
```

3 (cont.)

c. Complete the program below so that it prompts the user to enter a single integer followed by two double-precision variables, reads those values, and does the following:

- Reprint each double-precision value, along with the percentage of their total each value comprises (See the test cases below).
  - The integer, p, determines how many places should be printed for each value
  - To print a percent sign, use %% in your format string
    - For example, if `x == 5`, `printf("%d%%", x)` will output `5%`

```
Int + 2 doubles: 1 2 3
Input 1: 2.0 40.0%
Input 2: 3.0 60.0%
```

```
Int + 2 doubles: 3 0.1 0.3
Input 1: 0.100 25.000%
Input 2: 0.300 75.000%
```

```
void main() {
     int p;                                  // Output precision
     double input1, input2;                  // Input values


     // Prompt for and read integer + 2 doubles
     printf("Int + 2 doubles: ");




     /* Print each of the two doubles, followed by the
          percentage of their total each value comprises */









}
```