16.216: ECE Application Programming

Spring 2013

Exam 1
February 20, 2013

Name: _____ **ID #:** _____

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. I have provided comments to describe what your program should do, as well as written some of the program for you.
  - Note that each function contains both lines that are partially written (for example, a `printf()` call in which you are responsible for filling in the format string and expressions) and blank spaces in which you must write additional code.
- You can solve each part of Question 3 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/operators	/ 40
Q3: C input/output; conditional statements	/ 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. What is the output of the short code sequence below?

```
int x = 1;
while (x < 16) {
   x = x + 5;
   printf("%d ", x);
}
```

i. `6`

ii. `6 11`

iii. `1 6 11`

iv. `6 11 16`

v. `1 6 11 16`

b. What is the output of the short code sequence below?

```
int z = 1;
do {
   printf("z = %d\n", z);
   z = z + 1;
} while ((z % 3) != 0);
```

i. Nothing

ii. `z = 1`

iii.
```
z = 1
z = 2
```

iv.
```
z = 1
z = 2
z = 3
```

1 (cont.)

c. Given the code sequence below:

```
int inval;
int n = 0;
do {
   scanf("%d", &inval);
   n = n + 1;
} while ((inval > 1) && (inval <= 7));
```

Which of the following possible input values will cause the do-while loop to end? In other words, which value(s) will cause the loop condition to be false?

A. 0
B. 1
C. 5
D. 7
E. 9

i. A and B

ii. A and E

iii. A, B, and E

iv. A, B, D, and E

v. B, C, and D

d. Which of the following statements accurately reflect your opinion(s)? Circle all that apply (but please don't waste too much time on this "question")!

i. "This course is moving too quickly."

ii. "This course is moving too slowly."

iii. "I've attended very few lectures, so I don't really know what the pace of the course is."

iv. "I appreciate the opportunity to take a 50 minute nap three times a week."

v. "I hope the rest of the exam is this easy."

2. (40 points) ***Expressions/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. (14 points)

```
void main() {
     int val1, val2;
     double doub1, doub2;

     doub1 = (5 + 8 / 3) / 2;
     val1 = doub1 - 2;
     val2 = 4 * val1 + 1;
     doub2 = 10.0 / (val2 - val1);

     printf("%d\n%d\n", val1, val2);
     printf("\n%lf %lf\n", doub1, doub2);
}
```

2 (cont.)

b. (14 points)

```
void main() {
     double x = 16;
     double y = 4;
     double a, b, c;

     a = y / x;
     b = a * y + 0.138;
     c = b + 20;

     printf("%06.2lf %-6.2lf\n", a, b);
     printf("%+10.4lf\n", c);
}
```

c. (12 points)

For this program, assume the user input is as follows, with one space between each number:

```
  12 34 56 78
```

```
void main() {
     char c1, c2, c3;
     int i1, i2, i3;

     scanf("%d%c", &i1, &c1);
     scanf("%d %c", &i2, &c2);
     scanf(" %d %c", &i3, &c3);

     printf("%d %d %d\n", i1, i2, i3);
     printf("%c %c %c\n", c1, c2, c3);
}
```

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

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 spaces 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. This program calculates a student's overall GPA after 4 semesters, given the GPA and credits per semester. The program prompts for and reads the four GPAs and credit counts, then calculates and prints the appropriate values, as in the example below (input is underlined):

```
Enter GPAs: 3.5 3.2 2.7 3.8
Enter credits: 14 17 18 15
Total credits: 64
Overall GPA: 3.27          ← NOTE: GPA printed using 2 decimal places
```

The overall GPA is based on a weighted average. For example, after 2 semesters, a student who earned a 3.5 GPA while taking 12 credits and a 3.0 GPA while taking 15 credits would have a GPA of (3.5 * 12 + 3.0 * 15) / (12 + 15) = 3.22.

```
void main() {
   double G1, G2, G3, G4;      // Grade point averages
   int C1, C2, C3, C4;         // Credits per semester
   int total;                  // Overall total credits

   // Prompt for and read GPAs and credit counts
   printf("Enter GPAs: ");

   scanf(_____________________________, _____________________________);
   printf("Enter credits: ");

   scanf(_____________________________, _____________________________);

   // Calculate and print total credits and overall GPA
   // BE SURE TO FILL IN ALL BLANKS IN PRINTF() CALLS




   printf("Total credits: ______\n", _____________________________);



   printf("Overall GPA: _____________________\n",

          ____________________________________________________________);

}
```

3 (cont.)

b. This program should first prompt for and read two pairs of numbers; each pair represents the lower and upper bounds of a range. The program should then do the following:

- If both L1 and H1 are inside the range specified by L2 and H2, print "`R1 in R2`".
- If both L2 and H2 are inside the range specified by L1 and H1, print "`R2 in R1`".
- If the previous conditions are false but the ranges do overlap, print "`Overlap`".
- If all previous conditions are false, print "`No overlap`".

Three sample program runs are shown below, with user input underlined:

```
Range 1: 4 8          Range 1: 1.2 5.6       Range 1: 0 5
Range 2: 5 7          Range 2: 3.4 7.8       Range 2: 6 10
R2 in R1              Overlap                No overlap
```

```
void main() {
   double L1, H1;         // Endpoints of first range
   double L2, H2;         // Endpoints of second range

   // Prompt for and read ranges
   printf("Range 1: ");

   scanf(_____________________________, _____________________________);

   printf("Range 2: ");

   scanf(_____________________________, _____________________________);

   // Test for range overlaps and print appropriate statements


      printf("R1 in R2\n");



      printf("R2 in R1\n");



      printf("Overlap\n");



      printf("No overlap\n");
}
```

3 (cont.)

c. This program should first prompt for and read two numbers representing a month and a year—for example, this month (February 2013) would be `2 2013`. The program then prints the number of days in the month. If the user enters an invalid month, print an error. Note that:

- February (month 2) has 28 days in most years, but 29 days in a leap year.
  - Leap years occur every 4 years, except when the year is divisible by 100 and not also by 400. For example, 2000 and 2012 were leap years; 1800 and 1900 were not.
- April (month 4), June (6), September (9), and November (11) have 30 days.
- All other months have 31 days.

```
void main() {
   int month, year;  // Month and year

   // Prompt for and read month and year
   printf("Enter month & year: ");

   scanf(_____________________, _____________________);

   // Check month and print appropriate number of days
   // Remember to check for leap year if month is February




         printf("29 days\n");     // February in leap year


         printf("28 days\n");     // February--no leap year



      printf("30 days\n");          // April, June, Sept., Nov.



      printf("31 days\n");          // Jan., March, May, July
                                    // August, October, December


      printf("Error: invalid month %d\n", month);  // Error case

}
```