16.216: ECE Application Programming

Fall 2015

Exam 1
September 30, 2015

Name: _____

Section (circle 1): **201** *(12-12:50, Geiger)* **202** *(1-1:50, Geiger)* **203** *(1-1:50, Nasiri)*

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) 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 Question 3, you must complete a short program. I have provided comments to describe what your program should do and written some of the code.
  - Note that each program contains both lines that are partially written (for example, a `printf()` call missing the format string and expressions) and blank spaces in which you must write additional code. **You must write all code required to make each program work as described—do not simply fill in the blank lines.**
  - Each program is accompanied by one or more test cases. Each test case is an example of how the program should behave in one specific case—**it does not cover all possible results of running that program.**
- 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: C input/output; operators	/ 40
Q3: 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 one choice you think best answers the question.

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

```
int i = 10;
while (i > 0) {
   i = i / 2;
   printf("%d ", i);
}
```

i. `10`

ii. `5`

iii. `10 5 2 1`

iv. `5 2 1 0`

v. `10 5 2 1 0`

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

```
int z = -3;
do {
   printf("%d ", z);
   z = z * -2;
} while (z > 0);
printf("%d ", z);
```

i. No output—loop body never executes

ii. `-3`

iii. `-3 6`

iv. `-3 6 6`

v. `-3 6 -12`

1 (continued)

c. Given the code sequence below:

```
int x;
scanf("%d", &x);
switch (x / 4) {
case 0:
   printf("Zero ");
case 2:
   printf("Two ");
   break;
default:
   printf("Default");
}
```

Which of the following possible input values will produce the output "`Zero Two `"?

A. 0
B. 2
C. 4
D. 8
E. 9

i. Only A

ii. A and B

iii. A and C

iv. B and E

v. A, 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 hope the rest of the exam is as easy as this question."

2. (40 points) ***C input/output; 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. (12 points)

```
int main() {
    int v1, v2;
    double d1;
    double d2 = 18;

    v1 = d2 / 5.0;
    v2 = v1 + 15 / v1;
    d1 = d2 * 0.5 - v1;
    d2 = v2 / v1;

    printf("%d\n%d ", v1, v2);
    printf("\n%lf %lf\n", d1, d2);

    return 0;
}
```

2 (continued)
b. (14 points)

```
int main() {
    double d1, d2, d3;
    int x;

    d1 = 11.0 / 4;
    x = d1 + 10.25;
    d2 = -(d1 * 2) + x;
    d3 = d2 / 100;

    printf("%.0lf\n", d1);
    printf("%.2lf\n", d2);
    printf("%.1lf\n", d3);    // Print d3 with a precision of 1
    printf("%d\n", x);

    return 0;
}
```

2 (continued)
c. (14 points)

For this program, assume the user inputs the line below. The digit `'9'` is the first character the user types. There are two spaces (`'  '`) between the `'5'` in `9.30.2015` and the `'9'` in `900`, and two spaces between the second `'0'` in `900` and the `'1'` in `1000`.

You must determine how `scanf()` handles this input and then print the appropriate results. Note that the program may not read all characters on the input line.

```
  9.30.2015  900  1000

int main() {
   int ival1, ival2;
   double dval1, dval2;
   char ch1, ch2, ch3;

   scanf("%d%c%lf %c %c %d %lf",
          &ival1, &ch1, &dval1, &ch2,
          &ch3, &ival2, &dval2);

   printf("%d %d\n", ival1, ival2);
   printf("%.4lf %.4lf\n", dval1, dval2);
   printf("%c%c%c\n", ch1, ch2, ch3);

   return 0;
}
```

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.**

Remember, you must write all code required to make each program work as described—**you cannot simply fill in the blank lines and get full credit.** Also, remember that each example only applies to one specific case—**it does not cover all possible results for that program.**

a. This program reads three values representing the sides of a triangle. If any of the values are negative, the program prints an error and exits. Otherwise, it prints the type of triangle:

- If all three sides are equal, the triangle is equilateral.
- If two of the three sides are equal, the triangle is isosceles.
- If none of the sides are equal, the triangle is scalene.

Three test runs of the program are shown below, with user input underlined.

```
Sides: 1.2 3.4 1.2      Sides: 9.8 15 7         Sides: 5 -2 3
Isosceles               Scalene                 Error: negative side
```

```
void main() {
   double s1, s2, s3;      // Sides of a triangle

   // Prompt for and read triangle sides
   printf("Sides: ");

   scanf("_____________________________", _____________________________);

   // Input error: negative side length

   if (_____________________________________________________________) {
      printf("Error: Negative side\n");
      return;
   }

   // Determine triangle type--write code around printf() calls



      printf("Equilateral\n");



      printf("Isosceles\n");



      printf("Scalene\n");
}
```

3 (continued)

b. In the card game blackjack, players use a standard strategy to determine when to "hit" (take another card), "stand" (take no more cards), or "double down" (double your bet and hit). A simplified (and slightly incorrect) version of that strategy, which is based on the player's total and the one dealer card showing, is as follows:

   - The player doubles down if his total is 10 or 11 and the dealer's card is less than 10.
   - The player stands if his total is 17 or higher, or if his total is between 12 and 16 and the dealer's card is 6 or less.
   - The player hits in all other cases.

Complete the program below, which reads two integers (the player's total and the value of the dealer's card) and prints the correct choice based on the conditions described above.

Three sample program runs are shown below (user input underlined):

```
Player/dealer: 17 10      Player/dealer: 13 8      Player/dealer: 10 9
Stand                     Hit                      Double down
```

```
void main() {
   int plr;     // Player total
   int dlr;     // Value of dealer card

   // Prompt for and read player total and dealer card
   printf("Player/dealer: ");

   scanf("_____________________________", _____________________________);

   // Test conditions to determine when to double, hit, or stand
   // Write code around printf() calls



      printf("Double down\n");




      printf("Stand\n");



      printf("Hit\n");

}
```

3 (continued)

c. This program prompts the user to enter a single character command followed by three numbers: two integers representing the width and length of a rectangle, and a third integer representing the precision of the printed result. The command determines how the result is calculated:

- If the command is 'P' or 'p,' calculate the rectangle's perimeter (distance around the outside of the rectangle).
- If the command is 'A' or 'a', calculate the rectangle's area (product of its sides).
- For all other characters, print an error message showing the invalid command and exit.

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

```
Enter char, 3 ints: A 2 4 5          Enter char, 3 ints: P 4 8 0
Area = 8.00000                       Perimeter = 32
```

```
void main() {
   int width, length;     // Width and length of rectangle
   int prec;              // Precision of output
   char cmd;              // Single character command

   // Prompt for/read input values
   printf("Enter char, 3 ints: ");

   scanf("_____________________________", _____________________________);

   // Test cmd to determine what value to print
   // Complete printf() calls and write code needed around them




      printf("Area = __________\n", _____________________________);




      printf("Perimeter = __________\n", _____________________________);




      printf("Invalid command _______\n", _____________________________);

}
```