

EECE.2160: ECE Application Programming

Spring 2017

Exam 1

February 17, 2017

Name: _____

Section (circle 1): **201** (*Dr. Li, MWF 8-8:50*) **202** (*Dr. Geiger, MWF 12-12:50*)

For this exam, you may use only one 8.5" x 11" double-sided page of notes. All electronic devices (e.g., calculators, cell phones) are prohibited. If you have a cell phone, please turn off your ringer 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—we will assume it is part (c) if you mark none of them.**
- For each part of Question 3, you must complete a short program. We 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 = 1;
int sum = 10;
while (sum < 15) {
    sum = sum + i;
    printf ("%d %d ", i, sum);
    i = i * 2;
}
```

- i. 1 11 2 13
- ii. 2 11 4 13
- iii. 1 11 2 13 4 17
- iv. 2 11 4 13 8 17
- v. 1 10 2 11

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

```
int x = 10;
do {
    x = x / (-3);
    printf ("%d ", x);
    x = -x;
} while (x > 1);
printf ("%d ", x);
```

- i. -3 1
- ii. -3 -1
- iii. -3 -1 1
- iv. -3 1 -1
- v. None of the above

1 (continued)

c. Given the code sequence below:

```
int var;  
scanf ("%d", &var);  
switch (var % 3) {  
case 2:  
    var = var % 3;  
    break;  
case 1:  
    var = var % 3;  
case 0:  
    var = var + 1;  
}  
printf ("%d", var);
```

Which of the following possible input values will produce the output “2” (double quotes are not part of the output)?

- A. 0
- B. 2
- C. 4
- D. 6
- E. 8

- i. A and C
- ii. B and C
- iii. B and E
- iv. B, C and E
- v. None of the above

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 a;
    double d1, d2;
    d1 = 10.0;
    d2 = 5.0;
    d1 = d1 / d2;
    a = d1 + d2 * 0.5;
    d2 = d2 + a / d1;

    printf ("%d", a);
    printf ("\number: %lf\n%lf", d1, d2);
    return 0;
}
```

2 (continued)

b. (14 points)

```
int main () {
    int i;
    double d1, d2, d3;
    d1 = 15 / 3.0;
    d2 = d1 + 2.25;
    i = d2 - d1 + 1.23;
    d3 = 1.26 + 4 * (d2 - i);

    printf ("%d\n", i);
    printf ("% .21f\n", d1);
    printf ("% .11f\n", d2); // Print d2 with precision of 1
    printf ("% .01f\n", d3);
    return 0;
}
```

2 (continued)
c. (14 points)

For this program, assume the user inputs the line below. The digit '1' is the first character the user types. There are two spaces (' ') between the 1.38 and the -5.127.

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.

```
1.38 -5.127 1.63 .48
```

```
int main () {
    char ch1, ch2, ch3;
    int i1;
    double d1, d2, d3;

    scanf("%lf %c %lf %d %lf %c%c",
          &d1, &ch1, &d2, &i1, &d3, &ch2, &ch3);
    printf("%d\n", i1);
    printf("%lf %.2lf %.0lf\n", d1, d2, d3);
    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 should read three values representing possible sides of a triangle, then check (1) if the values represent a valid triangle, and (2) if valid, if they represent a right triangle. The values may be entered in any order—they're not necessarily sorted. Note that:

- For three numbers to be the sides of a valid triangle, the sum of any two sides must be greater than the third. You will have to check all possible combinations of two sides.
- For three numbers to be the sides of a right triangle, they must satisfy the Pythagorean theorem: $a^2 + b^2 = c^2$. You will have to check all possible combinations of two sides.

```
void main() {
    double s1, s2, s3;           // Side lengths
    double sq1, sq2, sq3;       // Squares of side lengths

    // Prompt for and read side lengths
    printf("Enter sides: ");

    scanf("_____ ", _____);

    // Test for valid triangle

    printf("Valid triangle\n");

    // Test for right triangle

    printf("Right triangle\n");

    // Account for cases that aren't valid triangles

    printf("Not a valid triangle\n");
}
```

3 (continued)

b. We obtained a copy of the program used to determine the university's response to a range of projected snow totals. You must complete the partial version below. The program prompts for and reads the minimum and maximum projected snowfall, then does the following:

- Checks to ensure min and max are in the correct order, printing an error message if not
- If the snowfall will be at least 0.1 inches, print "Parking ban"
- If the snowfall will be less than 3 inches, print "Still open"
- If the entire range of possible totals falls between 3 and 6 inches, print "Delay"
- If the snowfall could be more than 6 inches, print "Fine, we'll close"

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

Min/max snow: <u>0 0.05</u>	Min/max snow: <u>0.5 2.9</u>	Min/max snow: <u>5 7</u>
Still open	Parking ban	Parking ban
	Still open	Fine, we'll close

```
void main() {
    double min, max; // Minimum/maximum projected snowfall
    printf("Min/max snow: "); // Prompt for and read range
    scanf("_____", _____);
    // Error test

    if (_____)
        printf("You didn't follow directions!\n");

    // Test min/max & print messages (may print > 1 message)
    else {

        printf("Parking ban\n");

        printf("Still open\n");

        printf("Delay\n");

        printf("Fine, we'll close\n");

    }
}
```

3 (continued)

c. This program prompts the user to enter a 2- or 3-letter abbreviation indicating their engineering major: EE (Electrical), CpE (Computer), ChE (Chemical), CEE (Civil/Environmental), or PE (Plastics). The program should read all user input characters.

After reading the abbreviation, the program should test as many characters as possible to determine the correct major and print that major. Assume the user makes no errors. Three sample program runs are shown below, with the user input underlined:

Enter major: CpE
Computer

Enter major: EE
Electrical

Enter major: ChE
Chemical

```
void main() {
    char c1, c2, c3;        // Characters to hold major
    // Prompt for and read major
    printf("Enter major: ");
    scanf("_____", _____);
    // Test 1st character (and others, if needed) to pick major

    printf("Electrical\n");

    printf("Computer\n");

    printf("Chemical\n");

    printf("Civil/Environmental\n");

    printf("Mechanical\n");

    printf("Plastics\n");

}
}
```