

# EECE.2160: ECE Application Programming

Spring 2016

Exam 1

February 19, 2016

Name: \_\_\_\_\_

Section (circle 1):    201 (8-8:50, P. Li)                    202 (12-12:50, M. Geiger)

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—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 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
<b>TOTAL SCORE</b>	<b>/ 100</b>
<b>EXTRA CREDIT</b>	<b>/ 10</b>

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. Determine the output of the short code sequence below, given the following user input: C+

```
scanf ("%c", &grade);

switch (grade) {

case 'A': case 'a':
    printf("A");
    break;

case 'B': case 'b':
    printf("B");

case 'C': case 'c':
    printf("C");

case '+':
    printf("+");
    break;

default:
    printf("Not Applicable");
}
```

- i. A
- ii. B
- iii. C
- iv. C+
- v. Not Applicable

1 (continued)

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

```
int i = 0;
int j = 6;

while ((i * j) < 15) {
    i = i + 2;
    j = j - 1;
    printf("%d ", i);
}
```

i. 0

ii. 0 2

iii. 0 2 4

iv. 2 4

v. 2

1 (continued)

c. Which of the following code sequences will print "Bingo" if integer  $x$  is equal to -1?

A. `if ( (x >= -7) && (x < 1) || (x > 12) )  
 printf ("Bingo");`

B. `if ( !((x >= 2) || (x <= -5)) )  
 printf ("Bingo");`

C. `if ( (x >= 0) && (x <= 10) );  
 printf ("Bingo");`

D. `if (x = 0)  
 printf ("Miss");  
else if (x)  
 printf ("Again");  
else  
 printf ("Bingo");`

- i. Only B
  - ii. A and B
  - iii. A, B, and C
  - iv. A, B, and D
  - v. A, 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 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 we can easily recognize your final answer.

a. (12 points)

```
int main() {
    int v1;
    double d1, d2;
    int v2 = 3;
    double d3 = 14.75;

    d1 = v2 / 6;
    v1 = d1 * 7 + 12;

    d2 = (v2 + 6) % (v2 - 2);
    v2 = d3 + 2;

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

    return 0;
}
```

2 (continued)  
b. (14 points)

```
int main() {
    int i1 = 15;
    float f1, f2;
    double d1, d2;

    f1 = 3 + i1 % 4;
    f2 = f1 / (- 3.0);
    d1 = f1 + f2 + 0.987654;
    i1 = -i1;
    d2 = -i1 - 10.72;

    printf ("%d\n", i1);
    printf (".2lf, %.0lf\n", f1, f2);
    printf (".0lf\n", d1);
    printf (".0lf\n", d2);

    return 0;
}
```

2 (continued)  
c. (14 points)

For this program, assume the user inputs the line below. The '-' sign is the first character the user types. Each pair of numbers in the list "-17.17", "-2.35", and "510.99" is separated by a single space character (' '), while each pair of numbers in the list "510.99", "10.1234", and "5" is separated by two space characters.

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.

```
-17.17 -2.35 510.99  10.1234  5
```

```
int main() {
    int i1, i2;
    double d1, d2, d3;
    char ch1, ch2;
    scanf("%lf %d%lf %c%lf %c %d",
          &d1, &i1, &d2, &ch1,
          &d3, &ch2, &i2);
    printf("%d %d\n", i1, i2);
    printf("%.2lf %.3lf %.2lf\n", d1, d2, d3);
    printf("%c\n%c\n", ch1, ch2);

    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. Translate the flowchart you were given (on a separate sheet) into a complete C program. (Hint: you have to combine some logic in order to match the provided program structure.)

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

```
Enter 2 ints: -5 3           Enter 2 ints: -2 1           Enter 2 ints: 0 12
x = 1                          x = -1                          x = 0
```

```
void main() {
    int in1, in2;           // Input variables
    int x;                  // Value to be modified

    // Initialize variables as needed

    // Prompt for and read inputs
    printf("Enter 2 ints: ");

    scanf("_____", _____);

    // Test input values and perform appropriate operation on x

    x = x + 1;

    x = x - 1;

    x = 0;

    // Print value of x as formatted in test cases
    printf("_____ \n", _____);
}
```



3 (continued)

b. This program should read a time in standard form and convert it to military time. Note that:

- Standard form is hours:minutes, followed by the letter A or P for AM or PM.
- Military time is a 4-digit number between 0000 (12:00 AM) and 2359 (11:59 PM)
  - Times from 0000 to 1159 are in the morning (AM); 1200 to 2359 are afternoon or evening (PM)
  - Printing a 4-digit integer with leading 0s can be done with a precision of 4, as shown in the problem. (In other words, the output formatting is done for you.)
- If the user enters anything after the time besides 'A', 'a', 'P', or 'p', print an error message.

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

```
Enter HH:MM <A or P>: 2:25 A  
Mil. time: 0225
```

```
Enter HH:MM <A or P>: 8:16 p  
Mil. time: 2016
```

```
void main() {  
    int hr, min;    // Hours and minutes in standard form  
    char ap;       // Tracks AM or PM  
    char ch;       // Used to read : between hours & minutes  
  
    // Prompt for and read time  
    printf("Enter HH:MM <A or P>: ");  
  
    scanf("_____", _____);  
  
    // If hr is 12, set it to 0 (makes later calculations easier)  
  
    // Check if last character is A or P; convert time accordingly  
  
    printf("Mil. time: %.4d\n", _____);  
  
    printf("Mil. time: %.4d\n", _____);  
  
    printf("Invalid character\n");  
}  
}
```

3 (continued)

- c. This program prompts the user to enter two numbers representing the endpoints of a range, followed by a value to be tested. The program should check to see if the value falls within the range (including the endpoints), below the lowest endpoint, or above the highest endpoint, and print the appropriate relationship between the values, as shown in the test cases below (user input is underlined). Note: all output values should be printed with 2 decimal places.

```
Endpoints: 5 10
Value: 7
5.00 <= 7.00 <= 10.00
```

```
Endpoints: -1 1
Value: 1
-1.00 <= 1.00 <= 1.00
```

```
Endpoints: -1.2 3.4
Value: -5.6
-5.60 < -1.20
```

```
Endpoints: 2.999 3.456
Value: 4.567
4.57 > 3.46
```

```
int main() {
    double lo, hi;           // Range endpoints
    double val;             // Value

    // Prompt for and read endpoints
    printf("Endpoints: ");

    scanf("_____", _____);

    // Prompt for and read value to be tested
    printf("Value: ");

    scanf("_____", _____);

    // Test each of the three cases (in range, too low,
    // too high) and print appropriate output

    printf("_____\n", _____);

    printf("_____\n", _____);

    printf("_____\n", _____);

    return 0;
}
```