EECE.2160: ECE Application Programming

Fall 2017

Exam 1 Solution

1. (46 points) ***C input/output; operators***

a. (13 points) Show the output of the short program below 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.

```
int main () {
    double d1, d2, d3;
    int x = 4;
    d1 = x % 3 / 4.0;
    d2 = d1 * (x + 3);
    d3 = d2 - d1 * 2;

    x = d2 + d3;

    printf("%lf ", d1);
    printf("%.3lf\n", d2);
    printf("%.1lf ", d3);      // Print d3 with precision of 1
    printf("%d\n", x);

    return 0;
}
```

d1 = 4 % 3 / 4.0 = 1 / 4.0 = 0.25

d2 = 0.25 * (4 + 3) = 0.25 * 7 = 1.75

d3 = 1.75 – 0.25 * 2 = 1.75 – 0.5
= 1.25

x = 1.75 + 1.25 = 3.0 = 3

OUTPUT:

```
0.250000 1.750
1.3 3
```

Note: d3 may print as 1.2 on some machines—either answer is acceptable.

1 (continued)

b. (13 points) For this program, assume the user inputs the line below. The digit `'2'` in `2160` is the first character the user types. There is one space (`' '`) between `2160` and `8.00`, and there is one space between `8.00` and `-10.20`.

You must determine how `scanf()` handles this input and then print the appropriate results, exactly as they would be shown on the screen. The program may not read all characters on the input line, but `scanf()` will read something into all seven variables declared in the program.

```
     2160 8.00 -10.20

int main () {
    int i1, i2;
    double d1, d2;
    char ch1, ch2, ch3;

    scanf("%c %d%d%lf %c %c%lf",
       &ch1, &i1, &i2, &d1, &ch2, &ch3, &d2);

    printf("%d %d\n", i1, i2);
    printf("%.2lf %.2lf\n", d1, d2);
    printf("%c%c%c\n", ch1, ch2, ch3);

    return 0;
}
```

Solution:

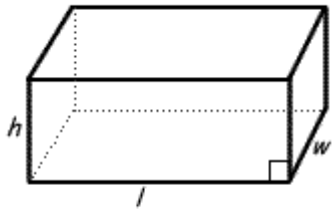
ch1 = '2' (first character)
i1 = 160
i2 = 8
d1 = .00
ch2 = '-' (first non-space character after 8.00)
ch3 = '1'
d2 = 0.20

OUTPUT:

```
160 8
0.00 0.20
2-1
```

1 (continued)

c. (20 points) Complete this program, which reads the dimensions of a rectangular prism (a box with 6 rectangular sides, as shown on the right) with length L, width W, and height H, reprints the dimensions, and calculates and prints the volume and surface area of the prism. Note that:



- The volume of a rectangular prism is the product of all three dimensions.
- The surface area is the sum of the areas of all 6 sides.

The formatting of your output should exactly match the formatting of the two test cases below, in which user input is underlined:

```
Enter L, W, H: 2 3 4
Length: 2.00
Width: 3.00
Height: 4.00
Volume: 24.0000
Surface area: 52.0000
```

```
Enter L, W, H: 1.2 3.4 5.6
Length: 1.20
Width: 3.40
Height: 5.60
Volume: 22.8480
Surface area: 59.6800
```

Students were responsible for entering bold, underlined, italicized code.

```
int main() {
    double L, W, H;       // Length, width, and height of prism

    // Prompt for and read dimensions
    printf("Enter L, W, H: ");
    scanf("%lf %lf %lf", &L, &W, &H);

    // Reprint dimensions
    printf("Length: %.2lf\n", L);
    printf("Width: %.2lf\n", W);
    printf("Height: %.2lf\n", H);

    // Calculate and print volume and surface area
    printf("Volume: %.4lf\n", L * W * H);
    printf("Surface area: %.4lf\n", 2 * (L*W + L*H + W*H));

    return 0;
}
```

2. (34 points) ***Conditional statements***

a. (14 points) For the short program shown below, the first line of output (the prompt "`Enter values: `") and the user input (`3 1`) is listed at the bottom of the page. Complete the rest of the output for this program, given those input values.

```
int main() {
   int in1, in2;

   printf("Enter values: ");
   scanf("%d %d", &in1, &in2);        in1 = 3, in2 = 1

   switch (in1 + in2) {               in1 + in2 = 4
      case 0:
         if (in1 > in2)
            printf("%d > %d\n", in1, in2);
         else
            printf("%d <= %d\n", in1, in2);
         break;

      case 2: case 4: case 6:         Program enters body of switch
                                            statement here
         if (in1 % 2 == 1)            Since in1 is odd, condition
            printf("%d is odd\n", in1);    is true

         if (in2 % 2 == 0)            Since in2 is odd, condition
            printf("%d is even\n", in2);   is false, and code in
         else                                else block executes
            printf("%d is odder still\n", in2);

      Lack of a break means program also executes default case
      default:
         printf("%d isn't 4, is it?\n", in1 + in2);
   }
   return 0;
}
```

OUTPUT (the first line is given; write the remaining line(s)):

```
Enter values: 3 1
3 is odd
1 is odder still
4 isn't 4, is it?
```

2 (continued)

b. (20 points) Complete this program, which implements a simple version of the game Yahtzee using only 3 dice. Your program should prompt for and read the 3 dice values, then check for one of the conditions below. Assume dice are entered in order from lowest to highest value.

- If all three dice match, print "3 of a kind".
- If only two of the three dice match, print "2 of a kind".
- If all three dice hold consecutive values (i.e., 3 4 5 or 1 2 3), print "Straight".
- In all other cases, print "Chance".

Three test cases are shown below, with user input underlined.

```
Enter 3 dice: 4 4 6    Enter 3 dice: 4 5 6    Enter 3 dice: 1 3 5
2 of a kind            Straight               Chance
```

Students were responsible for entering bold, underlined, italicized code.

```
void main() {
   int d1, d2, d3;        // Values of 3 dice

   // Prompt for and read dice values
   printf("Enter 3 dice: ");

   scanf("%d %d %d", &d1, &d2, &d3);

   // Check for possible results
   if (d1 == d2 && d2 == d3)
      printf("3 of a kind\n"); // All 3 dice match

   else if (d1 == d2 || d2 == d3 || d1 == d3)
      printf("2 of a kind\n"); // 2 out of 3 dice match

   else if (d1 == d2 - 1 && d2 == d3 - 1)
      printf("Straight\n");    // All 3 dice are consecutive

   else
      printf("Chance\n");      // All other cases
}
```

3. (20 points, 5 points each) ***While and do-while loops***
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 = 8;
int j = -1;
while (j < i) {
   i = i - 2;
   printf("%d %d ", i, j);
   j = j + 1;
}
```

i. 8 0 6 1 4 2

ii. 8 -1 6 0 4 1

iii. 6 0 4 1 2 2

iv. 6 -1 4 0 2 1

v. 8 6 7 5 3 0 9

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

```
int i = 17;
do {
   printf("%d ", i);
   i = i / 2;
} while (i % 2 == 0);
```

i. 17

ii. 17 8 4 2

iii. 8 4 2 1

iv. 17 8 4 2 1

v. This code sequence produces no output

3 (continued)

c. Which of the loops below will print the output: `0 1 2`?

i.
```
int i = 0;
while (i < 3) {
  printf("%d ", i);
}
```

ii.
```
int j = 2;
do {
   printf("%d ", j);
   j = j - 1;
} while (j >= 0);
```

iii.
```
int k = 9;
while (k - 1) {
   printf("%d ", k % 3);
   k = k / 2;
}
```

iv.
```
int m = 0;
do {
   printf("%d ", m);
   m = m + 1;
} while (m > 2);
```

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 next exam is as easy as this question."

All of the above are "correct."

3. (10 points) ***EXTRA CREDIT***

REMEMBER, YOU CANNOT GET EXTRA CREDIT WITHOUT WRITING AT LEAST PARTIAL SOLUTIONS FOR ALL OTHER PROBLEMS ON THE EXAM.

Write a short code sequence to read an expression of arbitrary length using integers and + and − operators, and calculate and print the result of the expression. For example, if the user enters:

```
1+2-3+4+5-6
```

your code would read that expression and print

```
Result = 3
```

You may assume:

- The shortest possible expression is a single number with no + or −. So, for example, if the user enters `15`, your code simply prints: `Result = 15`
- The only characters the user will input are digits, plus signs (`'+'`), minus signs (`'-'`) and a single newline character (`'\n'`), which indicates the end of the expression. No other characters—including spaces—will be entered, so don't account for them.
- The user always enters a valid expression, so don't account for any error cases.

If you would like additional space to write your solution, use the next page.

Solution notes:

- Because the expression is of arbitrary length, your solution must contain a loop that reads until the last character is entered. Knowing that the expression ends with a newline allows you to read until you see that character.
- Since you don't have to check for errors, having just two conditional statements that evaluate the valid inputs you need to consider inside the loop (+ and -) is sufficient. Those are the only two cases in which you read a number to add or subtract—the third possible case for your character input is the newline, which is handled in your loop condition.
- While you certainly can use more variables, you only need three—your character and two numbers. The first number keeps a running total of everything you've added or subtracted (starting with your first, and possibly only, input value) while the second number can be used to read each additional numeric input.

My solution is on the next page.

Code for extra credit solution

```
int v1, v2;  // v1 = 1st input and running total,
             // v2 = all other numeric inputs
char c;      // Input character ('+', '-', or '\n')

printf("Enter expression: ");    // Not required …

scanf("%d", &v1);      // Reads first input value

// Repeatedly read input character
// If it's '+' or '-', read number and perform operation
// If it's '\n', you've reached end of expression
do {
   scanf("%c", &c);
   if (c == '+') {
      scanf("%d", &v2);
      v1 = v1 + v2;
   }
   else if (c == '-') {
      scanf("%d", &v2);
      v1 = v1 - v2;
   }
} while (c != '\n');

printf("Result = %d\n", v1);
```