EECE.2160: ECE Application Programming

Spring 2017

Exam 1 Solution

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. ***<u>`-3 -1 1`</u>***

iv. `-3 1 -1`

v. None of the above

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. ***<u>B, C and E</u>***

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.”

<u>All of the above were “correct.”</u>

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;          d1 = 10.0 / 5.0 = 2.0

    a = d1 + d2 * 0.5;     a = 2.0 + 5.0 * 0.5 = 2.0 + 2.5
                             = 4.5 = 4 (truncated to int)

    d2 = d2 + a / d1;      d2 = 5.0 + 4 / 2.0 = 5.0 + 2.0 = 7.0

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

OUTPUT (note that both doubles print with default precision):

```
4
umber: 2.000000
7.000000
```

2 (continued)
b. (14 points)

```
int main () {
    int i;
    double d1, d2, d3;
    d1 = 15 / 3.0;              d1 = 5.0

    d2 = d1 + 2.25;             d2 = 5.0 + 2.25 = 7.25

    i = d2 - d1 + 1.23;         i = 7.25 - 5.0 + 1.23
                                  = 3.48 = 3 (truncated to int)

    d3 = 1.26 + 4 * (d2 - i);   d3 = 1.26 + 4 * (7.25 - 3)
                                   = 1.26 + 4 * 4.25
                                   = 1.26 + 17 = 18.26

    printf ("%d\n", i);
    printf ("%.2lf\n", d1);
    printf ("%.1lf\n", d2); // Print d2 with precision of 1
    printf ("%.0lf\n", d3);
    return 0;
}
```

OUTPUT:

```
3
5.00
7.2       ←7.3 is also acceptable—some systems round 7.25 up
18        ←d3 rounded to nearest integer when printed
```

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;
}
```

Solution:

```
d1 = 1.38
ch1 = '-' (first non-space character after 1.38)
d2 = 5.127 (next number after the '-')
i1 = 1 (next whole number after 5.127)
d3 = 0.63 (next real number after 1)
ch2 = '.' (first non-space character after 1.63)
ch3 = '4' (first character after '.')
```

OUTPUT:

```
1
1.380000 5.13 1
-.4
```

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

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.

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

```
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("%lf %lf %lf", &s1, &s2, &s3);

    // Test for valid triangle
    if ((s1 + s2 > s3) && (s1 + s3 > s2) && (s2 + s3 > s1)) {
     printf("Valid triangle\n");

     // Test for right triangle
     sq1 = s1 * s1;
     sq2 = s2 * s2;
     sq3 = s3 * s3;

     if ((sq1 + sq2 == sq3) ||
          (sq1 + sq3 == sq2) ||
          (sq2 + sq3 == sq1))
          printf("Right triangle\n");
    }
    // Account for cases that aren't valid triangles
    else
     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: 0 0.05     Min/max snow: 0.5 2.9    Min/max snow: 5 7
Still open               Parking ban              Parking ban
                         Still open               Fine, we'll close
```

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

```
void main() {
  double min, max;  // Minimum/maximum projected snowfall

  printf("Min/max snow: ");         // Prompt for and read range

  scanf("%lf %lf", &min, &max);

  // Error test

  if (min > max)
    printf("You didn't follow directions!\n");

  // Test min/max & print messages (may print > 1 message)
  else {
    if (min > 0.1)
      printf("Parking ban\n");

    if (max < 3)
      printf("Still open\n");

    else if ((min >= 3) && (max <= 6))
      printf("Delay\n");

    else
      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        Enter major: EE         Enter major: ChE
Computer                Electrical              Chemical
```

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

```
void main() {

    char c1, c2, c3;      // Characters to hold major

    // Prompt for and read major
    printf("Enter major: ");

    scanf(" %c%c%c", &c1, &c2, &c3);

    // Test 1st character (and others, if needed) to pick major
    switch (c1) {
     case 'E':
          printf("Electrical\n");
          break;
     case 'C':
          if (c2 == 'p')
               printf("Computer\n");
          else if (c2 == 'h')
               printf("Chemical\n");
          else if (c2 == 'E')
               printf("Civil/Environmental\n");
          break;
     case 'M':
          printf("Mechanical\n");
          break;
     case 'P':
          printf("Plastics\n");
          break;         // Not strictly necessary

    }
}
```