16.216: ECE Application Programming

Fall 2011

Exam 2 Solution

1. (24 points, 6 points per part) ***Multiple choice***

For each of the multiple choice questions below, clearly indicate your response by circling or underlining the choice you think best answers the question.

a. You are given the following short program:

```
int main() {
     int i;
     for (i = 0; i < 10; i++) {
          if (i == 0)
               continue;
          printf("i = %d\n", i);
     }
     printf("Loop done\n");
}
```

What is the first line of output this program prints?

i. `i = 0`

ii. `i = 1`

iii. `Loop done`

iv. Program prints no output—it exits before it prints anything.

1 (cont.)
b. You are given the code snippet below:

```
int y = 25;
printf("%d ", ++y);
printf("%d ", y += 4);
printf("%d\n", y++);
```

What does this code print, and what is the final value of `y`?

i. Output: `25 30 30`
Final value of `y`: `31`

ii. Output: `26 30 31`
Final value of `y`: `31`

iii. Output: `26 30 31`
Final value of `y`: `32`

iv. Output: `26 30 30`
Final value of `y`: `30`

v. ***Output: `26 30 30`***
Final value of `y`: `31`

1 (cont.)

c. Given the following code snippet:

```
int x = 100;
for (i = 1; i < 5; i++) {
   x = x – 10;
}
```

Which of the following choices can replace the `for` loop and produce the exact same value for `x`? Assume `x` is always initialized to 100.

i.
```
i = 0;
while (i < 5) {
   x = x – 10;
   i++;
}
```

ii.
```
i = x;
while (i > 60) {
   i = i – 10;
}
```

iii.
```
i = 0;
while (i < 8) {
   x += (-10);
   i += 2;
}
```

iv.
```
i = 5;
while (i >= 1) {
   x = x – 10;
   i--;
}
```

1 (cont.)

d. Say we have a function, declared as follows:

```
void foo(int *x, int *y);
```

If your program contains two integers, `a` and `b`, which of the choices below correctly calls `foo` and passes the addresses of `a` and `b` to that function?

i.
```
foo(a,b);
```

ii.
```
foo(*a,*b);
```

iii.
```
int *ptr = &a;
foo(*ptr, b);
```

iv.
```
int *ptr = &a;
foo(ptr, &b);
```

2. (40 points) ***Functions and pointers***

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.

I encourage you to 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;
     int *p1, *p2;
     v1 = v2 = 0;
     p1 = &v2;
     p2 = p1;
     v2 = 16;
     *p2 = 24;
     v1 = *p1 + 16;
     *p1 = *p2 – 10;
     printf("%d %d\n", v1, v2);
     printf("%d %d\n", *p1, *p2);
     return 0;
}
```

Solution:

```
     40 14

     14 14
```

2 (cont.)

b. (12 points)

```
double f(double x, double y) {
     y *= 2;
     x -= 3;
     return (x + y) / 2.0;
}
int main() {
     double q, r, s;

     q = f(5, 8);
     r = f(8, 5);
     s = f(q, r);

     printf("%.2lf %.2lf %.2lf\n", q, r, s);

     return 0;
}
```

Solution

```
     9.00 7.50 10.50
```

2 (cont.)

c. (16 points)

```
int f1(int *arg1) {
     (*arg1)++;
     return (*arg1) * 2;
}

int f2(int arg2) {
     return f1(&arg2) + 10;
}

int f3(int *arg3) {
     return f1(arg3) + 10;
}

int main() {
     int a, b, c;
     int x, y, z;

     a = b = c = 10;     // Set all three values to 10

     x = f1(&a);
     y = f2(b);
     z = f3(&c);

     printf("%d %d %d\n", a, b, c);
     printf("%d %d %d\n", x, y, z);

     return 0;
}
```

Solution:

```
     11 10 11
     22 32 32
```

3. (36 points, 18 per part) ***Loops***

For each part of this problem, you are given a short program to complete. Note that some of the code is provided for you. **CHOOSE ANY TWO OF THE THREE PARTS** and fill in the spaces with appropriate code. **If you complete all three, I will grade only the first two.**

a. The program below should repeatedly prompt the user to enter an integer value, then read that input value (`inval`), which should be handled as follows:

- If the value is 0, print "Success".
- If the value is -1, print "Done!" and end the program.
- If the user enters any other value, print "Incorrect input".

Sample run (with input underlined):

```
Enter #: 0
Success
Enter #: 3
Wrong input
Enter #: -1
Done!
```

Boldfaced, underlined code was to be filled in by student

```
void main() {
     int inval;      // Input value

     do {
          printf("Enter #: ");       // Print prompt and read input
          scanf("%d", &inval);

          if (inval == 0)
               printf("Success\n");       // Output if inval is 0

          else if (inval == -1) {
               printf("Done!\n");         // Output if inval is -1
               // NOTE: Many people chose to use break here
               //  with "while(1)" as loop condition—that also works

          }

          else
               printf("Wrong input\n");   // Output in other cases

     } while (inval != -1);
}
```

3 (cont.)

b. The program below reads three values as shown. It should then check all values between 0 and `max`, incrementing by `inc`, and print those values that are divisible by `div`. When done, it should print the total count of values that meet the condition.

See the sample run below; user input is underlined.

```
Enter max, inc, div: 12 2 3
0 is divisible by 3
6 is divisible by 3
12 is divisible by 3
Total # values: 3
```

Note: If 12, 2, and 3 are the inputs, you should produce the values: 0, 2, 4, 6, 8, 10, 12. Three of those values are divisible by 3: 0, 6, and 12. Those values are printed, and your count is incremented each time so it can be correctly printed at the end.

Boldfaced, underlined code was to be filled in by student

```
void main() {
     int max;         // Maximum value to test
     int inc;         // Amount to increment by
     int div;         // Divisor
     int count;       // # of values that meet condition
     int i;           // Loop variable

     printf("Enter max, inc, div: ");           // Prompt for and
     scanf("%d %d %d", &max, &inc, &div);       //   read inputs

     count = 0;
     for (i = 0; i <= max; i += inc) {

          if ((i % div) == 0) {

               // Found value that's divisible by div--print it
               printf("%d is divisible by %d\n", i, div);

               count++;
          }
     }

     printf("Total # values: %d\n", count);     // Print final results
}
```

3 (cont.)

c. The program below should repeatedly do the following:

- Prompt for and read an integer, num, then prompt for and read num different values.
- Sum all of those values, then print both the sum and the average.
- Ask the user to enter any character to repeat, or enter 'X' to exit.
  - If the user enters anything but 'X', repeat the above steps, starting at the prompt for num.

Sample run below (user input underlined; both columns are from a single run of the program):

```
Enter number of values: 3          Enter number of values: 2
Enter value: 1                     Enter value: 7
Enter value: 2                     Enter value: 8
Enter value: 3                     Sum: 15  Average: 7.5
Sum: 6  Average: 2.0               Enter X to exit: X
Enter X to exit: a
```

Boldfaced, underlined code was to be filled in by student

```
void main() {
     double sum;           // Sum of all values read
     double inval;         // Input value read
     int num;              // # of values to be read
     char exit;            // Character controlling exit from program
     int i;                // Loop variable

     do {
          printf("Enter number of values: ");   // Prompt for and
          scanf("%d", &num);                    //   read # of values

          sum = 0;
          for (i = 0; i < num; i++) {

               printf("Enter value: ");         // Prompt for and
               scanf("%lf", &inval);            //   read one input

               sum += inval;
          }

          printf("Sum: %.0lf  Average: %.1lf\n", sum, sum / num);
          printf("Enter X to exit: ");
          scanf("\n%c", &exit);

     } while (exit != 'X');
}
```