CSE142 Exam with answers

Midterm #1 Spring 2001
Problem numbering may differ from the test as given.

All multiple choice questions are equally weighted. You can generally assume that code shown in the questions is intended to be syntactically correct, unless something in the question or one of the answers suggests otherwise.

1. **You are modifying a program someone else wrote. The original programmer used a variable called "month" to store a number (1-12) representing the current month of the year. You wish to change the program so that instead of a number, the variable contains the name of the month, for example, "January" instead of 1.**

   What are the most likely declarations of "month" in the old and new versions of the program?

   A. /*old*/ int month;
   /*new*/ char month;
   B. /*old*/ double month;
   /*new*/ char month;
   C. /*old*/ int month;
   /*new*/ int January;
   D. /*old*/
   #define month 1
   /*new*/
   #define month January
   E. None of the above. We don't yet know a way to store a whole English word like "January."
   ANSWER: **E**
   A char in C can hold only a single character. Later we will use null-terminated strings, which are arrays of chars.

2. **x is a variable of type double. It holds a value already. Which expression can be used to round off the value contained in x?**

   Use the usual mathematical notion of rounding.

   Reminder: some examples of rounding:
   3.0 rounded is 3
   3.14 rounded is 3
   3.49999 rounded is 3
   3.50 rounded is 4
   3.51 rounded is 4

   A. (int)(x - 1)
   B. (int)(x + 0.5)
   C. x - x % 0.5
   D. (double)%.0x
   E. x - ((int) x)
   ANSWER: **B**

3. Consider this program fragment:

```
int count = 5;
int x = 30;
double y = 0.0;

y =  ((double) count) / x;    /* line A*/
```

What is true about line A?

i. Line A contains a cast.
ii. As a result of this statement, count will change in value.
iii. As a result of this statement, y will change in value.

A. i. only
B. ii. only
C. iii. only
D. i and iii only
E. i., ii. and iii are all true
ANSWER: **D**

4. Consider the following code fragment:

```
int my_function(int a, int b) {
  a = 2 * a + b;

  return a;
}

int main(void) {
  int a = 5;
  int b = 3;

  b = my_function(b, a);      /* line Z */
...
}
```

What is the value of a after executing line Z?

A. 16
B. 13
C. 11
D. 5
E. 0
ANSWER: **D**
Some students wondered "Which a? There are two of them." First hint should be that Line Z uses the a of main, so that a is the most natural one to examine. Beyond that, the a of my_function no longer is in memory. Local variables and parameters are allocated when the function is entered and deallocated when the function exits. The a of my_function doens't exist after the call in line Z (right-hand side of the assignment statement) executes.

5. In the program from the previous question, what is the value of b after executing line Z?

A. 16
B. 13
C. 11
D. 5
E. 0
ANSWER: **C**

6. Where should a local variable be declared in a C program?

A. At the beginning of the function in which you use it.
B. Right before it is initialized.
C. Variables in C do not actually have to be declared, although it is good style to do so.
D. Near the top of the program, after the #defines and before main.
E. On the left-hand side of an assignment statement.
ANSWER: **A**

7.
```
double income = 35000;          /* line A */

printf("Enter new income: ");
scanf("%lf", &income);
```

Line A is which of the following?
I. Assignment statement
II. Initialization of income
III Declaration of income

A. II only
B. III only
C. I, III only
D. II, III only
E. I, II, and III
ANSWER: **D**
See course packet slides C-19 and E-26.

8. When expressions are evaluated, rules of precedence and associativity determine the order in which various operations happen. Complete the following sentence:

The C expression
-1 + 4 * 5
is evaluated as if it were written...

A. - ((1 + 4) * 5), by precedence
B. (-1) + (4 * 5), by precedence
C. (-1) + (4 * 5), by associativity
D. (-(1) + 4) * 5, by precedence
E. (-(1) + 4) * 5, by associativity
ANSWER: **B**

9. Suppose you encounter two unknown binary operators, ^ and $ in a program. Suppose you look up in a reference manual and find that ^ has higher precedence, and that both operators are right-associative. What is the correct way to evaluate the following expression?

a ^ b ^ c $ d ^ e

A. ((a ^ b) ^ c) $ (d ^ e)
B. ((a ^ b) ^ (c $ d)) ^ e
C. a ^ ((b ^ (c $ d)) ^ e)
D. (a ^ (b ^ c)) $ (d ^ e)
E. a ^ ((b ^ c) $ (d ^ e))
ANSWER: **D**

10. Trace this code carefully and determine what the final values of y and z are:

```
int x = 5;
int y = 10;
int z = 11;

if( x < 10)
{
   y = 5 + y + z;
}
else
{
  y = 5;
  x = 7;
}

z = x + y;
y = z + x;
```

A. 36, 31
B. 15, 10
C. 19, 12
D. 10, 15
E. 10, 11
ANSWER: **A**

11. Which are NOT valid (syntactically) as variable names in C?

I. Fortythird_President
II. C++
III. int
IV. face2face

A. I only
B. II only
C. III only
D. IV only
E. more than one is invalid
ANSWER: **E**

12. A computer has a DVD disk and a digital TV attached as I/O. A program is written to play DVD movies on the TV. What is true for such a program?

A. The TV is a input device
B. The TV is an output device
C. The DVD is an input device
D. The DVD is an output device

A. A and C only
B. A and D only
C. B and C only
D. B and D only
E. All of A, B, C, and D
ANSWER: **C**

13. Comparing #define symbols and variables, what is true?

A. Varables names and #define symbols are identifiers
B. Variables and #define symbols can both change their values as the program executes
C. Variable names and #define symbols can both appear in expressions on the right-hand side of an assignment statement

A. A only
B. B only
C. C only
D. A and B
E. A and C
ANSWER: **E**

14. Which function header matches the description "the function named mapper takes two chars and returns an int"?

A. (char, char) mapper (int X)
B. char mapper (char X, int Y)
C. int mapper (char X, char Y)
D. void mapper (char X, char Y, int Z)
E. mapper (char1, char2, return int)
ANSWER: **C**

15. What is the value of this expression in C?

10 / 3 * ((int) 3.14159) % 5

A. 0
B. 1
C. 2
D. 3
E. 4
ANSWER: **E**

16. [Worth 5 multiple choice questions]

Stevie's Carpet Warehouse is running a Friday the 13th Sale. Stevie will install carpet in any size room, for 12.99 per square foot, plus a $100.00 installation charge. However, if the room is big enough, the installation is free. "Big enough" means that a piece of carpet 12 feet by 15 will fit in the room without being cut.

Write a program which computes the total cost of the carpet. The program displays the area, the total cost and also states whether or not installation was free. The user enters the width and length of the room. Width is never greater than length.

Sample run (display this information, though it doesn't have to follow this exact format):

```
Enter room length: 16.5

Enter room width: 8.2

No free installation.
Your area: 135.30. Your price: $1857.55
```

Another sample:

```
Enter room length: 16.1

Enter room width: 14.0

You get free installation!
Your area: 225.40. Your price: $2927.95
```

The program should be as complete and ready to compile as you can make it. Use your best style, except that comments may be kept short to save time.

Continue your answer on the next page if you run out of room.

```
#include <stdio.h>



int main (void) {









    return 0;
}
```

```
/************************* sample solution ******************/

#include <stdio.h>

#define STANDARD_LENGTH 15
#define STANDARD_WIDTH  12
#define COST_PER_SQ_FOOT  12.99
#define INSTALL_FEE  100.0

int main (void) {

        double rlength, rwidth; /*room length and width, entered by user*/
                        /*one may assume width <= length*/

        double area;  /*square footage being installed*/
        double basecost; /*cost based on sq. foot */
        double finalcost; /*cost including installation fee*/
        double installationcost = INSTALL_FEE;

        printf ("\nEnter room length: ");
        scanf ("%lf", &rlength);
        printf ("\nEnter room width: ");
        scanf ("%lf", &rwidth);

        area = rlength * rwidth;

        basecost =  area * COST_PER_SQ_FOOT;

        if (rlength >= STANDARD_LENGTH) {
                if (rwidth >= STANDARD_WIDTH) {
                        printf ("\nYou get free installation!");
                        installationcost = 0.0;
                }
                else
                printf ("\nNo free installation.");
        }
        else
                printf ("\nNo free installation.");


        finalcost = basecost + installationcost;

        printf ("\nYour area: %6.2f. Your price: $%6.2f", area, finalcost);

        return 0;

}
```

Notes: **Some of the principal things we looked for when grading:**

Style and structure
- **use of #define's**
- **comments (not long, but at least one somewhere!)**
- **good identifier choices**
- **indenting on if/else**

Correct operation
- **appropriate types for variables**
- **initial printfs for prompts**

scanfs (incl. thje &)
area calculation
cost based on sq. footage
size check logic (major points)
install cost added or not
"free install" printf
final answer printf

One of the most common errors was in deciding whether or not a 12x15 rug would fit in the room. Here's a wrong way to do it:
if (area >= 12*15) { /*free installation ...*/

Aside from the magic numbers 12 and 15 appearing (which would be points off separately for style), the logic is wrong. Imagine a room 10x20. This is >=180, but a 12x15 rug will not fit in it without being cut. So, no free installation.