CISC105 Midterm 1 Sample Questions from Previous Exams

NAME _____

General instructions:

There are problems worth a total of points . Read the problems very carefully. Identify what kind of answer the problem asks for. Carefully look at requirements for input and output, and any restrictions on how your code must be written.

You may assume that input will not produce errors for the procedures described, unless the questions says otherwise.

Do problems you are confident about first. If you finish the problems you know, write what you do know about other problems to gain partial credit; but erroneous information may detract from that credit or irritate the grader, so don't make stuff up.

Do not do unnecessary testing. For example, testing for both `x < 0` and `x >= 0` instead of using one test and then `else` would be considered unnecessary testing.

Do not make code unnecessarily inefficient.

Short answer. Use only the space provided.

1. What values are true in C?

2. What does the C statement "int q;" do inside the computer, as discussed in class?

3. Is #include a reserved word in C? Explain your answer very briefly.

4. When would you use a sentinel-controlled loop?

5. (6 pts) Write two complete examples, one of each of the preprocessor directives we have reviewed in class.

6. What is the main issue, as discussed in class, that you must consider when choosing a sentinel?

7. Structured programming says that all programs can be written using a combination of three kinds of control structures. What are they?

8. (4 pts) When you load a C program, what physical part of the computer does it *come from*?

9. (4 pts) When you load a C program, what physical part of the computer does it *go to*?

10. (6 pts) What are the structures we have studied in class so far for accomplishing selection of code in C?

11. (12 pts) Write the appropriate shell command next to each task:
    (a) compile a C program
    (b) list the contents of a file
    (c) show the location of the current directory
    (d) create a new directory
    (e) make a copy of a file

12. (4 pts) Advocates of this philosophy proved that all programming could be done using only sequence, selection, and repetition. What is its name?

13. 12 pts. Write the six stages of a C program in order. Next to each, write the shell command that accomplishes it.

Short Answers

Consider the **partial** program below. Imagine that you are going to add some functions to it. Make up function names if needed, and use parameters from the existing code.

```
#include <stdio.h>

/* Terry Harvey CISC105-20 TA: Liric Waterchard*/

int main(){

    int i, j;
    double x;
    ... /* more code here */
```

14. (4 pts) Write a prototype for a function that can be called from main() to print the value in i. Do not write the call or definition.

15. (4 pts) Show how you could call the function from 14 in main().

16. (4 pts) The function **sum** will take an integer and a double parameter and returns the sum of the two numbers. Write the **prototype only** for sum.

17. (4 pts) Show how you could call the function from 16 in main().

18. (10 pts) Write a definition for a function that takes a parameter that represents the number of asterisks in a line. The function prints that number of asterisks only.

19. (6 pts) Explain how C would go about evaluating the following expression: `4 < x < 9`

For the next section, choose the letter that shows the value that would be in x. Assume x and y are declared as integers. Choose "error" if you think a fragment will not compile and run. The fragments are unrelated (each question is separate from the others). HINT: Calculate your answer before you look at the selections.

20. (2 pts)

```
x = 2 - 3 * 2 - 7;
```

(a) `5` (b) `11` (c) `2` (d) `error` (e) none of the above

21. (2 pts)

```
x = 2 + 3 || 2 < 3;
```

(a) `5` (b) `0` (c) `1` (d) `error` (e) none of the above

22. (2 pts)

```
y = 3.6;
x = (double)y * 2;
```

(a) `7.2` (b) `7` (c) `6` (d) `error` (e) none of the above

23. (2 pts)

```
x = 15 % 3;
```

(a) `5` (b) `1` (c) `3` (d) `error` (e) none of the above

24. (2 pts)

```
x = 6 % 7;
```

(a) `1` (b) `7` (c) `6` (d) `error` (e) none of the above

25. (2 pts)

```
x = 3 / 4 && 4 / 3;
```

(a) `0` (b) `1` (c) `12` (d) `error` (e) none of the above

26. (2 pts)

```
y = 30 == 3;
x = !y;
```

(a) `1` (b) `0` (c) `27` (d) `-1` (e) error

27. (2 pts)

```
x = (8 > 5 > 4);
```

(a) `false` (b) `1` (c) `0` (d) `error` (e) none of the above

28. (2 pts)

```
y = 0 >= 0;
x = !y;
```

(a) `1` (b) `-1` (c) `0` (d) `error` (e) none of the above

29. (2 pts)

```
x = 8;
x *= 2;
```

(a) `4` (b) `8` (c) `16` (d) `error` (e) none of the above

30. (2 pts)

```
x = 0 / 10110110;
```

(a) `1` (b) `6` (c) `0` (d) `error` (e) none of the above

31. (2 pts)

```
x = 11 / 3;
```

(a) `3` (b) `2` (c) `3.333` (d) `error` (e) none of the above

32. (2 pts)

```
y = 15 / 16;
x = 16 / y;
```

(a) `15` (b) `1` (c) `pi` (d) `error` (e) none of the above

33. (2 pts)

```
double z = 8;
x = 8 || z/12;
```

(a) `0` (b) `1` (c) `0.66` (d) `error` (e) none of the above

34. (2 pts)

```
double z = 10;
x = z / 4.0;
```

(a) `.5` (b) `2.5` (c) `2` (d) `error` (e) none of the above

35. (2 pts)

```
x = 3 < 2;
x++;
```

(a) `0` (b) `4` (c) `3` (d) `error` (e) none of the above

36. (2 pts) Which of the following assigns the value of 2 times y to x?

(a)`x == 2 * y;` (b) `2 * y = x;` (c) `x = 2 x y;` (d) `x = 2 * y;`

37. (2 pts) The load stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

38. (2 pts) The link stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

39. (2 pts) The preprocess stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

40. (2 pts) The execute stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

41. (2 pts) The edit stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

42. (4 pts) Which of the following correctly orders memory speed from slowest to fastest?

    (a) disk cache RAM registers
    (b) disk RAM cache registers
    (c) RAM disk cache registers
    (d) cache RAM disk registers
    (e) none of the above

43. (3 pts) Which kind of memory is built into the CPU?

    (a) registers
    (b) ALU
    (c) cache
    (d) RAM
    (e) none of the above

    Use these choices to relate classroom discussions to the next four questions:

    (a) measured in Kilobytes
    (b) extremely fast, very high cost per byte
    (c) a chip connected to the CPU via the bus
    (d) very slow, but lowest cost per byte
    (e) none of the above

44. (3 pts) The best description to associate with disk is:

45. (3 pts) The best description to associate with cache is:

46. (3 pts) The best description to associate with ALU is:

47. (3 pts) The best description to associate with RAM is:

    For the next section, choose the letter that shows the value that would be in x. Assume x and y are declared as integers. Choose "error" if you think a fragment will not compile and run. The fragments are unrelated (each question is separate from the others).

48. (2 pts)

```
x = 3 + 3 * 2 - 7;
```

    (a) `5` (b) `11` (c) `2` (d) `error` (e) none of the above

49. (2 pts)

```
x = -4 || 0;
```

    (a) `0` (b) `-4` (c) `-1` (d) `error` (e) none of the above

50. (2 pts)

```
y = 0;
x = y++;
```

(a) `1` (b) `2` (c) `0` (d) `error` (e) none of the above

51. (2 pts)

```
x = 5 % 3;
```

(a) `2` (b) `1` (c) `3` (d) `error` (e) none of the above

52. (2 pts)

```
x = 2 % 3;
```

(a) `1` (b) `0` (c) `3` (d) `error` (e) none of the above

53. (2 pts)

```
x = 14 % 7;
```

(a) `0` (b) `2` (c) `4` (d) `error` (e) none of the above

54. (2 pts)

```
y = 30 != 3;
x = -3 && !y;
```

(a) `1` (b) `0` (c) `27` (d) `-3` (e) error

55. (2 pts)

```
x = (8 < 15 < 4);
```

(a) `false` (b) `1` (c) `0` (d) `error` (e) none of the above

56. (2 pts)

```
y = 0 >= 1;
x = !y;
```

(a) `1` (b) `-1` (c) `0` (d) `error` (e) none of the above

57. (2 pts)

```
x = 8;
x += 2;
```

(a) `2` (b) `8` (c) `6` (d) `error` (e) none of the above

58. (2 pts)

```
x = 0 / 6;
```

(a) `1` (b) `6` (c) `0` (d) `error` (e) none of the above

59. (2 pts)

```
x = 14 / 4;
```

(a) `3` (b) `2` (c) `3` (d) `error` (e) none of the above

60. (2 pts)

```
y = 5 / 6;
x = 2 / y;
```

(a) `5` (b) `11` (c) `-8` (d) `error` (e) none of the above

61. (2 pts)

```
y = 4.4;
x = y * 2;
```

(a) `8` (b) `8.8` (c) `2` (d) `error` (e) none of the above

62. (2 pts)

```
double z = 8;
x = z/12;
```

(a) `1.5` (b) `1` (c) `0.75` (d) `error` (e) none of the above

63. (2 pts)

```
double z = 18;
x = z / 12.0;
```

(a) `1.5` (b) `1` (c) `0.75` (d) `error` (e) none of the above

64. (2 pts)

```
x = 3 < 2;
x = 3 / x;
```

(a) `0` (b) `1` (c) `3` (d) `error` (e) none of the above

65. 26 pts. Write the value of x after each of the following code fragments, OR write ERROR if you think a fragment will not compile and run. The fragments are unrelated. Assume x and y are declared as integers. If you write calculations, be sure the eventual answer is circled so that I can find it.

(a)
```
x = 3 + 4 * 2 - 12;
```

(b)
```
x = 3 || 4;
```

(c)
```
y = 3 != 3;
x = 3 && y;
```

(d)
```
x = (3 < 5 < 2);
```

(e)
```
y = 0 >= 1;
x = !y;
```

(f)
```
x = 5 % 3;
```

(g)
```
x = 8;
x += 2;
```

(h)
```
x = 11 / 6;
```

(i)
```
y = 5 / 6;
x = 2 / y;
```

(j)
```
y = 2.5;
x = y * 2;
```

(k)
```
y = 8;
x = --y;
```

(l)
```
x = 5 + 2 * - 3;
```

(m)
```
x = 3 < 3 || 4 < 5;
```

(n)
```
2 * y = 8;
x = y == y;
```

(o)
```
x = (5 > 4 < 3);
```

(p)
```
x = 8 % 2;
```

(q)
```
x = 5 % 3;
```

(r)
```
x = 3 % 5;
```

(s)
```
x = 4;
x += 5;
```

(t)
```
x = 15 / 6;
```

(u)
```
x = 5 / 0;
```

(v)
```
x = 0 / 5;
```

66. 40 pts. Write code fragments that will generate *exactly* the output shown. Follow the instructions carefully. DECLARE ANY VARIABLES YOU NEED. You may write next to the output so that you have more space.

    (a) Write a while loop to generate this output:

    ```
    1
    2
    3
    4
    ```

    (b) Write a while loop to generate this output:

    ```
    2 4 6 8
    ```

    (c) Write a loop to generate this output:

    ```
    -50 0 50 100 150
    ```

    (d) Write a single loop to generate this output. Print only a *single* asterisk at a time. You may use only two printfs total.

```
***
***
***
```

(e) Write a loop to generate this output:

```
15 10 5 0 -5
```

67. 15 pts. Write a whole program that prompts the user for a value, reads the value into a variable, then prints the number out and stops.

68. 20 pts. Write a switch statement that will print "green" for input integers 0-3; "red" for 4-7; "yellow" for 8-11; and "purple" for any other integer. You may not use more than four different cases. Assume the input is already in the declared integer variable "input".

69. 20 pts. Write a whole program using `if`, `else`, and other C code so that it behaves as follows:

```
shell% a.out
Enter a positive integer: 7
7 is less than 10!
Enter a positive integer: 8
8 is less than 10!
Enter a positive integer: 11
11 is not less than 10.
Enter a positive integer: -1
bye!
shell%
```

70. 30 pts. Write a whole program that behaves as follows: Prompt the user for an integer, one thousand times. Then show the user the average of those numbers, the highest number, and the lowest number.

71. 15 pts. Write a main() function that

    (a) declares two variables

    (b) initializes them to the numbers 4.7 and 7.8

    (c) adds them together

    (d) prints the sum of the two

    (e) stops.

72. 30 pts. Write a whole program that behaves as follows: Prompt the user for an integer, one hundred times. Each time through the loop, show the highest number so far, the lowest number so far, and the average of all input so far.

73. 20 pts. Write a switch statement with no more than five cases that behaves as follows. READ THE OUTPUT VERY CAREFULLY! The results are not an obvious pattern. All possible cases are shown in the output.

    Assume the input is already in the declared integer variable "input".

```
strauss% a.out
Enter a positive integer, or -1 to stop:
5
5 is less than 20
Enter a positive integer, or -1 to stop:
35
35 is less than 40
Enter a positive integer, or -1 to stop:
55
55 is less than 60
Enter a positive integer, or -1 to stop:
61
61 is less than 100
Enter a positive integer, or -1 to stop:
85
85 is less than 100
Enter a positive integer, or -1 to stop:
106
106 is equal to or more than 100
Enter a positive integer, or -1 to stop:
-1
strauss%
```

74. 15 pts. Write a whole program that prompts the user for a value, reads the value into a variable, then prints the number out and stops.

75. 15 pts. Write a main() function that

    (a) declares two variables

    (b) initializes them to the numbers 4.7 and 7.8

    (c) adds them together

    (d) prints the sum of the two

    (e) stops.

76. 24 pts. Short answer. Use only the space provided.

    (a) What values are false in C?

    (b) What values are true in C?

    (c) Write a complete example of a preprocessor statement.

    (d) What are the structures we have studied in class so far for accomplishing repetition of code in C?

(e) Structured programming says that all programs can be written using a combination of three kinds of control structures. What are they?

(f) Of the six stages of a C program discussed in class, which stages are accomplished by the shell command "gcc lab1.c"?

(g) When you load a C program as discussed in class, what physical part of the computer does it go to?

77. 20 pts. Write the value of x after each of the following code fragments, OR write ERROR if you think a fragment will not compile and run. The fragments are unrelated. Assume x and y are declared as integers.

(a)
```
y = 5;
x = y++;
```

(b)
```
x = 3 + 4 * 2 - 12;
```

(c)
```
x = 3 || 4;
```

(d)
```
y = 3 != 3;
x = 3 && y;
```

(e)
```
x = (3 < 5 < 2);
```

(f)
```
y = 0 >= 1;
x = !y;
```

(g)
```
x = 5 % 3;
```

(h)
```
x = 8;
x += 2;
```

(i)
```
x = 11 / 6;
```

(j)
```
y = 5 / 6;
x = 2 / y;
```

78. 32 pts. Write code fragments that will generate *exactly* the output shown. Follow the instructions carefully. DECLARE ANY VARIABLES YOU NEED. You may write next to the output so that you have more space.

(a) Write a loop to generate this output:

```
1 2 3 4
```

(b) Write a loop to generate this output:

```
3
2
1
0
```

(c) Write a single loop to generate this output:

```
*****
*****
*****
*****
*****
```

(d) Write a loop to generate this output:

```
-50 0 50 100 150
```

79. 20 pts. Write a switch statement that will print "green" for input integers 0-3; "red" for 4-7; "yellow" for 8-11; and "purple" for any other integer. You may not use more than four different cases. Assume the input is already in the declared integer variable "input".

80. 20 pts. Write a whole program that behaves as follows: Prompt the user for an integer, one thousand times. Then show the user the average of those numbers, the highest number, and the lowest number.

81. Suppose you wanted to create a directory called "labs" to put all of your lab code in. Show how to create this directory.

60 pts. Multiple Choice: use the scan sheet for your answers.

82. (2 pts) Which of the following assigns the value of 2 times y to x?

(a)`x == 2 * y;` (b) `2 * y = x;` (c) `x = 2 x y;` (d) `x = 2 * y;`

83. (2 pts) The load stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

84. (2 pts) The link stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

85. (2 pts) The preprocess stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

86. (2 pts) The execute stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

87. (2 pts) The edit stage of a C program is accomplished by which command?

(a) `emacs` (b) `a.out` (c) `pwd` (d) `gcc` (e) none of the above

For the next section, choose the letter that shows the value that would be in x. Assume x and y are declared as integers. Choose "error" if you think a fragment will not compile and run. The fragments are unrelated (each question is separate from the others).

88. (2 pts)

```
x = 3 + 3 * 2 - 7;
```

(a) `5` (b) `11` (c) `2` (d) `error` (e) none of the above

89. (2 pts)

```
x = -4 || 0;
```

(a) `0` (b) `-4` (c) `-1` (d) `error` (e) none of the above

90. (2 pts)

```
y = 0;
x = y++;
```

(a) `1` (b) `2` (c) `0` (d) `error` (e) none of the above

91. (2 pts)

```
x = 5 % 3;
```

(a) `2` (b) `1` (c) `3` (d) `error` (e) none of the above

92. (2 pts)

```
x = 2 % 3;
```

(a) `1` (b) `0` (c) `3` (d) `error` (e) none of the above

93. (2 pts)

```
x = 14 % 7;
```

(a) `0` (b) `2` (c) `4` (d) `error` (e) none of the above

94. (2 pts)

```
y = 30 != 3;
x = -3 && !y;
```

(a) `1` (b) `0` (c) `27` (d) `-3` (e) error

95. (2 pts)

```
x = (8 < 15 < 4);
```

(a) `false` (b) `1` (c) `0` (d) `error` (e) none of the above

96. (2 pts)

```
y = 0 >= 1;
x = !y;
```

(a) `1` (b) `-1` (c) `0` (d) `error` (e) none of the above

97. (2 pts)

```
x = 8;
x += 2;
```

(a) `2` (b) `8` (c) `6` (d) `error` (e) none of the above

98. (2 pts)

```
x = 0 / 6;
```

(a) `1` (b) `6` (c) `0` (d) `error` (e) none of the above

99. (2 pts)

```
x = 14 / 4;
```

(a) `3` (b) `2` (c) `3` (d) `error` (e) none of the above

100. (2 pts)

```
y = 5 / 6;
x = 2 / y;
```

(a) `5` (b) `11` (c) `-8` (d) `error` (e) none of the above

101. (2 pts)

```
y = 4.4;
x = y * 2;
```

(a) `8` (b) `8.8` (c) `2` (d) `error` (e) none of the above

102. (2 pts)

```
double z = 8;
x = z/12;
```

(a) `1.5` (b) `1` (c) `0.75` (d) `error` (e) none of the above

103. (2 pts)

```
double z = 18;
x = z / 12.0;
```

(a) `1.5` (b) `1` (c) `0.75` (d) `error` (e) none of the above

104. (2 pts)

```
x = 3 < 2;
x = 3 / x;
```

(a) `0` (b) `1` (c) `3` (d) `error` (e) none of the above

105. (18 pts) Write code fragments that will generate *exactly* the output shown. Follow the instructions carefully. DECLARE ANY VARIABLES YOU NEED. You may write next to the output so that you have more space.

(a) Write a loop to generate this output:

```
1
2
3
4
```

106. (15 pts) Write a function named "min" in the space provided above main(). "Min" finds the smaller of its arguments, and does nothing else. It should be designed so that it works properly with this main(). Do not modify main().

```
int main(){

  printf("The smallest of these two is: %lf", min(4.5, 7.6));

  return 0;
}
```

107. (20 pts) Write a whole program that behaves as follows: Prompt the user for two integers and print the average of the two integers.

108. (15 pts) Write a switch statement that will print "green" for input integers 0,3,6; "red" for 1,4,7; "mauve" for 2,5,8; and "blue" for any other positive integer. You may not use more than four different cases. Assume the input is already in the declared integer variable "input".

109. Extra Credit:

(a) What is the emacs keyboard shortcut to open a file?

(b) To save a file?

(c) To indent a whole region of code at once?