CSC 101 Midterm 1

Instructions: Answer all questions on the paper provided. You have 50 minutes to complete the exam. It is open book and open note. Except for Question 9, you do not need to write comments for the programming questions. For multiple choice questions, circle the single best answer.

1. *(3 points)* A _____ converts a source code program into machine code.
   a. editor
   b. compiler
   c. converter
   d. test program
   e. none of the above

2. *(3 points)* Program variables are stored in what part of the computer?
   a. CPU
   b. main memory
   c. peripheral storage
   d. operating system
   e. none of the above

3. *(3 points)* What is the value of gamma after executing the following statements? _____

```
int gamma;
int delta;
gamma = 2;
delta = 15;
gamma = gamma * delta;
```

4. *(4 points)* What does the following code print? _____

```
int x, y, z;
x = 10;
y = 20;
z = (x < y && x > y) == ((x < y) && (x > y));
printf("%d\n", z);
```

5. *(8 points)* When the user runs the following program and enters `0`, I was expecting it to print

```
The number is zero
```

It does not behave this way.

```
 1  #include <stdio.h>
 2
 3  int main () {
 4      int n;
 5
 6      printf("Input a number: ");
 7      scanf("%d", &n);
 8
 9      if (n < 0) printf("The number is negative\n");
10      else if (n = 0) printf("The number is zero\n");
11      else printf("The number is positive\n");
12
13      return 0;
14  }
```

What does the program print after the user enters `0`?

a. Nothing, because the program logic never gets to any print statement.

b. `0`

c. `The number is negative`

d. `The number is positive`

e. None of the above.

What is the easiest way to fix the code so it does behave like I expect, i.e., it prints "`The number is zero`" when the user enters `0`. State your answer by giving the number of the line or lines that need to change, and what the change should be.

6. *(10 points)* What does the following code print?

```
int a;
int b;
int c;
int d;

a = 2;
b = -4;
c = a;
d = 0;

if ((a < 2 && b == 0) || c != b) {
    printf("One\n");
}

if (c == a && b <= -4 && !d) {
    printf("Two\n");
}

printf("Three\n");

if (a < 0 && a / d < 0) {
    printf("Four\n");
}
else {
    printf("Five\n");
    printf("Six\n");
}

if (b + 2 * a == 0)
    printf("Seven\n");
printf("Eight\n");

if (a * b < 0) {
    if (c == 'd') {
        printf("Nine\n");
    }
    else {
        printf("Ten\n");
    }
    printf("Eleven\n");
}
```

Write your answer here:

7. Consider the following code:

```
char input;
scanf("%c", &input);
switch(input) {
    case '1': printf("input is 1\n");
    case '2': printf("input is 1 or 2\n");
              break;
    case '3': printf("input is 3\n");
              break;
    default:  printf("input is not 1,2,3\n");
}
printf("done\n");
```

a. *(3 points)* What does it print if the user enters `1`?

b. *(3 points)* What does it print if the user enters `x`?

8. Write C code to accomplish each of the following things, in the space provided:

a. *(2 points)* Declare three integer variables named `hours`, `minutes` and `seconds`.

b. *(4 points)* Prompt the user to input a number and store the input in the variable `hours`.

c. *(4 points)* Multiply `hours` by `60` and store the result in `minutes`. Then multiply `minutes` by `60` and store the result in `seconds`.

9. *(20 points)* The program below is complete except for a missing function definition. In the space indicated, write the definition of a function named `is_even`, including a suitable function comment. The `is_even` function takes one integer parameter as input. It returns 1 if the given input is even, 0 if the input is odd. For this function, an input of 0 is considered to be even.

   Your answer receives full credit if the function body is one line and does not use an `if` statement. The "body" is the code between the curly braces in the function definition. You receive partial credit for a correct solution with the function body longer than one line or a body that uses an `if` statement.

```
#include <stdio.h>
```

Write your definition of the `is_even` function here, including a function comment:

```
/**
 * Prompt for user input, call is_even, and print the result.
 */
int main() {
    int i;

    printf("Enter an integer: ");
    scanf("%d", &i);

    if (is_even(i))
        printf("The number is even\n");
    else
        printf("The number is odd\n");

    return 0;
}
```

10. *(24 points)* The program below is complete except for the body of the `largest` function. In the space indicated, fill in this function body, per the provided function comment. That is, the `largest` function returns the largest of three integers.

```
#include <stdio.h>

/**
 * Return the largest of the given three numbers.  If two or three values are
 * equal, return the one value that is larger than or equal to the others.
 */
int largest(int x1, int x2, int x3) {
```

Fill in the body of the `largest` function here:

```
}

/**
 * Prompt for user input, call largest, and print the result.
 */
int main() {
    int i,j,k;

    printf("Enter three integers: ");
    scanf("%d%d%d", &i, &j, &k);

    printf("The largest of %d,%d,%d is %d\n", i, j, k, largest(i,j,k));

    return 0;
}
```

11. *(9 points)* For the program in Question 10, fill in the following testing plan with what you consider to be the three most sensible test cases to determine if the program works properly. In the **Inputs** column, provide three integer values for each case, with the values separated spaces. In the **Expected Output** column, provide one integer value.

    Full testing would need more than three cases. Here you're just providing the three case that you think make the most sense, given that you're limited to three.

Case	Inputs	Expected Output
1		
2		
3		