CS16 Exam #1 7/17/2012
75 Minutes 100 Points total

Name: _____

1. (10 pts) Write the definition of a C function that takes two integers `a` and `b` as input parameters. The function returns an integer holding the value $a^2 + b^2$. Write ONLY the function; not an entire program.

```
int function( int a, int b ) {
  return ( a * a ) + ( b * b );
}
```

2. (10 pts) Let `input` and `ret` be integer variables. Consider the following code:

```
ret = 0;
switch ( input ) {
  case 1:
   ret = ret + 1;
   break;
  case 2:
   ret = ret + 2;
   break;
  case 4:
  case 5:
   ret = ret + 4;
   break;
  case 6:
   ret = ret + 6;
  case 7:
   ret = ret + 7;
   break;
  case 8:
   ret = ret + 8;
  default:
   ret = -1;
}
```

Complete the following table asking what the value of `ret` is after executing this code for a given value of `input`. Note that these are independent runs, so `ret` always starts at 0.

If `input` is...	Then `ret` will be...
0	-1
1	1
2	2
3	-1
4	4
5	4
6	13
7	7
8	-1
9	-1

3. (10 pts) Given the integer variables `ret`, `x`, `y`, and `z`, write an if/else if/else construct that will assign the largest of `x`, `y`, and `z` to `ret`. In other words, `ret` = max(max(`x`, `y`), `z`). Do NOT use any built-in max() function or define your own max() function.

```
if ( x >= y && x >= z )
  ret = x;
else if ( y >= x && y >= z )
  ret = y;
else
  ret = z;
```

4. (2 pts) What would most likely happen if the following code were to be run? Why? (Hint: "%s" is the placeholder for strings.)

```
int x = 10;
printf( "%s", x );
```

It'd probably crash. `x` would be treated as an address of a string (i.e. where the string is in memory).

5. (3 pts) What is wrong with the following code?

```
void foo();
int x = 0;
int y;

void foo() {
  if ( x > 5 )
    return;
  x++;
}

y = foo();
```

foo() doesn't return anything (has a void return type), but we attempt to assign its return value to `y`.

6. (20 pts) For each of the following C expressions, clearly mark the type of the expression and the value of the expression. The first two are provided as examples.

Expression	Value	Type			
16	16	<u>int</u>	double	char	char*
0.2 * 0.2	0.04	int	<u>double</u>	char	char*
15 % 4	3	<u>int</u>	double	char	char*
4 % 15	4	<u>int</u>	double	char	char*
2 + 3 * 3	11	<u>int</u>	double	char	char*
2 + 2 / 4.0	2.5	int	<u>double</u>	char	char*
1.0 / 10	0.1	int	<u>double</u>	char	char*
9 % 3	0	<u>int</u>	double	char	char*
"2 + 7.0"	"2 + 7.0"	int	double	char	<u>char*</u>
'2'	'2'	int	double	<u>char</u>	char*
2 / 3	0	<u>int</u>	double	char	char*
5 + (char)2	7	<u>int</u>	double	char	char*

7. (20 pts) There is a partially complete program below that can calculate force (in N) given mass (in kg) and acceleration (m / s^2). It includes the function definition `calcForce`, which performs the actual force calculation to a result in N. The program still needs the following components:
   a. Code that prompts the user for input (using `printf`) asking for mass and acceleration
   b. Code that reads in mass and acceleration from the user (using `scanf`)
   c. Code that calls `calcForce` to get the actual result
   d. Code that prints out the result of `calcForce`
   e. Anything else required so it will compile

Sample Run	Hints
-bash-2.05b$ ./prog Enter mass (kg): 1.5 Enter acceleration (m/s^2): 3.0 Force: 4.5 N -bash-2.05b$	• The printf/scanf placeholder for doubles is %lf • Use function prototypes whenever a function is used before it is defined • Remember to tell C that we are using the standard input / output library • After the program completes, the prompt for bash should be on a new line

```
// program for converting mass and acceleration to force
#include <stdio.h>
double calcForce( double massInKg, double accInMSS );


int main() {
  double mass, acc;
  printf( "Enter mass (kg):  " );
  scanf( "%lf", &mass );
  printf( "Enter acceleration (m/s^2): " );
  scanf( "%lf", &acc );
  printf( "Force: %.1lf N\n", calcForce( mass, acc ) );
  // bonus if you used the .1 part for precision




  return 0;
}
double calcForce( double massInKg, double accInMSS ) {
  return massInKg * accInMSS;
}
```

8. (2 pts) What does the following code print?

```
int x = 0;

if ( x = 0 ) {
  printf( “true” );
} else {
  printf( “false” );
}
```

false (assignment instead of equality used)

9. (2 pts) What are the values of the variables `x` and `y` after the following code is run?

```
int x = 0;
int y = x++ + 1;
```

x is 1, y is 1

10. (3 pts) Considering an unsigned char that is one byte in size:
    a. What is the smallest value?

00000000

    b. What is the largest value?

11111111

    c. What is 128?

10000000

11. (2 pts) Show how the string “foo” is represented in memory in C, byte by byte. You may use the C single quote notation for representing individual characters (i.e. the character “c” is represented in C with ‘c’.)

‘f’, ‘o’, ‘o’, ‘\0’

12.(4 pts) Convert the decimal number `5` to binary (i.e. to base 2).

101

13.(10 pts) For each one of the following error conditions, say what kind of error it is (considering the four types of errors discussed in class):

a. There is a missing semicolon between two printf statements.

Syntax error

b. The program attempts to divide an integer by 0.

Runtime error

c. A function in math.h was used but the programmer forgot to tell C that the math library was being used.

Linker error

d. The formula for a particular calculation was written down incorrectly.

Logic error

e. An attempt is made to increment an unsigned variable that is one byte large and holding the value 255.

Runtime error

14.(2 pts) Give two reasons why a programmer should use symbolic constants. (i.e. constants declared using #define)

-Only need to change them in one place
-Gives semantic meaning to the values