Nested count-driven Loops

1. Consider the following:

   a. Given that `a` was declared as an `int`, how many times does the body of the following loop execute?

```
for(a=2; a>0; a--) {

}
```

2 times

   b. Given that `b` was declared as an `int`, how many times does the body of the following loop execute?

```
for(b=1; b<=3; b++) {

}
```

3 times

   c. Consider the following program and note that it contains loops like those given in (a) and (b) above. What is the output? Use a trace table to help you walk through the code.

```
#include <stdio.h>

int main( void ) {
    int a, b;
    int count = 0;

    for(a=2; a>0; a--) {
        for(b=1; b<=3; b++) {
            count++;
        }
    }
    printf( "%d\n", count );

    return 0;
}
```

a	b	count	a>0	b<=3
–	–	0		
2			T	
	1	1		T
	2	2		T
	3	3		T
	4			F
1			T	
	1	4		T
	2	5		T
	3	6		T
	4			F
0				F

```
Output: 6
```

   d. What would be the output if we changed the inner loop to:

```
for(b=a; b<=3; b++)
```

a	b	count	a>0	b<=3
–	–	0		
2			T	
	2	1		T
	3	2		T
	4			F
1			T	
	1	3		T
	2	4		T
	3	5		T
	4			F
0				F

```
Output: 5
```

2. Trace the following program to determine its behavior. Update the documentation, function name and variable names in `mystery` to reflect its behavior.

```
#include <stdio.h>

void print_rectangle(int height, int width);

int main( void ) {
    print_rectangle(3, 2);

    return 0;
}

/*
 * Purpose: prints a square width by height
 * Parameters: int height, number of rows, >=0
 *             int width, number of columns, >=0
 * Returns: nothing
 */
void print_rectangle(int height, int width) {
    int row, column;

    for (row = 0; row<height; row++) {
        for (column=0; column<width; column++) {
            printf("*");
        }
        printf("\n");
    }
}
```

m	n	s	t	s<m	t<n
3	2	-	-		
		0		T	
			0		T
			1		T
			2		F
		1		T	
			0		T
			1		T
			2		F
		2		T	
			0		T
			1		T
			2		F
		3		F	

output

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

3. Design a function that takes a non-negative integer height and prints a right-angle triangle shape of stars. Challenge: make use of the function you designed in Worksheet 7, Question 7 (print_line shown below) :

```
/*
 * Purpose: print n copies of ch on one line with no newline at the end
 * Parameters: int n, >=0
 *             char ch, character in single quotes (ie. 'a')
 * Returns: nothing
 */
void print_line(int n, char ch) {
    int count;

    for (count = 0; count<n; count++) {
        printf("%c", ch);
    }
}
```

Examples:

if height is 0, it prints nothing

if height is 2, it prints:

```
*
**
```

if height is 3, it prints:

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

```
/*
 * Purpose: print a right angle triangle of * characters with height rows
 * Args: int height, >=0
 * Returns: nothing
 */
void print_triangle(int height) {
    int row;

    for (row = 1; row<=height; row++) {
        int num_stars = row;
        print_line (num_stars, '*');
        printf("\n");
    }
}
```

4. Design a function that takes a non-negative integer `size` and prints pattern of / and \ relative to `size`. Again, make use of `print_line` in your function definition. Don't forget the \ is a special character!

Examples:

if `size` is 0 it prints nothing

if `size` is 1, it prints:

```
/
```

if `size` is 3, it prints:

```
///
//\
/\\
```

if `size` is 4, it prints:

```
////
///\
//\\
/\\\
```

```
/*
 * Purpose: print a pattern shape relative to size
 * Parameters: int size, >=0
 * Returns: nothing
 */
void print_shape(int size) {
    int row;

    for (row = 0; row<size; row++) {
        int num_fwd_slashes = size - row;
        print_line(num_fwd_slashes, '/');
        print_line(row, '\\');
        printf("\n");
    }
}
```

5. Design a function that takes a non-negative integer `size` and prints a number pattern relative to `size`. Examples:

if `size` is 0 it prints nothing

if `size` is 1, prints:

```
1
```

if `size` is 3, prints:

```
123
123
123
```

if `size` is 4, prints:

```
1234
1234
1234
1234
```

```
/*
 * Purpose: print a square pattern of numbers
 * Parameters: int size, >=0
 * Returns: nothing
 */
void print_square_numbers(int size) {
    int row, col;

    for (row=1; row<=size; row++) {
        for (col = 1; col<=size; col++) {
            printf("%d", col);
        }
        printf("\n");
    }
}
```

6. Design a function that takes a non-negative integer `size` and prints a number pattern relative to `size`. Examples:

if `size` is 0 it prints nothing

if `size` is 1, prints:

```
1
```

if `size` is 3, prints:

```
321
 21
  1
```

if `size` is 4, prints:

```
4321
 321
  21
   1
```

```
/*
 * Purpose: print a triangle pattern of numbers
 * Parameters: int size, >=0
 * Returns: nothing
 */
void print_triangle_numbers(int size) {
    int row, col;

    for (row = 0; row<size; row++) {
        for (col = 0; col<row; col++) {
            printf(" ");
        }
        int start_number = size-row;
        for (col = start_number; col>0; col--) {
            printf("%d", col);
        }
        printf("\n");
    }
}
```