CS16 Exam #3 8/30/2012
75 Minutes 100 Points total

Name: _____

1. (3 pts) Consider the following struct definition:

```
struct MyStruct {
  char myChar;
  struct MyStruct* pointer;
};
```

Declare a struct of type `MyStruct` named `foobar`. Initialize `foobar`'s `myChar` field to `'a'` and `foobar`'s `pointer` field to `NULL`. For full credit, you must do all this in one statement.

2. (3 pts) Consider the following complete program:

```
#include <stdio.h>
#define INDEX <<some value>>

int main( int argc, char** argv ) {
  printf( "%s", argv[ INDEX ] );
  return 0;
}
```

Assume that this program is compiled to an executable named `prog`, and then executed with the following command line:

```
./prog first second third
```

For each of the following possible values of `<<some value>>`, what is the corresponding output of `prog`?

If <<some value>> is...	prog will output...
0	
1	
2	

3. (20 pts) Write the definition for a recursive function named `findSum` that will add together all the integers between the integers `n` and `m`, inclusive. If `n > m`, then `findSum` should return `0`. `findSum` should always return an integer. To get a better idea of what `findSum` should do, look at the following table illustrating its behavior:

Value of `n`	Value of `m`	Summation Sequence	Result
4	5	4 + 5	9
1	3	1 + 2 + 3	6
4	4	4	4
5	4	<<none>>	0

4. (2 pts) Why won't the following code compile?

```
int x = 10;
void* p = &x;
*p = 11;
```

5. (6 pts) Consider the following struct definition:

```
struct SomeStruct {
  int n;
  double d;
};
```

Write the definition for a function named `addStruct` that will take:

- A pointer to a `SomeStruct`
- An `int` named `nAdd`
- A `double` named `dAdd`

The function should add `nAdd` and `dAdd` to the provided `SomeStruct`'s `n` and `d` fields, respectively. The function should not return anything.

6. (2 pts) If a variable is declared from within a block in a non-recursive function, where does the memory get allocated from?

Circle one: heap stack

7. (2 pts) When one of malloc, calloc, or realloc gets called, where does the memory get allocated from?

Circle one: heap stack

8. (2 pts) If a variable is declared from within a block in a recursive function, where does the memory get allocated from?

Circle one: heap stack

9. (7 pts) Consider the following code:

```
#include <stdio.h>
#define LENGTH ... /* some positive integer */

int filterLess( int* src, int* dest, int less ) {
  int srcPos;
  int destPos = 0;
  for ( srcPos = 0; srcPos < LENGTH; srcPos++ ) {
    if ( src[ srcPos ] < less ) {
      dest[ destPos ] = src[ srcPos ];
      destPos++;
    }
  }
  return destPos;
}

int main() {
  int src[ LENGTH ] = { ... }; // some arbitrary integers
  int dest[ LENGTH ];
  int x, num;
  // YOUR CODE HERE -- SEE BELOW
  for ( x = 0; x < num; x++ ) {
    printf( "%i\n", dest[ x ] );
  }
  return 0;
}
```

The purpose of this code is to print out all integers in `src` that are less than `10`. In order to do this, you will need to call the `filterLess` function with the appropriate parameters. This call will be inserted where the `YOUR CODE HERE` line is above. Only one statement is needed.

10. (2 pts) In your own words, explain why a programmer would ever want to use a `void` pointer.

11. Consider the following recursive function definition:

```
int whoKnows( int* array, int a ) {
  if ( a == 0 ) {
    // CASE 1
    return 0;
  } else if ( array[ 0 ] < 0 ) {
    // CASE 2
    return array[ 0 ] + whoKnows( array + 1, a - 1 );
  } else {
    // CASE 3
    return whoKnows( array + 1, a - 1 );
  }
}
```

a. (3 pts) For each of the above cases, circle whether it is a base case or a recursive case:

CASE 1: base case recursive case

CASE 2: base case recursive case

CASE 3: base case recursive case

b. (2 pts) In your own words, what is the purpose of `whoKnow`'s parameter `a`? In other words, what does the parameter specify?

c. (5 pts) The name `whoKnows` is a fairly bad name for any function. From the following list of candidate names, select a better name that most closely matches what this function does: `average`, `countEven`, `countMax`, `countMin`, `countNeg`, `countOdd`, `countPos`, `countSevens`, `indexFirstEven`, `indexFirstOdd`, `indexOfMax`, `indexOfMin`, `isSorted`, `maxValue`, `minValue`, `noDuplicates`, `sum`, `sumEven`, `sumNeg`, `sumOdd`, `sumPos`.

12. (2 pts) Complete this sentence: "The values held by pointers are _____".

13.(9 pts) Consider the following code:

```
struct Point {
  int x;
  int y; };
struct Circle {
  struct Point center;
  int radius; };
struct ConcentricCircle {
  struct Circle* circles;
  int numCircles; };

struct Point point;
struct Point* pointp;
struct Circle circle;
struct Circle* circlep;
struct ConcentricCircle concCircle;
struct ConcentricCircle* concCirclep;

// OMITTED INITIALIZATION CODE
```

With respect to the above code, determine the type of each of the following expressions, writing ERROR if there is a type error. The first one has been done for you.

Expression	Result Type of Expression
point	struct Point
&point	
*(&point)	
circle.center	
circle.radius	
concCircle->numCircles	
concCirclep->numCircles	
concCircle.circles	
concCircle.circles[0]	
(*circlep).radius	

14. (5 pts) Write an expression that will return a pointer to a block of memory that can hold `numInts` integers, where `numInts` is an `int` variable that has been initialized to some positive integer. The initial value of each integer in the block should be zero. For full credit, you can only use a single expression.

15. (2 pts) What is wrong with the following code? Name the specific kind of error.

```
void error1( int n ) {
  int* p = malloc( sizeof( int ) );
  if ( n % 2 == 0 ) {
    free( p );
  }
}
```

16. (2 pts) What is wrong with the following code? Name the specific kind of error.

```
int* error2() {
  int* p = malloc( sizeof( int ) );
  free( p );
  return p;
}
```

17.(3 pts) Consider the following code:

```
void swap( int x, int y ) {
  int temp = x;
  x = y;
  y = temp;
}

int main() {
  int x = 2, y = 7;
  swap( x, y );
  printf( "x: %i\n", x );
  printf( "y: %i\n", y );
  return 0;
}
```

After calling the `swap` function, the programmer finds that the values of `x` and `y` in `main` have not changed. Why? Suggest a fix. You do not have to implement the fix.

18.(10 pts) Write the definition of a function named `copyString` that will take a pointer to a string and return a pointer to a newly allocated string that is a copy of the original one. (Hint: you will need `strlen`, `strcpy`, and `malloc`/`calloc`, and you may assume that all relevant libraries have already been included.)

19.(8 pts) What does the following code print, starting execution at `main`?

```
int x = 0;
void one( int x ) {
  printf( "one: %i\n", x );
}

void two( int notX ) {
  x = 2;
  printf( "two: %i\n", x );
}

void three( int notX ) {
  x = 3;
  printf( "three: %i\n", x );
}

void four( int x ) {
  if ( 1 ) {
    int x = 4;
  }
  printf( "four: %i\n", x );
}

void five( int x ) {
  if ( 1 ) {
    x = 5;
  }
  printf( "five: %i\n", x );
}

int main() {
  one( x );
  if ( 1 ) {
    int x = 8;
    two( x );
    printf( "main1: %i\n", x );
    three( x );
    printf( "main2: %i\n", x );
  }
  four( x );
  five( x );
  printf( "main3: %i\n", x );
  return 0;
}
```