Computing Machinery I
Assignment 4

Structures and Subroutines

Create an ARMv8 assembly language program that implements the following program:

```
#define FALSE  0
#define TRUE   1

struct point {
  int x, y;
};

struct dimension {
  int width, length;
};

struct cuboid {
  struct point origin;
  struct dimension base;
  int height;
  int volume;
};


struct cuboid newCuboid()
{
  struct cuboid c;

  c.origin.x = 0;
  c.origin.y = 0;
  c.base.width = 2;
  c.base.length = 2;
  c.height = 3;
  c.volume = c.base.width * c.base.length * c.height;

  return c;
}

void move(struct cuboid *c, int deltaX, int deltaY)
{
  c->origin.x += deltaX;
  c->origin.y += deltaY;
}

void scale(struct cuboid *c, int factor)
{
  c->base.width *= factor;
  c->base.length *= factor;
  c->height *= factor;
  c->volume = c->base.width * c->base.length * c->height;
}

void printCuboid(char *name, struct cuboid *c)
{
  printf("Cuboid %s origin = (%d, %d)\n", name, c->origin.x, c->origin.y);
  printf("\tBase width = %d  Base length = %d\n", c->base.width,
         c->base.length);
  printf("\tHeight = %d\n", c->height);
  printf("\tVolume = %d\n\n", c->volume);
```

```
}
int equalSize(struct cuboid *c1, struct cuboid *c2)
{
  int result = FALSE;

  if (c1->base.width == c2->base.width) {
    if (c1->base.length == c2->base.length) {
      if (c1->height == c2->height) {
        result = TRUE;
      }
    }
  }

  return result;
}

int main()
{
  struct cuboid first, second;

  first = newCuboid();
  second = newCuboid();

  printf("Initial cuboid values:\n");
  printCuboid("first", &first);
  printCuboid("second", &second);

  if (equalSize(&first, &second)) {
    move(&first, +3, -6);
    scale(&second, 4);
  }

  printf("\nChanged cuboid values:\n");
  printCuboid("first", &first);
  printCuboid("second", &second);
}
```

Implement all the subroutines above as unoptimized closed subroutines, using stack variables to store all local variables. Note that the function newCuboid() must have a local variable (called *c*) which is returned by value to main(), where it is assigned to the local variables *first* and *second*. In other words, create code similar to what the C compiler produces, even if it seems inefficient.

Also run the program in *gdb*, displaying the values of *first* and *second* after they have been set by function calls. You should show that the functions are working as expected. Capture the *gdb* session using the *script* UNIX command, and name the output file *script.txt*. Name your program *assign4.asm*.

Other Requirements

Make sure your code is readable and fully documented, including identifying information at the top of each file. You must comment each line of assembly code. Your code should also be well designed: make sure it is well organized, clear, and concise.

New Skills Needed for this Assignment:

- Implementation of structs and nested structs
- Implementation of subroutines in assembly
- Returning structs by value from functions
- Use of pointers as arguments to subroutines

Submit the following:

1. Your assembly source code file for the program, and your script output file. Use the *Assignment 4* Dropbox Folder in D2L to submit electronically. The TA will assemble and run your program to test it.

Be sure to name your program and script file as described above.

Computing Machinery I
Assignment 4 Grading

Student:_____

Functionality			
newCuboid() function	8	_____	
move() function	4	_____	
scale() function	6	_____	
equalSize() function	6	_____	
printCuboid() function	6	_____	
main() function	8	_____	
Correct implementation of structs	4	_____	
Correct use of stack variables	4	_____	
Script showing *gdb* session	2	_____	
Complete documentation and commenting	4	_____	
Formatting (use of columns and white space)	4	_____	
Design quality	2	_____	
Total	**58**	_____	_____**%**