

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, height;
};

struct box {
    struct point origin;
    struct dimension size;
    int area;
};

struct box newBox()
{
    struct box b;

    b.origin.x = 0;
    b.origin.y = 0;
    b.size.width = 1;
    b.size.height = 1;
    b.area = b.size.width * b.size.height;

    return b;
}

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

void expand(struct box *b, int factor)
{
    b->size.width *= factor;
    b->size.height *= factor;
    b->area = b->size.width * b->size.height;
}

void printBox(char *name, struct box *b)
{
    printf("Box %s origin = (%d, %d) width = %d height = %d area = %d\n",
        name, b->origin.x, b->origin.y, b->size.width, b->size.height,
        b->area);
}

int equal(struct box *b1, struct box *b2)
{

```

```

    int result = FALSE;

    if (b1->origin.x == b2->origin.x) {
        if (b1->origin.y == b2->origin.y) {
            if (b1->size.width == b2->size.width) {
                if (b1->size.height == b2->size.height) {
                    result = TRUE;
                }
            }
        }
    }

    return result;
}

int main()
{
    struct box first, second;

    first = newBox();
    second = newBox();

    printf("Initial box values:\n");
    printBox("first", &first);
    printBox("second", &second);

    if (equal(&first, &second)) {
        move(&first, -5, 7);
        expand(&second, 3);
    }

    printf("\nChanged box values:\n");
    printBox("first", &first);
    printBox("second", &second);
}

```

Implement all the subroutines above as unoptimized closed subroutines, using stack variables to store all local variables. Note that the function `newBox()` must have a local variable (called *b*) 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. Name the program *assign4.asm*.

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*.

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

newBox() function	8	_____	
move() function	4	_____	
expand() function	6	_____	
equal() function	6	_____	
printBox() function	6	_____	
main() function	8	_____	
Correct implementation of structs	4	_____	
Correct use of stack variables	4	_____	
Script showing <i>gdb</i> session	2	_____	
Complete documentation and commenting	4	_____	
Design quality	2	_____	
Total	54	_____	_____%