1) RISC-V Assembly Programs and Translation

C version:

```c
int f(int *a, int b, int length) {
    int res = 0;
    for (int i = 0; i <= (length); i++) {
        int c = a[i] + b;
        if (c & 1 == 0) {
            res += 2;
        } else {
            res += c - 1;
        }
    }
    return res;
}
```

Convert this program into RISC-V assembly
2 TOO LOW THEY BVILD WHO BVILD BENEATH THE STARS

Calling convention

The function `drawFrame` draws one frame from a 17 second animation of the solar system onto a 512×512 pixel display. It takes two arguments; the first is the frame number and the second is brightness. The source for this function (and the additional functions it calls) is on the next page, but for reference, a version in C is given below.

```c
void drawFrame(int time, int intensity) {
    int ex = EARTHPOS[2*time];
    int ey = EARTHPOS[2*time+1];
    int mx = MOONPOS[2*time];
    int my = MOONPOS[2*time+1];
    point(ex, ey, intensity);
    point(mx, my, intensity);
}

void drawTerraLuna(int time, int intensity) {
    int ex = EARTHPOS[2*time];
    int ey = EARTHPOS[2*time+1];
    point(ex, ey, intensity);
}

void point(int x, int y, int intensity) {
    VRAM[y<<9 + x] = intensity;
}
```

**Question a** Draw the function call tree of `drawFrame(62, 999)`.

**Question b** Execution is halted at the label `BREAK`, and a dump of the stack is given in listing 2. For each word shown, indicate the source register of the `lw` instruction that placed the word on the stack, or, if the source register cannot be determined, write “unknown”.

**Question c** Rewrite `drawTerraLuna` to use only caller-saved registers (excepting sp).
drawFrame:  
andi a0, a0, 0x1FF  
addi sp, sp, -8  
sw ra, a(sp)  
sw a1, 0(sp)  
call drawTerraLuna  
li a0, 0  
li a1, 0  
lw a2, 0(sp)  
call point  
lw ra, a(sp)  
addi sp, sp, +8  
ret

drawTerraLuna:  
addi sp, sp, -12  
sw ra, 8(sp)  
sw s0, 4(sp)  
sw s1, 0(sp)  
slli s0, a0, 3  
# s0 ← 2×time (address offset)  
mv s1, a1  
# s1 ← intensity  
li t0, 0x4000  
# start of EARTHPOS data  
add t0, t0, s0  
lw a0, 0(t0)  
lw a1, 4(t0)  
mv a2, s1  
call point  

BREAK:  
li t0, 0x5000  
# start of MOONPOS data  
add t0, t0, s0  
lw a0, 0(t0)  
lw a1, 4(t0)  
mv a2, s1  
call point  

lw s1, 0(sp)  
lw s0, 4(sp)  
lw ra, 0(sp)  
addi sp, sp, +12  
ret

point:  
li t0, 0x8000  
# start of VRAM (video memory)  
slli a0, a0, 2  
add t0, t0, a0  
slli a1, a1, 11  
add t0, t0, a1  
sw a2, 0(t0)  
ret
3 Premonitions
Boolean simplification

Simplify the following expressions; specifically, into minimal sums of products.

\[ AB + (c + A\bar{b}) \]

\[ (\bar{A}B\bar{C})(A(\bar{A}+B) + C(A+B)) \]
4) Composing Bluespec

Convert the following into Bluespec:

Self-Masking Circuit:

This circuit takes a 4-bit input A and performs a masking operation on it by xor-ing it with itself shifted left once, right twice, and left 3 times.

Parameterized Functions

Convert the 4-bit Self-Masking Circuit into an n-bit version that xors A with 1 through n-1 shifts of A in alternating directions
5) Circuit Synthesis

Create circuits for the following boolean expressions

1) \((AB) + (C + (A\overline{B}))\)

2) \((\overline{ABC}) (A + B) (\overline{A} + C)\)
3) \((AB\overline{C}D) \land (\overline{A}B\overline{C}D)\)

Give a boolean expression for each output bit of this circuit.