Top-down vs. Bottom-up Register Allocation

Source code

program main;
    var a, b, c, d, e, f, g, h: integer;
begin
    a := 1;
    b := 2;
    c := b - 4;
    d := a + b;
    e := d + 1;
    f := e - c * e;
    g := (d + e) + f;
    h := g + a;
    writeln(h)
end.

1. Show the ILOC code for the above source program assuming a register-register model. Code shape: Once a value of a variable is loaded into a virtual register, it stays in this register throughout its lifetime. Use the memory model as discussed in class, i.e., use the dedicated register r0 to hold the base address 1024. Addresses above 1024 are used for spilling registers.

2. Give the life ranges for all virtual registers (ignore r0). What is MAX_LIVE for your ILOC code?

3. Assume that there are two registers in the feasible set, called f1 and f2, and there is the dedicated register r0 that we do not consider for register allocation, i.e., is not part of the available register set.

   (a) Show the ILOC code that would be generated by the top-down algorithm discussed in EAC, i.e., no MAX_LIVE consideration, for (MAX_LIVE - 1) available registers (k-F registers are available).
(b) Show the ILOC code that would be generated by the top-down algorithm discussed in class, i.e., with MAX_LIVE consideration, for (MAX_LIVE - 1) available registers (k-F registers are available).

(c) Show the ILOC code that would be generated by the bottom-up algorithm discussed in class for (MAX_LIVE - 1) available registers (k-F registers are available).