CSE 401 - Section 10 - Dataflow and Single Static Assignment

1. **Dataflow Review** For each of the following optimizations, list the dataflow analysis that would be most directly applicable. You may use a single dataflow analysis for multiple optimizations, or none. The possible dataflow analyses are reproduced here for reference:
   Live Variable Analysis (Determining if there is any path from the definition of a variable to its use along which it is not redefined)
   Reaching Expressions (For an expression, determining which other basic blocks are reached without redefining any of the variables in that expression)
   Reaching Definitions (Determining which other basic blocks could potentially see the value of a given definition)
   Very Busy Expressions (Determining if an expression is evaluated and used along every path that leaves a basic block, and if the value would be consistent in the parent basic block)

a) Constant Propagation - If a variable x is defined to be a constant in one part of the code, replace uses of the variable x with its defined constant.

b) Copy Propagation - If a variable x is defined to be equal to a variable y in one part of the code, replace uses of the variable x with the variable y.

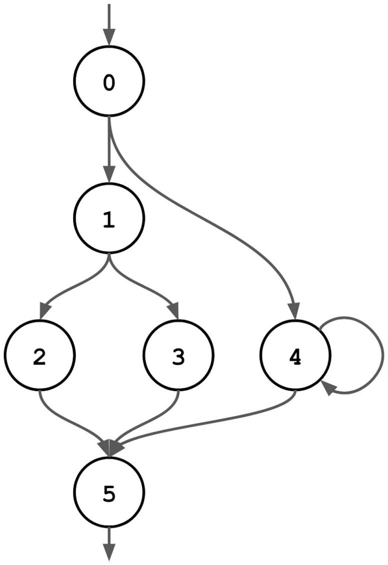
c) Common Subexpression Elimination - If an expression is computed twice and will have the same value in both locations, compute it only once (Note: only applies to expressions without side effects).

d) Code Hoisting - Reducing the size of the code by factoring out duplicate code that appears in all possible paths in a part of the program.

e) Dead Store Elimination - Removing assignments to a variable if that assignment will never be used in the program.

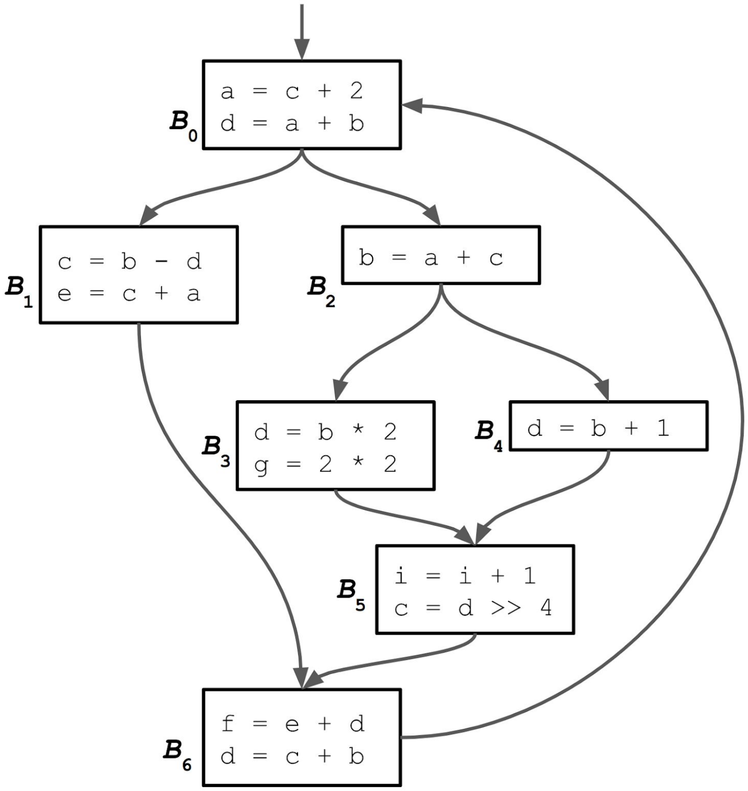
2. **Dominators and Dominance Frontiers** Consider the following simplified control flow graph. For each node in the graph, fill in the table with the set of nodes that are strictly dominated by that node and the set of nodes in its dominance frontier.

   For reference, a node **X** is considered to dominate **Y** iff every path from the entry point of the control flow graph to **Y** includes **X**. A node **X** strictly dominates **Y** if **X** dominates **Y** and **X** ≠ **Y**. Finally, a node **Y** is in the dominance frontier of node **X** if **X** dominates an immediate predecessor of **Y** but **X** does not strictly dominate **Y**.



NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0		
1		
2		
3		
4		
5		

3. **Single Static Assignment** The following is a control flow graph depicting the execution of some arbitrary code. Convert this code to Single Static Assignment form. Hint: you may find it helpful to compute the dominance frontiers of each basic block.


B_0
a = c + 2
d = a + b
B_1
c = b - d
e = c + a
B_2
b = a + c
B_3
d = b * 2
g = 2 * 2
B_4
d = b + 1
B_5
i = i + 1
c = d >> 4
B_6
f = e + d
d = c + b