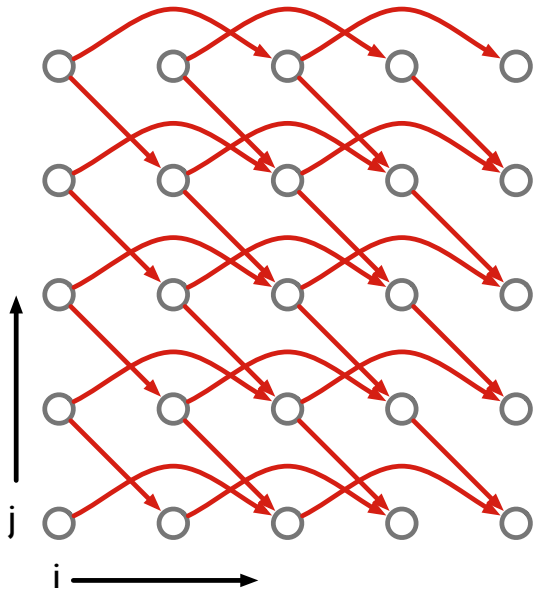
ECE 468
Problem Set 6: Dependence analysis and Dataflow analysis

1. Draw the iteration space graph for the following piece of code (be careful about the index expressions!):

```
for (i = 0; i < 5; i++)
  for (j = 0; j < 5; j++)
    A[j+1][i] = A[j][i+1] + A[j+1][i+2];
```

Answer: Note that the index expression use i and j in the opposite locations than you would expect (as if the array were being accessed in column major order instead of row major order). That does *not* change the rules for dependences. So when i is 2 and j is 2, the body of the loop writes to `A[3][2]`. That location is *read* when i is 1 and j is 3, which is an *earlier* iteration (so this is an anti-dependence). That location is also read when i is 0 and j is 2, which is also earlier, and is an anti-dependence. The iteration space graph for this loop is below. Note that the dependence arrows are colored red to indicate that these are *anti* dependences (the reads happen before the writes):



2. What are the distance vectors? The direction vectors?

Answer: There are two distance vectors, both of which are anti-dependences: $(2, 0)$ and $(1, -1)$. The corresponding direction vectors are: $(+, 0)$ and $(+, -)$

3. Can the loops be interchanged? Why or why not?

   Answer: The loops cannot be interchanged: the $(+, -)$ direction vector would be changed to a $(-, +)$, which is not allowed.

4. Use the GCD test to argue that the following loop does not have any loop-carried dependences:

```
for (i = 0; i < N; i++) {
  A[3i + 2] = A[6i + 3];
}
```

   Answer: The GCD test asks whether there are two iterations i and i' such that:

$$3i + 2 = 6i' + 3$$

   This is equivalent to asking whether the following equation has integer solutions:

$$3i - 6i' = 1$$

   The GCD test says that this equation cannot have a solution, as the GCD of the coefficients of i and i' is 3, which does not divide 1.

5. Give an example of a piece of code where the GCD test shows that the loop has a dependence, but where the loop itself does not have a dependence (hint: remember that the GCD test does not take into account loop bounds; try to come up with a loop where the dependence can only exist outside the bounds of the loop)

   Answer:

   There are many possible solutions, but here is one:

```
for (i = 0; i < 10; i++) {
  A[i + 20] = A[i]
}
```

   The GCD test will say that there is a possible dependence (as the GCD of the coefficients would be 1, which divides 20). However, that dependence only exists if

the writing iteration is 20 iterations after the reading iteration (think about what location is written to when $i = 0$, and what i would have to be to read from the same location). But the loop only runs for 10 iterations: the values of i that would read from the locations written to in the loop never happen.

6. Show the results of running a *reaching definitions* analysis on the following piece of code: for each line of code, show which definitions reach that line of code by indicating the line number the definition occurred in.

```
    1: x = 4;
    2: y = 7;
L1  3: if (x > c) goto L4
    4:   if (y > 3) goto L2
    5:     a = x + 1;
    6:     b = a + x;
    7:     goto L3
L2  8:     a = a + x;
    9:     b = x + 1;
L3 10:   y = a + b;
   11:   goto L1;
L4 12: halt
```

Answer:

Answer: To begin, let us determine the IN and OUT sets for each statement (this is like building the statement-level CFG):

Instruction	Predecessors	Successors
1	None	2
2	1	3
3	2, 11	4, 12
4	3	5, 8
5	4	6
6	5	7
7	6	10
8	4	9
9	8	10
10	7, 9	11
11	10	3
12	3	None

Next, for each statement, we calculate the GEN and KILL sets. In the case of reaching definitions, the GEN set for a statement is the set of definitions (line number & variable) created in that statement. The KILL set is all other definitions of that variable.

Line #	GEN	KILL
1	[x, 1]	
2	[y, 2]	[y, 10]
3		
4		
5	[a, 5]	[a, 8]
6	[b, 6]	[b, 9]
7		
8	[a, 8]	[a, 5]
9	[b, 9]	[b, 6]
10	[y, 10]	[y, 2]
11		
12		

We can then compute the IN and OUT sets for each statement. Recall that the formula for each statement's IN and OUT sets in reaching definitions is:

$$\begin{aligned} IN(s) &= \bigcup_{t \in pred(s)} OUT(t) \\ OUT(s) &= (IN(s) - KILL(s)) \cup GEN(s) \end{aligned}$$

Every IN and OUT set is initialized to {}. We begin by computing OUT(1):

$$OUT(1) = [x, 1]$$

which affects its successors, so we compute IN(2) and OUT(2)

$$\begin{aligned} IN(2) &= [x,1] \\ OUT(2) &= [x,1],[y,2] \end{aligned}$$

which propagates to 3. Note that 3 has multiple predecessors, but OUT(11) is currently empty:

$$IN(3) = OUT(2) \cup OUT(11) = [x,1],[y,2]$$

$$OUT(3) = [x,1],[y,2]$$

This propagates to both 4 and 12. Let's compute 4 first:

$$\begin{aligned} IN(4) &= [x,1],[y,2] \\ OUT(4) &= [x,1],[y,2] \end{aligned}$$

This propagates to 5 and 8. Let's compute 5:

$$\begin{aligned} IN(5) &= [x,1],[y,2] \\ OUT(5) &= [x,1],[y,2],[a,5] \end{aligned}$$

Which affects 6:

$$\begin{aligned} IN(6) &= [x,1],[y,2],[a,5] \\ OUT(6) &= [x,1],[y,2],[a,5],[b,6] \end{aligned}$$

And 7:

$$\begin{aligned} IN(7) &= [x,1],[y,2],[a,5],[b,6] \\ OUT(7) &= [x,1],[y,2],[a,5],[b,6] \end{aligned}$$

7 propagates to 10, which we can compute (recall that OUT(9) is currently empty):

$$IN(10) = OUT(7) \cup OUT(9) = [x,1],[y,2],[a,5],[b,6]$$

$$OUT(10) = [x,1],[y,10],[a,5],[b,6]$$

Which propagates to 11, then back around to 3:

$$IN(3) = OUT(2) \cup OUT(11) = [x,1],[y,2],[y,10],[a,5],[b,6]$$

$$OUT(3) = [x,1],[y,2],[y,10],[a,5],[b,6]$$

OUT(3) has changed, so we propagate this information to 4, and so on.

When the whole process finishes, we are left with the following:

Line #	**IN**	**OUT**
1		[x, 1]
2	[x, 1]	[x, 1], [y, 2]
3	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]
4	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]
5	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 2], [y, 10], [a, 5], [b, 9], [b, 6]
6	[x, 1], [y, 2], [y, 10], [a, 5], [b, 9], [b, 6]	[x, 1], [y, 2], [y, 10], [a, 5], [b, 6]
7	[x, 1], [x, 10], [y, 6], [a, 5]	[x, 1], [x, 10], [y, 6], [a, 5]
8	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 2], [y, 10], [a, 8], [b, 9], [b, 6]
9	[x, 1], [y, 2], [y, 10], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 2], [y, 10], [a, 8], [b, 9]
10	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]
11	[x, 1], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]
12	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]	[x, 1], [y, 2], [y, 10], [a, 5], [a, 8], [b, 9], [b, 6]