

Reaching Definitions - Example

```
def m(): Int = {  
  /*pc 1*/  var x = 5;  
  /*pc 2*/  val y = 1;  
  /*pc 3*/  while (x > 1) {  
    /*pc 4*/    y = x * y  
    /*pc 5*/    x = x - 1  
  }  
  /*pc 6*/  x + y  
}
```

Handwritten annotations illustrating reaching definitions:

- For `var x = 5;`: $(x, 1)$
- For `val y = 1;`: $(y, 2)$
- For `while (x > 1) {`: $(x, 1)$ (pointing down to the loop body)
- For `y = x * y`: $(y, 4)$
- For `x = x - 1`: $(x, 5)$ (pointing down to the loop body), $(x, 1)$ (pointing up to the loop header), and $(x, 5)$ (pointing to the assignment).

Very Busy Expressions Analysis - Example

```
def m(a: Int, b: Int): Int = {  
  /*pc 0*/ var x, y = 0  
  /*pc 1*/ if(a > b) {  
    /*pc 2*/ x = b - a  
    /*pc 3*/ y = a - b  
    else {  
      /*pc 4*/ y = b - a  
      /*pc 5*/ x = a - b  
    }  
  }  
  /*pc 6*/ n(x, y)  
}
```

Handwritten annotations:

- Handwritten $f^1 = a - b$ and $f^2 = b - a$ with arrows pointing to the expressions in the if-else branches.
- Handwritten $x = f^2$ and $y = f^1$ in the right column, indicating the state of variables after the if-else block.
- Handwritten $x = f^1$ and $y = f^2$ in the right column, indicating the state of variables after the if-else block.
- Handwritten $-- a \neq b$ below the closing brace of the if-else block.



Very Busy Expressions Analysis - applied

```
def m(b1 : Boolean, b2 : Boolean, a: Int, b : Int ): Int = {  
  if(b1){  
    m()  
    a/b  
  } else {  
    if(b2) {  
      val c = a + b  
      a/b * c  
    } else {  
      val c = a * b  
      a/b - c  
    }  
  }  
}
```

Live Variable Analysis - example

```

def m(): Int = {
  /*pc 1*/  var x = 2
  /*pc 2*/  var y = 4
  /*pc 3*/  x = 1
  /*pc 4*/  val z = if( y > x ) {
  /*pc 5*/    val z = y
                } else {
  /*pc 6*/    val z = y * y
                }
  /*pc 7*/  x = z + 1
}
    
```

kill	gen	
$\{ \}$	\emptyset	$L_{entry} \neq L_{exit} \{x\}$
$\{ \}$	\emptyset	
$\{ \}$	\emptyset	
\emptyset	$\{x, y\}$	
$\{z\}$	$\{y\}$	
$\{z\}$	$\{y\}$	
$\{x\}$	$\{z\}$	

