Typical line of code:

```r
> results.RNA[results.RNA$batch > 1, 2:3] <- log2(matrix(1:10, nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]
```

1) Look for `<-` to divide up the code; if no `<-`, then not assigning the results to anything¹

```
__put_results_here__ <- __compute_this___
```

2) Look for left parentheses (`(`
   a) the words just to the left are functions
   ```r
   > results.RNA[results.RNA$batch > 1, 2:3] <- log2(matrix(1:10, nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]
   ```
   ```r
   log2(matrix(1:10, nrow = 5, ncol = 2))
   matrix(1:10, nrow = 5, ncol = 2)
   ```
   b) everything to the right until the closing parenthesis are directions (i.e., arguments) for the function

3) Look for subsetting - `$`, `[` or `[[`
   ```r
   $ - pulls out a named column from a data frame or a named item from a list
   ```
   ```r
   [ - starts subsetting; if [] either a vector or list, if [,] either a matrix or data frame
   ```
   ```r
   results.RNA[results.RNA$batch > 1, 2:3]
   ```
   ```r
test.data$value[i.type == "cDNA"]
```

4) Math symbols `+`, `-`, `*`, `/`, `^` have normal functions:

```
log2(matrix(1:10,nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]
```
5) Other symbols ==, !=, <, > ask for comparisons and yield logical (T/F) results:

```
results.RNA$batch > 1
i.type == "cDNA"
```

6) Words in quotes are character data only, not objects or functions

"cDNA"

5) Any remaining words not followed by a single = refer to objects in the workspace

```
> results.RNA[results.RNA$batch > 1 , 2:3 ] <- log2(matrix(1:10,nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]
```

5) Words inside () and followed by a single = can either:

a) name arguments to a function: `matrix( 1:10, nrow = 5, ncol = 2)`

b) create a named column or list item: `Tally <- data.frame(number = 1:3, color = c("black", "red", "green"))`

6) Finally, can break up functions within functions into different lines of code for ease of understanding!

```
> u <- matrix(1:10,nrow = 5, ncol = 2)
> v <- log2( u )
> w <- test.data$value
> x <- i.type == "cDNA"
> y <- w[ x ]
> z <- results.RNA$batch > 1
> results.RNA[ z , 2:3 ] <- v * y
```

1 A single equal sign = can be used as the assignment operator instead of <-. However, not all instances of single equal signs are assignment operators; if the = is within parentheses, it is NOT equivalent to <- ! For this reason, most coders will only use <- as the assignment operator, but = works for backwards-compatibility and you may see some codes with it.