Lab Class 6 had three problems:
- Reversing Input
- Computing Averages
- Frequency Counting

We shall look at each one in turn
Read five words (strings) from the user.
Print them back in reverse order.
Program should be easy to convert it to read 10, 15 or 100 (whatever!) words.
set var “count” to 5

read in words (0...4)

print out words (4...0 — in reverse)

We need an array (called words) to carry information between the 2nd and 3rd steps.
var words = new Array(count);

for( i=0 ; i < count; i++ )
    words[i] = prompt("Enter word number "+i);

for( i = count-1; i >= 0; i-- )
    document.write("<P>Word "+i+" : "+words[i])
Lab 6 — Problem 2

- Store 100 numbers (each between 1 and 100).
- You may compute the numbers randomly.
- Print out the average of all the numbers.
initialise “magic nos.” (count, size)

initialise “count” numbers at random

compute average

print average
var numbers = new Array (count);

for( i=0; i < count; i++) {
    numbers[i] = random(1,size)
}
Lab 6 — Problem 2 Sum-Loop Flowchart

```
total = 0; i = 0
```

```
if (i < count) then
    total = total + numbers[i]
    i = i + 1
else
    // Do nothing
```

var total = 0;

for( i=0; i < count; i++ ) {
    total = total + numbers[i];
}

var average = total / count ;
Write a small function, called `random`.

It takes two numbers as parameters.

It returns a random number between those two bounds (inclusively).
We clearly define the “interface” (inputs and outputs) of the function, as well as how they are related:

function random

- **Inputs**: 2 numbers, lower and upper bound
- **Action**: generate random number in that range
- **Output**: the random number
We can immediately produce the function “frame”:

```javascript
function random(low,high){
    var rand = (Math.random()*(high-low))+low;
    return rand
}
```

We can then fill in the details.
Generate 1000 numbers, each between 0 and 99.
Keep a count of how often each number is generated.
At the end of the run print out the counts.

For example, you might get this output (it’s unlikely, but you might):
0 was generated 100 times
5 was generated 700 times
62 was generated 200 times
This is a tally problem — count the number of times each item appears.

3 1 3 0 2 3 4 3 3 2 1 2 2 1 0 4
0 11 (2)
1 111 (3)
2 1111 (4)
3 11111 (5)
4 11 (2)
setup 1000 nos; zero counts

i = 0

\[ i < \text{count} \]

\[ \text{num} = \text{numbers}[i] \]

\[ \text{counts}[\text{num}] = \text{counts}[\text{num}] + 1 \]

\[ i = i + 1 \]
The tallying code:

```javascript
for( i=0; i<count; i++) {
    var num = numbers[i];
    counts[num] = counts[num]+1;
}
```
Updating the tally can be done a number of ways:

```cpp
counts[num] = counts[num] + 1;
```
or
```
counts[num] += 1;
```
or even
```
counts[num]++;
```