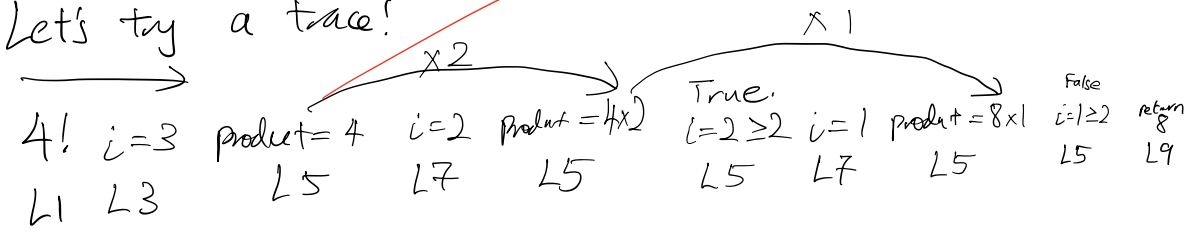
11.1

```
1  long factorial(long n)
2  {
3      long i = n - 1;
4      long product;
5      for (product = n; i >= 2; product *= i)
6      {
7          i -= 1;
8      }
9      return product;
10 }
```

change to n to more it ahead!

missing 3!

Let's try a trace!



11.2

The algo counts how many times /k can be done!

(8,2) → 3
(81,3) → 4
(100,5) → 2

$\lfloor \log_k(n) \rfloor$

(c) n=0 → -1 (But not undefined)

(d) k=1

Something doesn't change ....

something >= 1 ....

No Stop.

12.1a

T_0 : $m = \max[l_0]$

T_i : $m = \max[l_0, \cdots, l_i]$ ← Assertion.

Assume T_{i-1} is true:

Case $l_i > m$: $m = l_i$ $\Rightarrow m = \max[l_0, \cdots l_i]$

Case $l_i \leq m$: $m > l_i$

So T_i is true.

By induction, T is true.

12.1b

$m = 0$ — EG.

$L = [3, 4, 5]$

$m \notin L$ $\quad m \neq \max(L)$