**CASE 1**

4

a b aa b

13 21 37 46

4

0 3 aaabb

1 3 aaabb

2 3 aaabb

3 3 aaabb

* Visualization of the Automaton and the Fenwick Tree

A blackboard with arrows and numbers

Description automatically generated

* Subtrees, failure links, final states and start states:

0 | fl: 0 | sub: 4 | ft: 4 | st: 0

1 | fl: 0 | sub: 2 | ft: 3 | st: 1

2 | fl: 0 | sub: 1 | ft: 4 | st: 3

3 | fl: 1 | sub: 1 | ft: 3 | st: 2

* This is like the Fenwick tree is built: notice how the ranges depends on st and ft

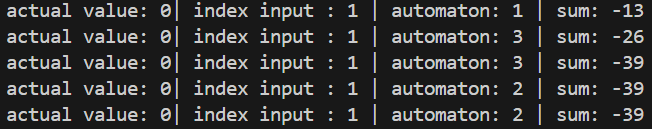
A black background with white text

Description automatically generated

Result of 1 3 aaabb : 208

Strategy:

In left:



And in right:

A screenshot of a computer program

Description automatically generated

**CASE 2**

6

meet meat eat eating tiny in

1 2 3 4 5 6

5

0 5 meetxxmeatxxeatingxxtiny

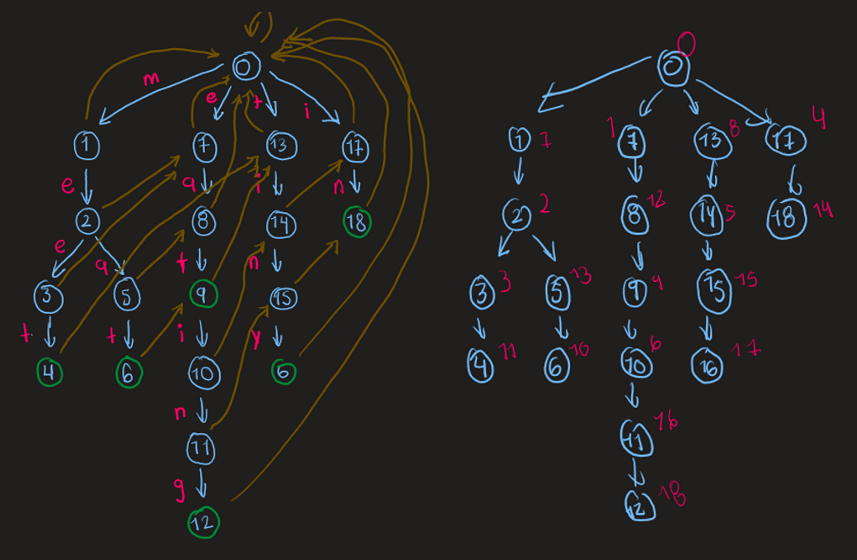
1 5 meetxxmeatxxeatingxxtiny

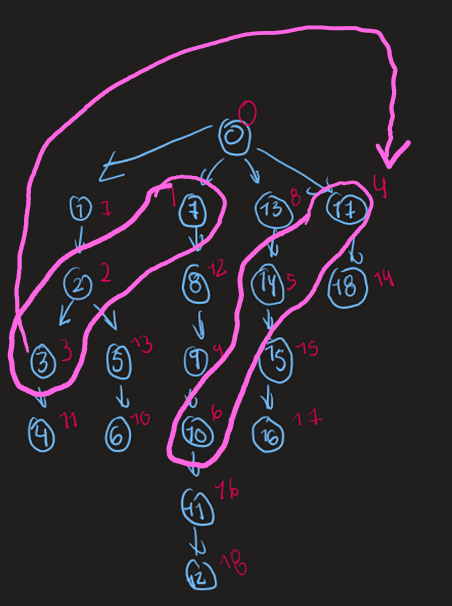
2 5 meetxxmeatxxeatingxxtiny

3 5 meetxxmeatxxeatingxxtiny

4 5 meetxxmeatxxeatingxxtiny

* Visualization of the Automaton, and the startStates.





* Notice how start states are obtain thanks to the failure links.
* Final States are obtained based on start states and subtrees. We are interested in start states and final states because, if you notice, for example, always that you pass through nodes 18, 15, 11, you obtain “in”, so add the value of “in”, 5, in the range 14, 15, 16 in the fenwick tree. In order to do that, update x in 14, and update -x in 17, in that way x is only in the range of 14-17.
* We do left and then right+1, because they need to be at least 1 of separation. Image the range 0-0. If you put 0, 0, the automaton has not yet insert values in the fenwick tree. it does from on 1.