Wavelet Tree Operations

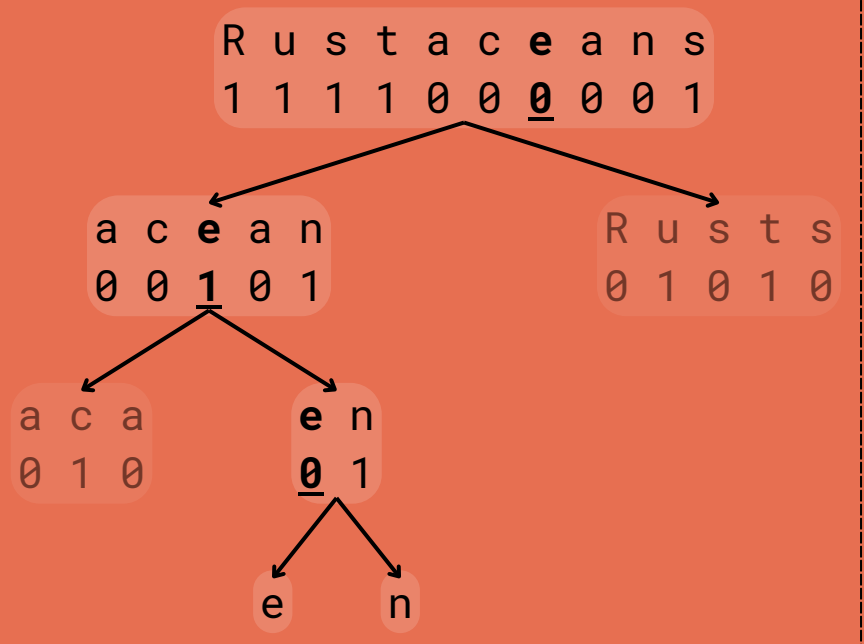
Part I: Access

Access: Zugriff auf das *i*-te Element des Wavelet Trees *S*

Am Beispiel „*Rustaceans*", mit *i* = 7:

Idee:

- Wenn aktuelle *Node* ein *Leaf* ist, gib das Symbol des *Leaves* zurück
- Wenn *Bitmap* an Stelle *i* gleich *0*, dann setze *i* auf die Anzahl der Nullen bis zur Stelle *i* und rufe *Access* auf für die linke *Node*
- Andernfalls setze *i* auf die Anzahl der Einsen bis zur Stelle *i* und rufe *Access* auf für die rechte *Node*




```
    i = 7
rank_0 = 3

    i = 3
rank_1 = 1

    i = 1
rank_0 = 1

Access = e
```

Wavelet Tree Operations

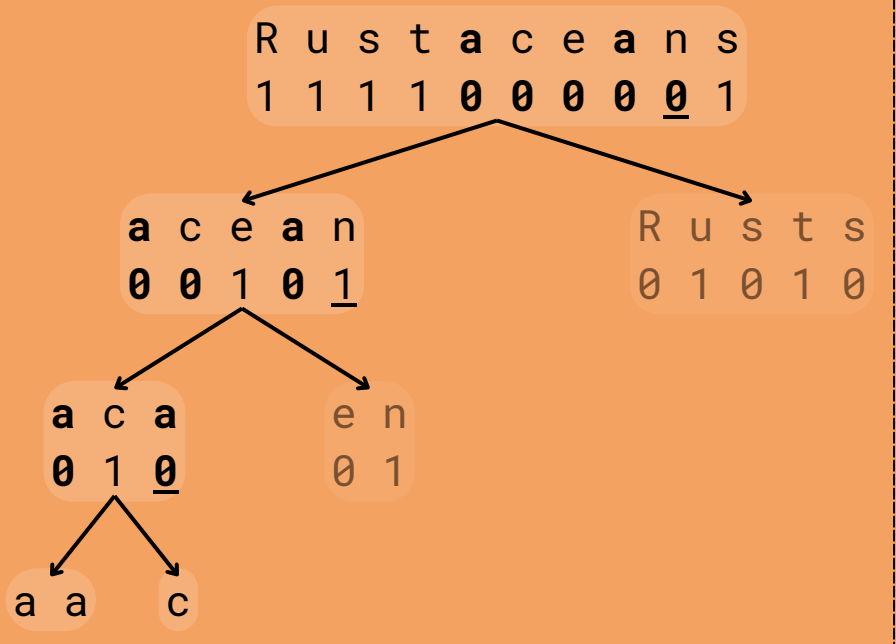
Part II: Rank

Rank: Bestimmen der Anzahl an Vorkommen eines Symbols *a* bis zur Stelle *i*

Am Beispiel *„Rustaceans"*, mit *a* = a, *i* = 9:

Idee:

- Wenn aktuelle *Node* ein *Leaf* ist, gib *i* zurück
- Wenn *a* in der aktuellen *Bitmap* einer *0* entspricht, setze *i* auf die Anzahl der Nullen bis zur Stelle *i* und rufe *Rank* auf für die linke *Node*
- Andernfalls setze *i* auf die Anzahl der Einsen bis zur Stelle *i* und rufe *Rank* auf für die rechte *Node*



i = 9
rank_0 = 5

i = 5
rank_0 = 3

i = 3
rank_0 = 2

Rank = 2

Wavelet Tree Operations

Part III: Select

Select: Bestimmen der Position des *j*-ten Vorkommens des Symbols *a*

Am Beispiel *„Rustaceans"*, mit *a* = s, *j* = 1:

Idee:

- Wenn aktuelle *Node* ein *Leaf* ist, gib *j* zurück
- Wenn *a* in der aktuellen *Bitmap* einer *0* entspricht, setze *j* gleich der Rückgabe von *Select* für die linke *Node*; anschließend gebe die Position der *j*-ten *0* der *Bitmap* zurück
- Andernfalls setze *j* gleich der Rückgabe von *Select* für die rechte *Node*; anschließend gebe die Position der *j*-ten *1* der *Bitmap* zurück

Leserichtung

```
Select = 3

select_1 = 3
       j = 3

select_0 = 3
       j = 2

select_1 = 2
       j = 1

       j = 1
```