CONTEXT FREE GRAMMAR:

A context free grammar **(CFG)** can be define in 4-tuple (S, T, N, P) where -

- **S** is a special variable called the **Start symbol**, S ∈ N *(S is a representation of start symbol as we have q0 in FA, it name may be change)*
- **T** or ∑ is a finite non-empty set of **Terminal symbols.**
  - Represented by Small letters (a to z) E.g. ∑ = {a, b}
  - ***OR*** Represented by digits (0 to 1) E.g. ∑ = {0, 1}
  - ***OR*** *special symbols* E.g. $, #, @ etc.
  - *It is always on right side of production.*
- **N** or **VN** is a finite non-empty set of variables or **non-terminal symbols.**
  - *Represented by Capital letters (A to Z) E.g.* {S, A}
  - *It is always on left side of production, but also possible on right side as well.*
- **P** is Production rules for Terminals and Non-terminals.
  - A production rule can be in the form of **α → β** where

    α → β (**α** determine **β**)

    α ∈ N , **α** must be only 1

    β ∈ (∑∪ N)*

For Example:

- S → Aa
- A → a
- A → b
- A → ε
- A → 0/1/2/3

The above is the example of CFG, each line is called **rule of grammar**, due to that we define grammar as; "a set of rules and regulations".

Backus-Nour Form / Backus Normal Form:

It states that if left side of production is same then we can write it together instead of separate.

For Example:

- S → A
- A → a
- A → b

- $A \rightarrow \varepsilon$

We notice that on left side of production letter are same, then we can write it together, like

- $A \rightarrow a / b / \varepsilon$

Why we use Context Free Grammar?

- Before going to main question we must know that what is regular languages and non-regular languages?
- **Regular languages** are those languages for which we can draw Finite automata
- (NFA/DFA)
- **Non-Regular languages** are those languages for which we cannot draw FA.
- Now the question is that how to handle Non-Regular languages, answers is simple to handle Non-Regular languages context-free grammar is use.

Examples of Context-Free Grammar

How to generate a string from CFG?

1. Let's we have given a grammar.

- $S \rightarrow A$
- $A \rightarrow A+A$
- $A \rightarrow A*A$
- $A \rightarrow 0/1/2/3/4/5/6/7/8/9$

- And tell us that generate the string "2+3*5" from above grammar.

Let's solve it...

- S → **A** // Replace the A with A+A, follow above grammar)
- A → A+**A** // Now replace right A with A*A, follow grammar)
- A → **A**+A*A // Now replace left A with 2, follow grammar)
- A → 2+**A***A // Now replace middle A with 3, follow grammar)
- A → 2+3***A** // Now replace right A with 5, follow grammar)
- A → 2+3*5 // finally, we got the desire string)

2. Let's we take another example, suppose we have given a grammar.

- S → A
- A → ε
- A → 0 / 1
- A → 0 A 0
- A → 1 A 1

❖ And tell us that generate the string "0110" from above grammar.

Let's solve it...

S → **A** // Replace the **A** with **0A0,** follow above grammar)

A → 0**A**0 // Now replace right **A** with **1A1**, follow grammar)

A → 01**A**10 // Now to remove the **A**, use **ε**)

A → 01**ε**10 // **ε** not effect any string)

A → 0110 // finally, we got the desire string)

How to convert Regular Expression to CFG?

---

1. Let's we have R.E.

R.E = a // Grammar start from S**,** it generate only "a"

CFG = S → a

2. Let's we have R.E.

R.E = 1

CFG = S → 1 // Grammar start from S**,** it generate only "1"

3. Let's we have R.E.

R.E = a + b

CFG = S → X / Y

X → a

Y → b

OR we can write it as well,

S → a / b

OR we can write it as well,

S → a

S → b

We know that, a+b = union and a.b / ab = concatenation in R.E, now in CFG, for union we use this symbol " / " and concatenation will be same.

4. Let's we have R.E.

R.E = (a+b) (a+b)

CFG = S → X X

X → A / B

A → a

B → b