A **Regular Expression** can be recursively defined as follows −

* **ε** is a Regular Expression indicates the language containing an empty string. **(L (ε) = {ε})**
* **φ** is a Regular Expression denoting an empty language. **(L (φ) = { })**
* **x** is a Regular Expression where **L = {x}**
* If **X** is a Regular Expression denoting the language **L(X)** and **Y** is a Regular Expression denoting the language **L(Y)**, then
  + **X + Y** is a Regular Expression corresponding to the language **L(X) ∪ L(Y)** where **L(X+Y) = L(X) ∪ L(Y)**.
  + **X . Y** is a Regular Expression corresponding to the language **L(X) . L(Y)** where **L(X.Y) = L(X) . L(Y)**
  + **R\*** is a Regular Expression corresponding to the language **L(R\*)**where **L(R\*) = (L(R))\***
* If we apply any of the rules several times from 1 to 5, they are Regular Expressions.

Some RE Examples

|  |  |
| --- | --- |
| **Regular Expressions** | **Regular Set** |
| (0 + 10\*) | L = { 0, 1, 10, 100, 1000, 10000, … } |
| (0\*10\*) | L = {1, 01, 10, 010, 0010, …} |
| (0 + ε)(1 + ε) | L = {ε, 0, 1, 01} |
| (a+b)\* | Set of strings of a’s and b’s of any length including the null string. So L = { ε, a, b, aa , ab , bb , ba, aaa…….} |
| (a+b)\*abb | Set of strings of a’s and b’s ending with the string abb. So L = {abb, aabb, babb, aaabb, ababb, …………..} |
| (11)\* | Set consisting of even number of 1’s including empty string, So L= {ε, 11, 1111, 111111, ……….} |
| (aa)\*(bb)\*b | Set of strings consisting of even number of a’s followed by odd number of b’s , so L = {b, aab, aabbb, aabbbbb, aaaab, aaaabbb, …………..} |
| (aa + ab + ba + bb)\* | String of a’s and b’s of even length can be obtained by concatenating any combination of the strings aa, ab, ba and bb including null, so L = {aa, ab, ba, bb, aaab, aaba, …………..} |