**Regular Expression**

1. **Quantifiers** 🡪 to define the quantity
2. \* - preceding text can occur zero or more time

Eg: pattern =[a-z]\* 🡪 “”,a,ab,abc,abcd,abcd………………(infinite)

1. + - preceding text can occur one or more times

Eg: pattern=[a-z]+ 🡪 ”a”,”ab”,”abc”,”abc…………………….”(infinite)

1. ? – preceding text may or may not be present (0 or 1 time)

Eg: pattern=[a-z]? 🡪 “123” , “123a”

1. {x} – preceding text will occur x times

Eg: pattern=[a-z]2 🡪 “ab” , “hg”

1. {x,y} – preceding text can occur min x times and max y time

Eg: pattern=[a-z]{2,4} 🡪 “ab”,”abc”,”abcd”

1. {x,}- preceding text can occur x and infinite times

Eg: pattern=[a-z]{2,} 🡪 “ab”, “abc”,”abcd……………….”

1. **Character classes** 🡪
2. \d 🡪 digits
3. \D 🡪 non-digits
4. \w 🡪 any words/strings/text 🡪 [a-zA-Z0-9\_]
5. \W 🡪 non words 🡪 [^a-zA-Z0-9\_]
6. ^ 🡪Negation – it will not allow the digits / alphabets which is defined under negation.

Eg: pattern = [^1-5]{3,5} 🡪 067,0678,06789

1. **Anchors**
2. . 🡪 wild card character (if we provide . I can use any one character)

Eg: “1”,”a”,”#”

.\* 🡪 “2”,”22”,”22…………..”

.+ 🡪 “1”,”11”,”11………….”

1. ^ 🡪 start matching from here

Eg: pattern =^[0-9]{2} 🡪 12,23,53,53

1. $ 🡪 ending

Eg: pattern = [0-9]{2}$ 🡪 12,23,53,44

1. | 🡪 or
2. **Escape characters**
   1. \ - escape character

Eg: pattern = \\*{3} 🡪 \*\*\*

* 1. [] 🡪 to define range or set of eligible chars
  2. {} 🡪 to define quantity
  3. () 🡪 to group

Examples: Email = [bellam.naveen.babu@company.com](mailto:bellam.naveen.babu@company.com)

1. bellam.naveen.babu 🡪 [a-z0-9]+
2. @ 🡪 @
3. company 🡪 [a-z]+
4. . 🡪 \.
5. com 🡪 [a-z]+ (or) [a-z]{2,3} 🡪 Strict checking (com | in)

**pattern = [a-z0-9]+@ [a-z]+ \.[a-z]+**