Splitting Algorithm:-

1. splited Reg = split_Regex ()
2. NFA = creat_NFA_from_splits ()
3. Merge(NFA)

split_Regex (Regex) → ~~array of strings~~ (new Reg) map

1. iterate over each character in Regex
   1. if c == '(' ✓
      1. bracket is open = True // we should handle // the neasted brackets
   2. if c == ')' ✓
      1. list.append (created_str)
         bracket is open = false $f1
   3. else
      created_str += c

→ take care of symbols (→ * → + ?) → should be treated

وحطهم جوا ال (map) كل حرف الرموز الصريحة
بتشاور على الحروف بتاعتنا

f1 → (abc)
f2 → (f1 | f2)

create_NFA_from_splits ()
1. Zu Z Algo for each symbol
2. Merge ✓✓

f1 →

NFA Algorithm:

1. Assuming No symbols or brackets For now.

1. Define a bool indicates whether we should insert empty state → is Epsilon
2. Define a list which contains our states → [] empty.
3. initialize characters idx which points To the current character of the given regular expression with -1
   - we will do our step actions
4. Create initial state S1 (init=T, Term= F, trans = {})
5. Append it to the states list
6. iterate For each state in the states list
   - 6.1. apply logic (state, char idx, is Epsilon, regex, states
   - 6.2. is Epsilon = ~ is Epsilon → toggle it
   - 6.3. char idx += ! is Epsilon we move the pointer in case that we are not @ Epsilon state

// we should continue the NFA algo, but lets stop here now & then return

Implement apply logic

---

apply logic (state, char idx, is Epsilon, regex, states

1. create new empty state → Si
2. Append it to the states list → states. append [Si]
3. if (is Epsilon)
   - 3.1. state.Trans ['ε'] . append [Si]
4. else
   - 4.1. if (regex [char idx + 1] is alphanumeric)
     - 4.1.1. state. Trans [*]. append [Si]
   - 4.2. else
     // we should think how to handle these cases.

→ expand_square_brackets (str → [data])

1. Two pointers s & f = 0, 1

2. while (s < str.len)

   1. if (f == '-')
      f++
      for (c = s, c < f, c++)
         retStr += 'c' or

   2. elif (f == str.len)
      retStr += str(s)

   3. elif (s != f)
      s++
      f++

a-zA-Z0-9

retStr

---

class_ Stato

↳ JSON