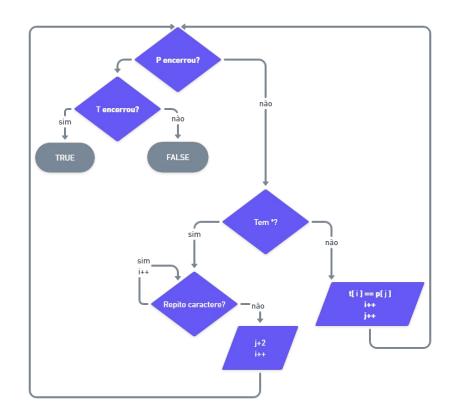
## Regular Expression Matching

Simulação

## Teste 1





8.

9.

j))

else:

def match(i: int, j: int)-> bool:

return first\_match and match(i+1, j+1)

```
t: a a p: a
```

7.

8.

9.

j))

else:

return match(i, j+2) or (first\_match and match(i+1,

return first\_match and match(i+1, j+1)

```
t: a p: a *
```

def match(i: int, j: int)-> bool:

8.

9.

8.

9.

j))

else:

def match(i: int, j: int)-> bool:

return first\_match and match(i+1, j+1)

```
t: a a p: a
```

j))

else:

8.

9.

def match(i: int, j: int)-> bool:

return first\_match and match(i+1, j+1)

```
t: a a p: a
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. TRUE first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

```
t: a a p: a *
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
         else:
4.
    TRUE first_match = i < len(text) and pattern[j] in {text[i],
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return(match(i, j+2)) or (first_match and match(i+1,
7.
     j))
8.
           else:
```

```
t: a a p: a * ____
```

```
1.     def match(i: int, j: int)-> bool:
2.         if (j == len(pattern)):
3.             return ( i == len(text) )
4.         else:
5.             first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.             if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
```

return match(i, j+2) or (first\_match and match(i+1,

return first\_match and match(i+1, j+1)

7.

8.

9.

j))

```
t: a a p: a *
```

```
t: a a p: a
```

- 1. def match(i: int, j: int)-> bool:
  2. if (j == len(pattern)):
  3. return ( i == len(text) )
  4. else:
  5. True first\_match = i < len(text) and pattern[j] in {text[i], '.'}
  6. if (j+1 < len(pattern) and pattern[j+1] == '\*'):
  7. return match(i, j+2) or (first\_match and match(i+1, j))</pre>
  - 8. else:9. return first\_match and match(i+1, j+1)

```
t: a a p: a
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
    True first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
              return match(i, j+2) or (first_match ard match(i+1, False True
7.
     j))
8.
            else:
```

```
t: a a p: a
```

```
def match(i: int, j: int)-> bool:
1.
2.
         1f (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
5.
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a p: a
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. True    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. True    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

```
t: a a p: a *
```

```
def match(i: int) j: int)-> bool:
1.
2.
         if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a p: a *
```

```
def match(i: int, j: int)-> bool:
1.
2.
          if (j == len(pattern)):
                                          False
           return ( i == len(text)
3.
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
    True first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return(match(i, j+2)) or (first_match and match(i+1,
7.
     j))
                        False
8.
           else:
```

```
t: a a p: a
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
    True first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
6.
           if (j+1 < len(pattern) and pattern[j+1] == '*');</pre>
7.
             return match(i, j+2) or (first_match and (match(i+1,
                        False
                                          True
     j))
```

8.

9.

```
t: a a p: a

aa a_
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5.        first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

```
t: a a p: a

aa a_
```

8.

9.

```
t: a a p: a

aa

aa

aa
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5.False    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

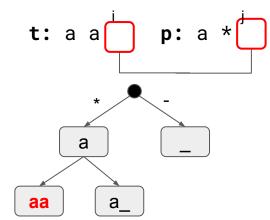
9.

```
t: a a p: a *
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

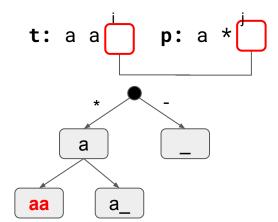
```
t: a a p: a *
```

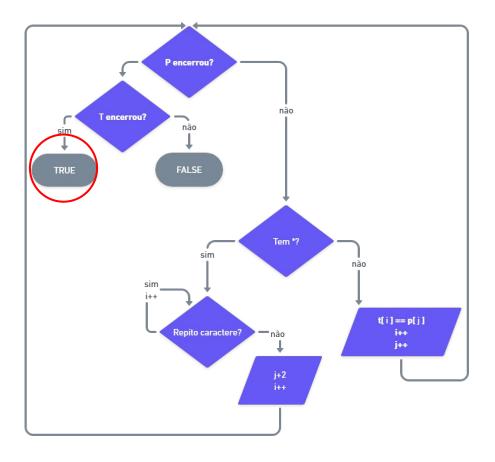
```
def match(i: int. i: int)-> bool:
1.
         if (j == len(pattern)):
2.
3.
                         -- len(text)
            return ( i
          else:
4.
5.
            first_match = i < len(text) and pattern[j] in {text[i],</pre>
      '.'}
            if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
               return match(i, j+2) or (first_match and match(i+1,
7.
      j))
8.
            else:
9.
               return first_match and match(i+1, j+1)
                                             P encerrou?
                                       T encerrou?
                                              Repito caractere?
```



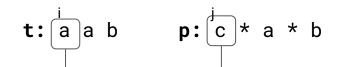
```
def match(i: int, j: int)-> bool:
1.
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
                                          True
4.
         else:
5.
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
9.
              return first_match and match(i+1, j+1)
                                          P encerrou?
```

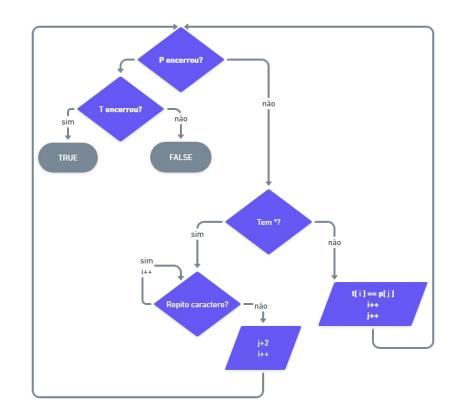
Repito caractere?





## Teste 2





```
2.
        if (j == len(pattern)):
           return ( 1 == len(text) )
3.
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
9.
              return first_match and match(i+1, j+1)
```

def match(i: int, j: int)-> bool:

```
a b
            p:
                                        2.
                                                 if (j == len(pattern)):
                                        3.
                                                  return ( i == len(text) )
                                        4.
                                                 else:
                                        5. False (first_match) i < len(text) and pattern[j] in {text[i],
                                             '.'}
                                                   if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                        6.
                                        7.
                                                     return match(i, j+2) or (first_match and match(i+1,
                                             j))
                                        8.
                                                   else:
                                        9.
                                                     return first_match and match(i+1, j+1)
```

def match(i: int, j: int)-> bool:

```
2.
         if (j == len(pattern)):
3.
         return ( i == len(text) )
4.
         else:
5. False first_match = i < len(text) and pattern[j] in {text[i],
     '.'}
          if (j+1 < len(pattern) and pattern[j+1] == '*'</pre>
6.
             return match(i, j+2) or (first_match and match(i+1,
7.
     j))
8.
           else:
9.
             return first_match and match(i+1, j+1)
```

def match(i: int, j: int)-> bool:

```
t: a a b p: c
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
         return ( i == len(text) )
4.
         else:
5. False first_match = i < len(text) and pattern[j] in {text[i],
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return(match(i, j+2)) or (first_match and match(i+1,
7.
     j))
8.
           else:
```

```
t: a a b p: c * a *

c -= ""
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
         return ( i == len(text) )
4.
         else:
5. False first_match = i < len(text) and pattern[j] in {text[i],
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return(match(i, j+2)) or (first_match and match(i+1,
7.
     j))
8.
           else:
```

```
t: a a b p: c * a *
```

```
def match(i:)int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a b p: c * a *
```

return match(i, j+2) or (first\_match and match(i+1,

def match(i: int, j: int)-> bool:

7.

8.

9.

j))

else:

```
t: a a b p: c * a *
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5. True
          (first_match ) i < len(text) and pattern[j] in {text[i],</pre>
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a b p: c * a *

c -
```

9.

else:

```
t: a a b p: c * a
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5. True
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return(match(i, j+2)) or (first_match and match(i+1,
7.
     j))
8.
           else:
```

```
t: a a b p: c * a * b
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5.        first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
t: a a b p: c * a * b
```

```
def match(i: int, j: int)-> bool:
2.
        if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
9.
             return first_match and match(i+1, j+1)
```

```
t: a a b p: c * a * b
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5. False
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a b p: c * a * b
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. False    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
t: a a b p: c * a * b
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5. False first_match = i < len(text) and pattern[j] in {text[i],
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
                       False
8.
           else:
```

```
t: a a b p: c * a
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5. True
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return(match(i, j+2)) or (first_match and match(i+1,
7.
     j))
                         False
8.
           else:
```

```
t: a a b p: c * a * b
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5. True
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
6.
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
             return match(i, j+2) or (first_match an match(i+1,
7.
                         False
                                           True
     j))
8.
           else:
```

```
t: a a b p: c * a
```

```
def match(i: )nt, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a b p: c * a
```

```
def match(i: int, j: int)-> bool:
2.
        if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a b p: c * a
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
          (first_match) i < len(text) and pattern[j] in {text[i],</pre>
5. True
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a b p: c * a
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. True    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

else:

```
t: a a b p: c * a * b
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
5. True
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             returr( match(i, j+2)) or (first_match and match(i+1,
7.
     j))
8.
           else:
```

```
t: a a b p: c * a * b
```

aa

8.

9.

else:

```
1. def match(i:)int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5.        first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
t: a a b p: c * a *
```

9.

else:

```
t: a a b p: c * a * b
```

aa

8.

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. False    first_match i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
t: a a b p: c * a * b
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. False    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
p: c * a *
С
    a
```

aa

8.

9.

else:

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
         else:
4.
5. False first_match = i < len(text) and pattern[j] in {text[i],
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
                       False
```

```
t: a a b p: c * a * b
```

def match(i: int, j: int)-> bool: 2. if (j == len(pattern)): 3. return ( i == len(text) ) 4. else: first\_match = i < len(text) and pattern[j] in {text[i],</pre> 5. True '.'} if (j+1 < len(pattern) and pattern[j+1] == '\*'):</pre> 6. returr( match(i, j+2)) or (first\_match and match(i+1, 7. j)) False 8. else:

return first\_match and match(i+1, j+1)

```
t: a a b p: c * a * b
```

- def match(i: int, j: int)-> bool: 2. if (j == len(pattern)): 3. return ( i == len(text) ) else: 4. 5. True first\_match = i < len(text) and pattern[j] in {text[i],</pre> '.'} 6. if (j+1 < len(pattern) and pattern[j+1] == '\*'):</pre> return match(i, j+2) or (first\_match and match(i+1, 7. False True j))
  - 9. return first\_match and match(i+1, j+1)

else:

```
p: c *
С
     a
aa
```

9.

```
def match(); int, j: int)-> bool:
2.
          if () == len(pattern)):
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
t: a a b p: c * a *

c a a -
```

```
def match(i: int, i: int)-> bool:
2.
        if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return match(i, j+2) or (first_match and match(i+1,
7.
     j))
8.
           else:
```

```
t: a a b p: c * a * b
```

- - 9. return first\_match and match(i+1, j+1)

else:

```
t: a a b p: c * a * b
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. False    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

else:

```
p: c *
                         * b
     С
           a
     aa
          aa_
aaa
```

- 1. def match(i: int, j: int)-> bool:
  2. if (j == len(pattern)):
  3. return ( i == len(text) )
  4. else:
  5. False first\_match = i < len(text) and pattern[j] in {text[i], '.'}
  6. if (j+1 < len(pattern) and pattern[j+1] == '\*'):
  7. return match(i, j+2) or (first\_match and match(i+1, j))</pre>
  - 9. return first\_match and match(i+1, j+1)

else:

```
p: c * a *
t: a a
       С
            a
      aa
           aa_
 aaa
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

```
p: c * a *
t: a a
       С
            a
      aa
           aa_
 aaa
```

9.

else:

```
p: c * a *
t: a a
       С
            a
      aa
           aa_
 aaa
```

9.

else:

```
p: c * a *
t: a a
       С
            a
      aa
           aa_
 aaa
```

- 9. return first\_match and match(i+1, j+1)

else:

```
p: c * a *
     С
          a
     aa
         aa_
aaa
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. True    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

True

else:

8.

```
p: c * a *
t: a a
       С
            a
      aa
           aa_
 aaa
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. True    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

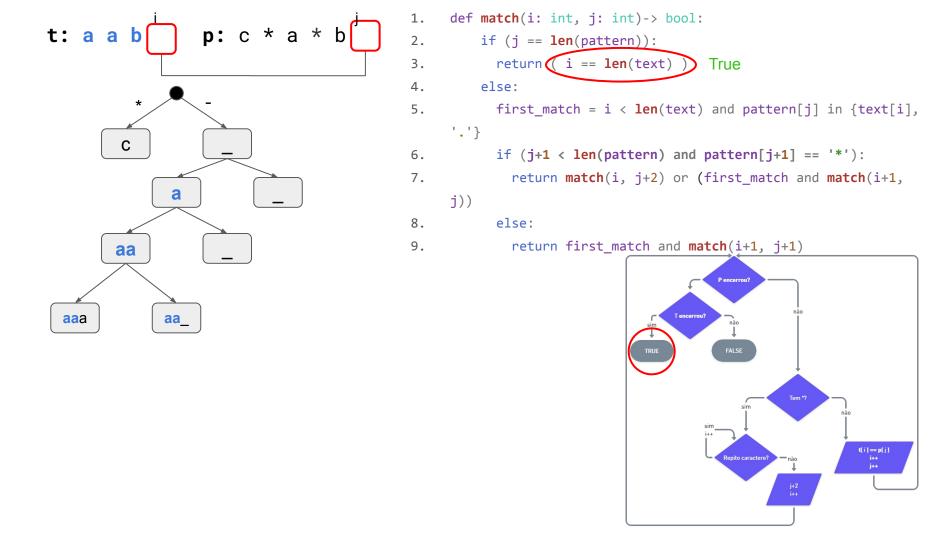
return first\_match and match(i+1, j+1)

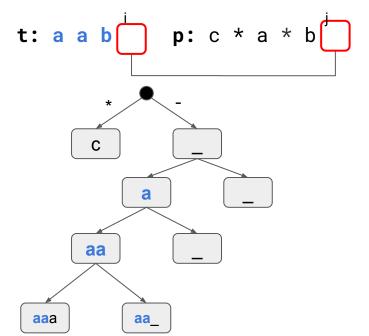
True

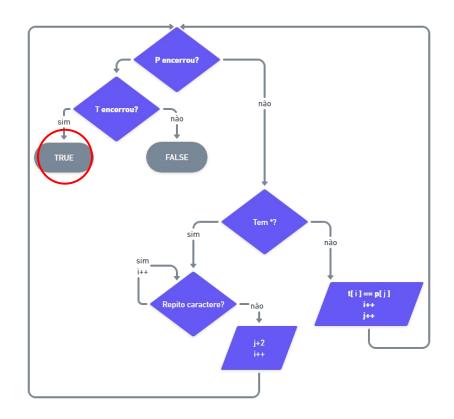
```
p: c * a * b
t: a a b
        *
       С
            a
      aa
           aa_
 aaa
```

```
def match(i: in)t, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
           return ( i == len(text) )
4.
         else:
5.
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
              return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

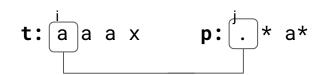
```
def match(i: int, j: int)-> bool:
                                               1.
                    p: c * a * b
t: a a b
                                               2.
                                                        if (j == len(pattern)):
                                               3.
                                                          return ( i == len(text) )
                                               4.
                                                         else:
                                               5.
                                                          first_match = i < len(text) and pattern[j] in {text[i],</pre>
         С
                                                          if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                               6.
                                               7.
                                                            return match(i, j+2) or (first_match and match(i+1,
                a
                                                    j))
                                               8.
                                                          else:
                                               9.
                                                            return first_match and match(i+1, j+1)
         aa
               aa_
  aaa
```



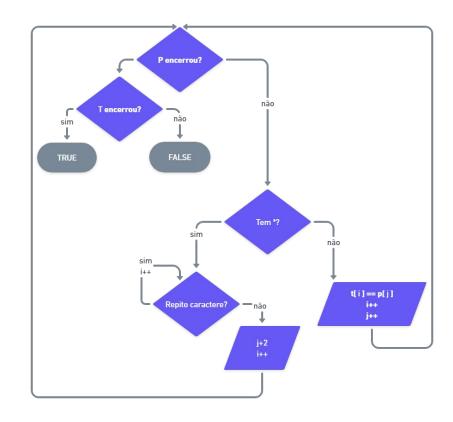




## Teste 3



Ta errado! Ç\_Ç)



```
a a x
                p:
                                         2.
                                                 if (j == len(pattern)):
                                         3.
                                                    return ( i == len(text) )
                                         4.
                                                   else:
                                         5.
                                                    first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                               '.'}
                                                    if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                         6.
                                         7.
                                                       return match(i, j+2) or (first_match and match(i+1,
                                              j))
                                         8.
                                                    else:
                                         9.
                                                       return first_match and match(i+1, j+1)
```

def match(i: int, j: int)-> bool:

```
a a x
                p:
                                         2.
                                                  if (j == len(pattern)):
                                         3.
                                                   return ( i == len(text) )
                                         4.
                                                  else:
                                                   first_match i < len(text) and pattern[j] in {text[i],</pre>
                                         5. True
                                                    if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                         6.
                                                      return match(i, j+2) or (first_match and match(i+1,
                                        7.
                                              j))
                                         8.
                                                    else:
                                         9.
                                                      return first_match and match(i+1, j+1)
```

def match(i: int, j: int)-> bool:

```
def match(i: int, j: int)-> bool:
a a x
                p:
                                        2.
                                                 if (j == len(pattern)):
                                        3.
                                                  return ( i == len(text) )
                                        4.
                                                 else:
                                                   first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                        5. True
                                                if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                        6.
                                        7.
                                                      return match(i, j+2) or (first_match and match(i+1,
                                              j))
                                        8.
                                                    else:
                                        9.
                                                      return first_match and match(i+1, j+1)
```

```
t: a a a x p: . * a*

a - - = ""
```

def match(i: int, j: int)-> bool:

9.

```
def match)i: int, j: int)-> bool:
1.
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
         else:
4.
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
5.
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
9.
             return first_match and match(i+1, j+1)
```

```
t: a a a x p: . * a

* -= ""
```

8.

9.

```
t: a a a x p: . * a - = ""
```

8.

9.

```
t: a a a x p: . * a

*
-= ""
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. TRUE    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
(def match(i) int, j: int)-> bool:
                                         1.
                        * a *
                p: .
a a x
                                         2.
                                                  if (j == len(pattern)):
                                         3.
                                                    return ( i == len(text) )
                                         4.
                                                  else:
                                                    first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                         5.
                                              '.'}
  а
                                                    if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                         6.
                                        7.
                                                      return match(i, j+2) or (first_match and match(i+1,
         а
                                              j))
                                         8.
                                                    else:
                                         9.
                                                      return first_match and match(i+1, j+1)
```

```
def match(i: int, j: int)-> bool:
                        * a *
                p: .
a a x
                                        2.
                                                 if (j == len(pattern)):
                                        3.
                                                   return ( i == len(text) )
                                        4.
                                                  else:
                                                   first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                        5.
                                              '.'}
  а
                                                   if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                        6.
                                        7.
                                                      return match(i, j+2) or (first_match and match(i+1,
         а
                                              j))
                                        8.
                                                   else:
                                        9.
                                                      return first_match and match(i+1, j+1)
```

```
def match(i: int, j: int)-> bool:
                        * a *
                p: .
a a x
                                        2.
                                                 if (j == len(pattern)):
                                                                               False
                                                   return ( i == len(text)
                                        3.
                                                 else:
                                        4.
                                                   first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                        5.
                                              '.'}
  а
                                                   if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                        6.
                                        7.
                                                      return match(i, j+2) or (first_match and match(i+1,
         а
                                              j))
                                        8.
                                                   else:
                                        9.
                                                      return first_match and match(i+1, j+1)
```

```
t: a a a x p: . * a

-= ""
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
5. True
     '.'}
6.
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
             return match(i, j+2) or (first_match and (match(i+1,
7.
                         False
                                          True
     j))
```

8.

9.

```
t: a a a x p: . * a
```

```
1. def match(i: int j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5.        first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

8.

9.

```
t: a a a x p: . * a

-= "
```

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. True    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

8.

9.

```
def match(i: int) j: int)-> bool:
               p: . * a *
a a x
                                        2.
                                                 if (j == len(pattern)):
                                        3.
                                                  return ( i == len(text) )
                                        4.
                                                 else:
                                                   first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                        5.
                                             '.'}
  а
                                                   if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                        6.
                                        7.
                                                     return match(i, j+2) or (first_match and match(i+1,
        a
                                             j))
                                        8.
                                                   else:
 aa
                                        9.
                                                     return first_match and match(i+1, j+1)
```

```
def match(i: int, j: int)-> bool:
               p: . * a *
a a x
                                        2.
                                                 if (j == len(pattern)):
                                                                              False
                                        3.
                                                   return ( i == len(text)
                                                 else:
                                        4.
                                                   first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                        5.
                                             '.'}
  а
                                                   if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                        6.
                                        7.
                                                     return match(i, j+2) or (first_match and match(i+1,
        a
                                             j))
                                        8.
                                                   else:
 aa
                                        9.
                                                     return first_match and match(i+1, j+1)
```

```
t: a a x p: . * a

a _ _ = "
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
         return ( i == len(text) )
4.
         else:
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
5. True
     '.'}
6.
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
             return match(i, j+2) or (first_match and (match(i+1,
7.
                                          True
                         False
     j))
```

8.

9.

```
t: a a a x p: . * a

-= "
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5.        first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.        if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.        return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. True    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
def match(i: int), j: int)-> bool:
                       p: . * a *
t: a a a x
                                                2.
                                                         if (j == len(pattern)):
                                                3.
                                                          return ( i == len(text) )
                                                4.
                                                         else:
                                                5.
                                                           first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                                     '.'}
          а
                                                           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                                6.
                                                7.
                                                             return match(i, j+2) or (first_match and match(i+1,
                a
                                                     j))
                                                8.
                                                           else:
         aa
                                                9.
                                                             return first_match and match(i+1, j+1)
 aaa
               aa_
```

```
def match(i: int, j: int)-> bool:
                       p: . * a *
t: a a a x
                                                2.
                                                         if (j == len(pattern)):
                                                3.
                                                           return ( i == len(text)
                                                                                       False
                                                         else:
                                                4.
                                                           first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                                5.
                                                     '.'}
          а
                                                           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                                6.
                                                7.
                                                             return match(i, j+2) or (first_match and match(i+1,
                a
                                                     j))
                                                8.
                                                           else:
         aa
                                                9.
                                                             return first_match and match(i+1, j+1)
 aaa
               aa_
```

```
t: a a a x
                  p: .
       а
            a
       aa
 aaa
           aa_
```

9.

else:

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
5. True
     '.'}
6.
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
7.
             return match(i, j+2) or (first_match and match(i+1,
                       False
                                          True
     j))
```

```
t: a a a x
                 p: .
       а
            a
       aa
 aaa
           aa_
```

```
(def match(i) int, j: int)-> bool:
1.
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5.
           first_match = i < len(text) and pattern[j] in {text[i],</pre>
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
     j))
8.
           else:
```

9.

```
t: a a a x
                  p: .
       а
            a
       aa
 aaa
           aa_
```

9.

else:

```
1. def match(i: int, j: int)-> bool:
2.    if (j == len(pattern)):
3.        return ( i == len(text) )
4.    else:
5. False    first_match = i < len(text) and pattern[j] in {text[i], '.'}
6.    if (j+1 < len(pattern) and pattern[j+1] == '*'):
7.    return match(i, j+2) or (first_match and match(i+1, j))</pre>
```

```
t: a a a x
                 p: .
       а
            a
       aa
 aaa
           aa_
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
5. False first_match = i < len(text) and pattern[j] in {text[i],
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
             return (match(i, j+2)) or (first_match and match(i+1,
7.
     j))
8.
           else:
```

```
def match(i: int, j: int)-> bool:
                                                1.
                       p: . * a *
t: a a a x
                                                2.
                                                         if (j == len(pattern)):
                                                3.
                                                          return ( i == len(text) )
                                                4.
                                                         else:
                                                5.
                                                           first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                                     '.'}
          а
                                                           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                                6.
                                               7.
                                                             return match(i, j+2) or (first_match and match(i+1,
                a
                                                     j))
                                                8.
                                                           else:
         aa
                                                9.
                                                             return first_match and match(i+1, j+1)
 aaa
               aa_
```

```
def match(i: int, j: int)-> bool:
                       p: . * a *
t: a a a x
                                                2.
                                                         if (j == len(pattern)):
                                                           returr ( i == len(text) )
                                                3.
                                                                                       False
                                                         else:
                                                4.
                                                          first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                                5.
                                                     '.'}
          а
                                                           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                                6.
                                               7.
                                                             return match(i, j+2) or (first_match and match(i+1,
                a
                                                     j))
                                                8.
                                                           else:
         aa
                                                9.
                                                             return first_match and match(i+1, j+1)
 aaa
               aa_
```

```
t: a a a | X
                   p: .
        а
             a
       aa
 aaa
            aa_
```

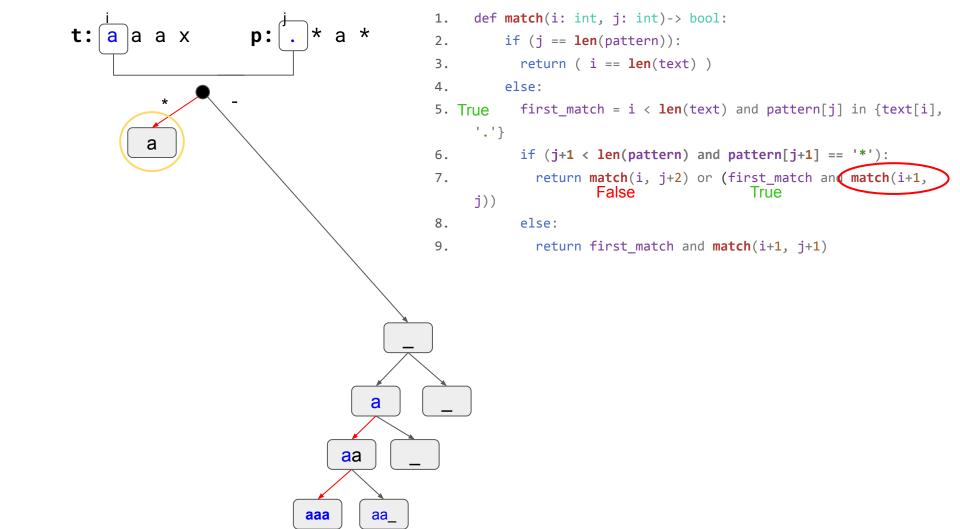
9.

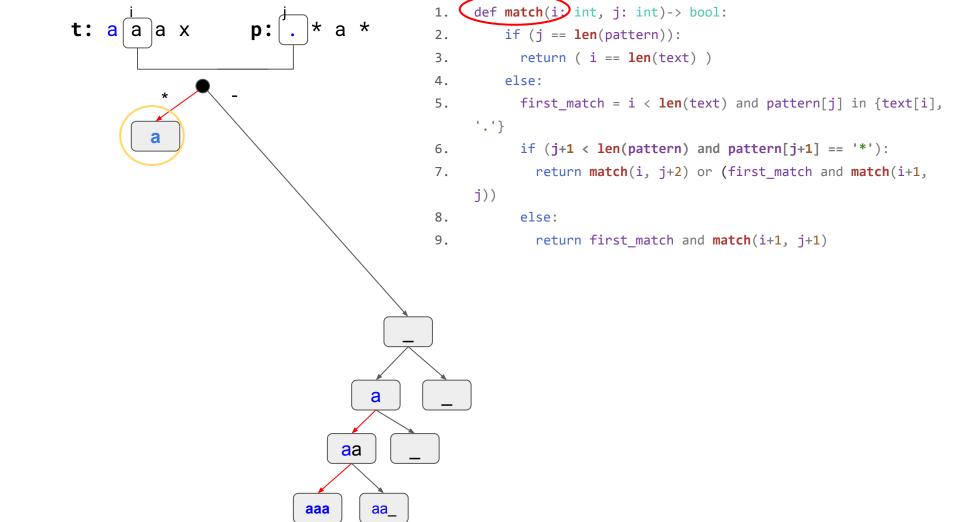
else:

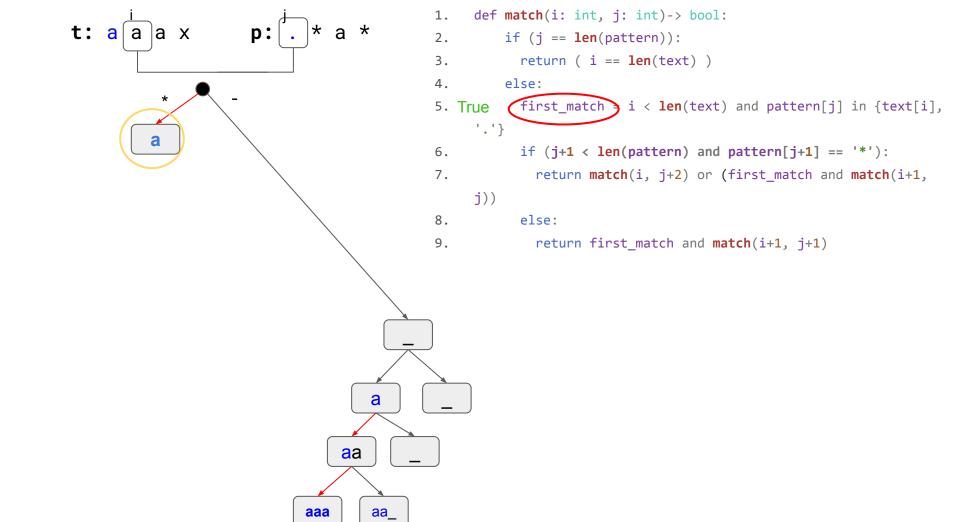
```
t: a a a x p: . * a *

a
(...)
```

```
def match(i: int, j: int)-> bool:
2.
         if (j == len(pattern)):
3.
          return ( i == len(text) )
4.
         else:
          first_match = i < len(text) and pattern[j] in {text[i],</pre>
5. True
     '.'}
           if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
6.
7.
             return match(i, j+2) or (first_match and match(i+1,
                     False
                                           True
     j))
8.
           else:
```







```
* a *
                p:
a a x
                                        2.
                                                 if (j == len(pattern)):
                                        3.
                                                  return ( i == len(text) )
                                        4.
                                                  else:
                                                   first_match = i < len(text) and pattern[j] in {text[i],</pre>
                                        5. True
                                                    if (j+1 < len(pattern) and pattern[j+1] == '*'):</pre>
                                        6.
                                                      return match(i, j+2) or (first_match and match(i+1,
                                        7.
                                              j))
                                        8.
                                                    else:
                                        9.
                                                      return first_match and match(i+1, j+1)
                                a
                            aa
```

aaa

aa\_

def match(i: int, j: int)-> bool:

