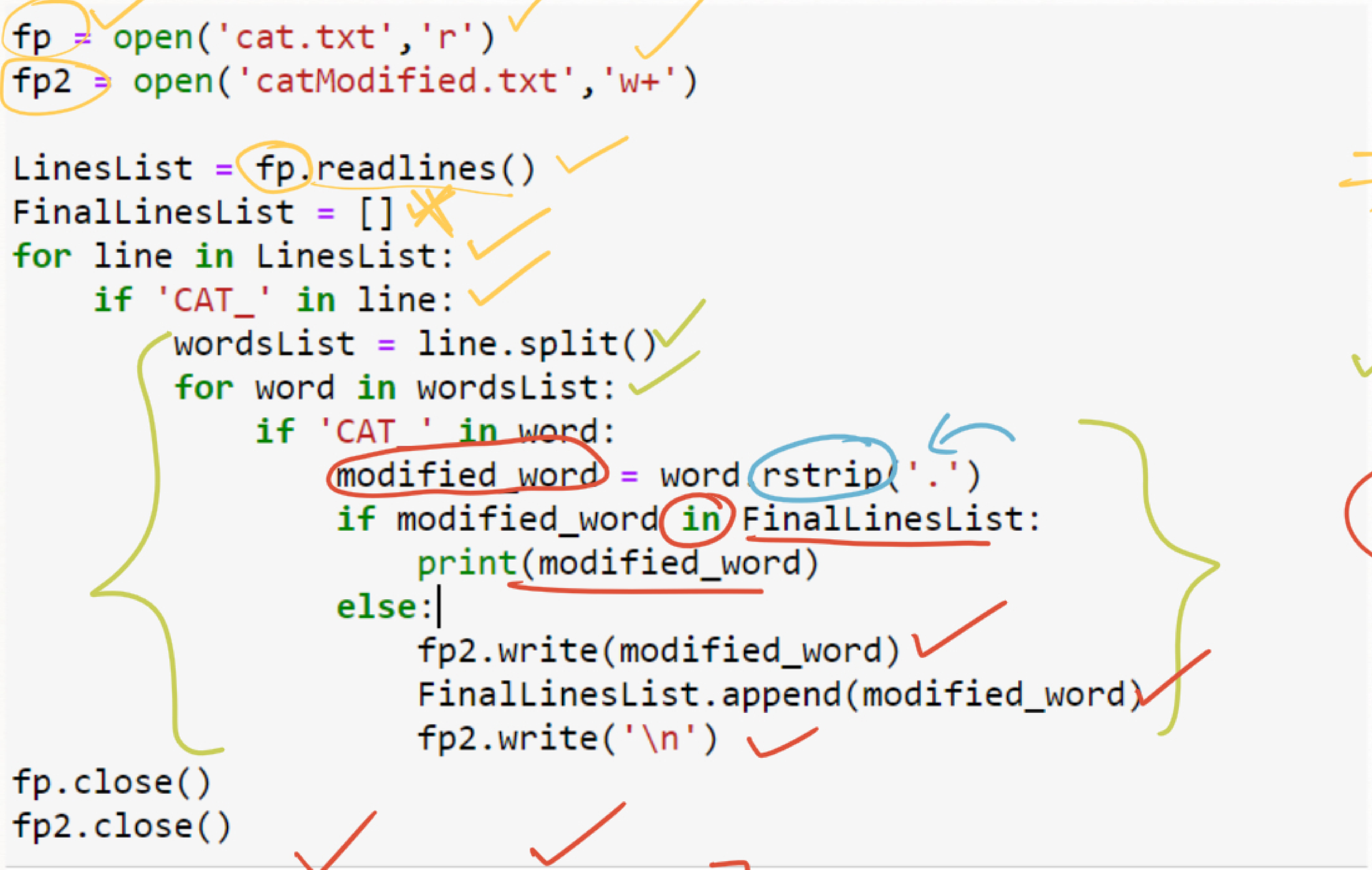


```python
fp = open('cat.txt','r')
fp2 = open('catModified.txt','w+')

LinesList = fp.readlines()
FinalLinesList = []
for line in LinesList:
    if 'CAT_' in line:
        wordsList = line.split()
        for word in wordsList:
            if 'CAT_' in word:
                modified_word = word.rstrip('.')
                if modified_word in FinalLinesList:
                    print(modified_word)
                else:
                    fp2.write(modified_word)
                    FinalLinesList.append(modified_word)
                    fp2.write('\n')
fp.close()
fp2.close()
```

['CATXY', 'CATAM']

LinesList = ['Ap CATXY.', 'X-CATAM', ~~'AB'~~]

=> CATXY ✓

CATAM ✓

✓ wordsList = [~~'Ap'~~, 'CATXY.']

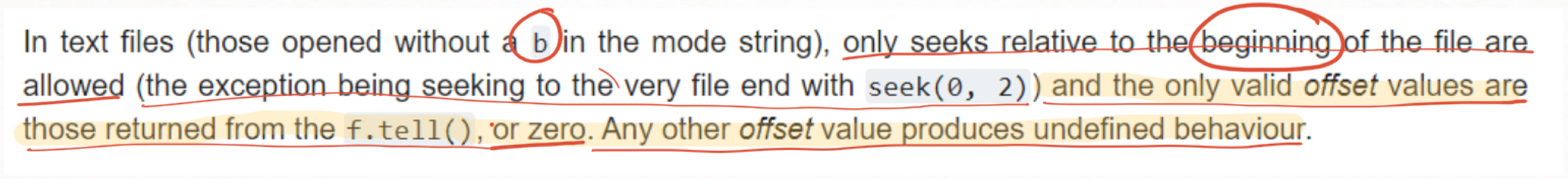
wordsList = [✗, ✗, 'CATAM']

~~'CATXY.'~~.rstrip('.')

=> CATXY ✓

Filepointer, fp.tell(), fp.seek()

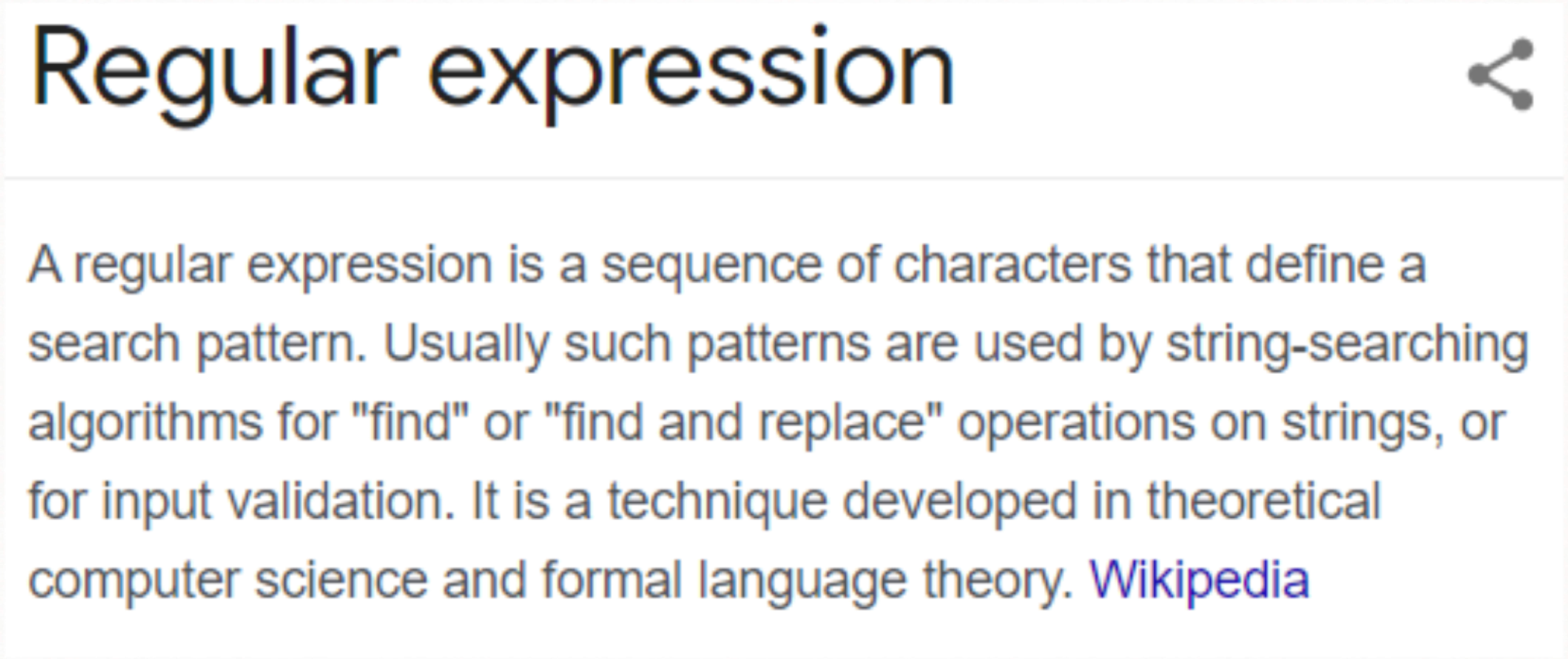
Tells the position



In text files (those opened without a `b` in the mode string), only seeks relative to the beginning of the file are allowed (the exception being seeking to the very file end with `seek(0, 2)`) and the only valid *offset* values are those returned from the `f.tell()`, or zero. Any other *offset* value produces undefined behaviour.

fp.seek() → Moves the position of fp relative to beginning of file

$$\text{fp.seek}(\overset{\text{fp.tell()+5}}{\text{offset}}, \text{relative\_position}) \quad \text{relative\_position} \rightarrow 0 \rightarrow \text{Beginning}$$



Regular expression

A regular expression is a sequence of characters that define a search pattern. Usually such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation. It is a technique developed in theoretical computer science and formal language theory. Wikipedia