ӘӨЖ 004. 832.22

ДЕРЕКТЕРДІ ТҰРАҚТЫ ӨРНЕКТЕР ТӘСІЛІМЕН ӨҢДЕУ

Құсайн Мархабат Қанатбекқызы
margo1998kz@gmail.com
Л.Н. Гумилев атындағы Еуразия Ұлттық Университеті, Ақпараттық технологиялар факультеті, “Информатика” мамандығының 2-курс магистранты
Ғылыми жетекшісі – т.ғ.к, доцент м.а. Разахова Б.Ш.

Аңдатпа: Бұл мақалада тұрақты өрнектердің көмегімен мәтіндерден әрі қарай талдау үшін қажетті нысандарды шығаруға болатын бірнеше мысалдар көрсетілді. Тұрақты өрнектер арнайы үлгілердің көмегімен үлкен мәтіндерден дұрыс мәліметтерді табуға көмектеседі.

Түйінді сөздер: тұрақты өрнектер, деректерді өңдеу, мәліметтер, re кітапханасы, үлгі, әдістер.

Тұрақты өрнектер (regexp немесе regex деп те аталады) мәтінді табу және ауыстыру механизмі болып табылады. Оны әзірлеушілер қолданба кодында, тестерлер автотесттерде және жай ғана пәрмен жолында жұмыс істегенде пайдаланады. Бұл әдіс қарапайым іздеуге қарағанда жақсырақ, себебі ол үлгіні орнатуға мүмкіндік береді [1].

Тұрақты өрнектер үлкен мәтіндерде дұрыс мәліметтерді табуға көмектеседі. Мысал ретінде бір саланы алайық. Ол салық болсын. Салық заңнамасының мәтіндерімен жұмыс істеп жатқанда аты-жөндер, даталар анықтау қажет болды делік, немесе, мысалы: "Мүлікке салықты төлеуді кешіктіргені үшін төлеушінің айыппұл санкцияларын қандай көлемде төлейтінін" табу қажет болса, тұрақты өрнектерді пайдалануға болады. Оның көмегімен қажетті субъектілерді табу үшін үлгілер тағайындауға болады.

Мысалдар Python ортасында орындалады.

Тұрақты өрнектермен жұмыс істеу үшін алдымен re кітапханасын импорттау керек. Бұл кітапханада бірнеше түрлі әдістер бар, жиі қолданылатын төрт әдіске назар аударайық [2]. Қалған әдістер желіде қолжетімді және оларды кітапхана құжаттамасынан толықтай оқуға болады.

Егер нақты бір нәрсені тапқыңыз келетін жол болса, алдымен кітапхананың атауын, содан кейін іздеу әдісін, ал жақшаларда үлгінің өзін және үтір арқылы іздегіңіз келетін жолды көрсету керек. Мысалы:

match () әдісі көрсетілген үлгіні жолдың басында ғана іздейді. Мысалы, “айыппұл салық айыппұл” деген жолда бұл әдіс “айыппұл” сөзін таба алады:

```
s = ‘айыппұл салық айыппұл’
result = re.match(r‘айыппұл’, s)
result.group (0)
```

НӘТИЖЕ: “айыппұл”

Бірақ “салық” сөзін іздеу үшін осы әдісті қолдансаңыз, ештеңе таппайды:

```
s = 'айыппұл салық айыппұл'
result = re.match(r'салық', s)
result.group (0)
```

НӘТИЖЕ: ҚАТЕ

search() әдісі де осы тапсырманы орындай алады: ол бүкіл жолды іздейді, бірақ ол тапқан алғашқы нәтижені ғана қайтарады. Егер “салық” сөзін іздесеңіз, тек бір ғана нәтиже қайтарылады.

```
s = 'айыппұл салық айыппұл'
result = re.search(r'салық', s)
result.group (0)
```

НӘТИЖЕ : ‘салық’

Іздеу критерийлеріне сай келетін барлық субъектілерді тапқыңыз келсе, findall() әдісін пайдалану қажет. Тізім түрінде ол табылған сәкестіктердің бәрін қайтарады.

```
s = 'айыппұл салық айыппұл'
result = re.findall(r'айыппұл', s)
result
```

НӘТИЖЕ : ['айыппұл', ' айыппұл"]

Ал тағы бір әдіс — sub(). Онымен табылған үлгіні басқа бір нәрсемен ауыстыруға болады. Мысалы, “Мүлік салығын есептеу және төлеу кезінде бұзушылықтар үшін төлеушілерге санкциялар қолданылады” деген жолда алдымен “санкциялар” деген сөзді тауып, кейін оны “айыппұл” деген сөзбен ауыстыруға болады.

```
s = 'Мүлік салығын есептеу және төлеу кезінде бұзушылықтар үшін салық төлеушілерге санкциялар қолданылады'
re.sub('санкциялар', 'айыппұл', s)
```

НӘТИЖЕ: “Мүлік салығын есептеу және төлеу кезінде бұзушылықтар үшін төлеушілерге айыппұл салынады”

Мұнда негізгі тұрақты өрнек әдістерінің жұмыс істеу әдісінің қарапайым сипаттамасы берілген. Әмбебап үлгілерді көбірек жазу үшін, қай сөзді іздегеніңізді нақты білмесеңіз, сондай-ақ қай жағдайда іздеу керектігін білмесеңіз қолдануға болады.

Орыс тіліндегі мәтіндер үшін “Наташа” сияқты дайын кітапханалар бар. Оның көмегімен орыс тіліндегі мәтіндерден атауларды шығарып алуға болады. Бірақ егер мұндай кітапхана болмаса, тұрақты өрнектерді қолданбасқа болмас еді. Мысалы құжатта:

“Қазақстан Республикасының Президенті Қ.Қ. Тоқаев 20.03.2019 – бүгінге дейін.”- деген жол бар дерлік. Ал бізге “Тоқаев Қ.Қ.”-деген тіркесті табу керек болсын.

Алдымен сөз іздейтін үлгіні жазу керек, ол үшін сөз, одан кейін бос орын, сосын әріп, нүкте, әріп, нүкте іздейтіндей болуы керек. Бос орындардан басқа таңбаларды іздеу үшін “\w+” операторын пайдалануға болады. Осылайша жолдан бос орындармен бөлінген барлық сөздер табылады.

Әрі қарай, сөзден кейін бос орын екенін көрсету қажет. Ол үшін “\s” операторын қолданады. Сонда сөзден кейін бос орыны бар сөздер ғана қайтарылатын болады. Біздің мәселенің аты-жөні жоғалып кетеді.

Енді бос орыннан кейін бір әріп бар екенін көрсету керек. Ол үшін біз орыс әліпбиінің барлық әріптерін, үлкен және кіші әріптерін іздегеніміз тік жақшада жазылады. Ал оның жанында ирек жақшада осы әріптердің қаншасы болуы керектігі көрсетіледі.

Әрі қарай "\" көмегімен жол болуы керек деп жазылады. "\" бұл жерде нүктені іздегенімізді ерекше атап өту үшін қолданылады, себебі нүктенің өзі де оператор болып табылады және жолдың үзілуінен басқа кез келген таңбаны білдіреді. Әріп пен нүктені іздейтін құрылыс екі рет жазылуы тиіс, өйткені мақсаты жалпы аты-жөнін алу болып табылады. Сондай-ақ сақтандыру жасап, аты-жөндер арасына бос орын қоюға болатынын ескеруге болады. Ол үшін олардың арасына "\s*" операторы кірістіріледі. Сол жақтағы құрылыс қатарда болуы да, болмауы да мүмкін екені көрсетіледі.

Сонымен қатар, "\w+[А-ЯЁа-яё]+"-ны жаза отырып, тегінің орнына мысалы сан болып қалуы мүмкін, бірақ одан кейін бос орын, әріп, нүкте, әріп, нүкте болатын жағдайлардан қорғау үшін жазуға болады.

```
s = ‘Қазақстан Республикасының Президенті Тоқаев Қ.Қ. 20.03.2019 – бүгінге дейін.’
pattern = r"[А-ЯЁа-яё]+\s[А-ЯЁа-яё]{1}\.\s*[А-ЯЁа-яё]{1}\."
name = re.findall(pattern, s)
name
НӘТИЖЕ: "Тоқаев Қ.Қ. "
```

Салық бойынша мәтіндік құжатпен жұмыс істеген кезде онда қандай объектілерге салық салынатынын табуға тырысып көрейік. Құжатта “Мүлік салығын төлеуді кешіктіргені үшін төлеуші әрбір мерзімі өткен күн үшін төленбеген соманың 0,2 пайызы мөлшерінде айыппұл төлейді” деген жол бар. “0,2 пайыз” тіркесі - біз тапқымыз келетін нысан. Сондықтан біздің өрнегіміз “соманың” деген сөзден басталатынын көрсетеміз, сосын “.+” операторымен кейбір таңбалар арасында болатындығын, ал олардан кейін “пайыз” сөзі бар екенін көрсетеміз. Сөздің соңындағы жұрнақ-жалғауларының өзгеру мүмкіндігіне байланысты “процент” сөзінің соңы кез-келген түрде бола алатындығы көрсететін құрылым қосу керек, өйткені келесі сөз “пайызы” болуы мүмкін. Мұндай үлгімен сізге қажетті нәрсені бірден табуға болады, бірақ оны әмбебап ету қажет. Мысалы, айыппұл төленбеген соманың жалпы пайызымен ғана емес, әрбір күннен пайыз алынып отыратындығын хабарлайтын жағдайды да қамтамасыз ету керек. Мысалы, “Әрбір мерзімі өткен күн үшін төленбеген соманың 0,2 пайызы”. Сондықтан “әрбір мерзімі өткен күн үшін төленбеген соманың” деген сөз тіркесі “*” көмегімен өрнектелуі керек ол мәтінде кездесуі немесе болмауы мүмкін дегенді білдіреді.

```
def find_taxes_term(text):
return re.sub('соманың', '', re.search(r"әрбір мерзімі өткен күн үшін
төленбеген*\s(.+)\s*((.+)*соманың\w+)*", text)[0]).strip()
s = ‘Мүлік салығын төлеуді кешіктіргені үшін төлеуші әрбір мерзімі өткен күн үшін
төленбеген соманың 0,2 пайызы мөлшерінде айыппұл төлейді’
find_taxes_term(s)
НӘТИЖЕ : “мерзімі өткен әрбір күн үшін төленбеген соманың 0,2 пайызы”
```

Қорытынды

Кез келген басқа код сияқты, бұл әдістің кемшіліктері мен проблемалары бар. Тұрақты өрнек программист күткен нәрсені таба алмауы мүмкін. Немесе қосымша нәрсе тауып беруі мүмкін, әсіресе, егер тұрақты тіркестер тізбегі болса. Сондықтан, тұрақты өрнек жазған кезде оны міндетті түрде тексеру керек.

Пайдаланған әдебиеттер тізімі

1. Фридл, Дж. Регулярные выражения = Mastering Regular Expressions. — СПб.: “Питер”, 2001. — 352 с. — (Библиотека программиста). — ISBN 5-318-00056-8.
2. Смит, Билл. Методы и алгоритмы вычислений на строках (regexp) = Computing Patterns in Strings. — М.: “Вильямс”, 2006. — 496 с. — ISBN 0-201-39839-7.