Дәріс 13

Тақырыбы: Жолдар

Жоспар:

1. Жолдарды енгізу және шығару

3. Жолдарға қолданылатын функциялар

Оқытудың әдістемесі мен формасы: Баяндау, дәріс

Сабақ мақсаты: жолды өңдеуге мүмкіндік беретін символдық айнымалыларды қолдану ерекшеліктерімен, функцияларымен танысу.

Әдебиеттер:[4,6]

Деректердің жолдық типі, апостроф ‘ ‘ немесе қос тырнақша “ “ жолды бейнелейді.

Біз деректерді жолдарды пайдаланып оқығанды жөн көреміз, содан кейін қажет болған жағдайда деректерді талдап, түрлендіреміз.

Осылайша, біз қателерді және/немесе пайдаланушының деректерді дұрыс енгізбеуін көбірек басқарамыз.

Енгізілген сандар жол түрінен сандық түрге ауыстырылуы керек.

0	**1**	**2**	**3**	**4**
S	**a**	**l**	**e**	**m**

File Edit Format Ru

```
s1 = 'Salem'

print(s1[0])
```

Жолды толық шығару:

File Edit Format Run Options Window Help

```
s1 = 'Salem'
print(s1)
s2 = 'Kal kalai?'
print(s2)
s3= "Abzats bitty. Azat shol"
print(s3)
```

Жолдарды қосуға болады. Бұл операция жолды біріктіру − деп аталады «конкатенация».

```
File  Edit  Format  Run  Options  Window  Help
s1 = 'Salem'
print(s1)
s2 = 'Kal kalai?'
print(s2)
s3= "Abzats bitty. Azat shol"
print(s3)
```

Сондай-ақ, жолдарды санға көбейтуге болады.

```
File  Edit  Format  Run  Options
s1 = 'Salem'

print(('-'+s1+'!\n')*2)
```

```
==================
-Salem!
-Salem!
```

Жолмен жұмыс жасауға арналған фунцкиялар:

`str.`**`replace`**(*old*, *new*[, *count*])
Return a copy of the string with all occurrences of substring *old* replaced by *new*. If the optional argument *count* is given, only the first *count* occurrences are replaced.

`str.`**`rfind`**(*sub*[, *start*[, *end*]])
Return the highest index in the string where substring *sub* is found, such that *sub* is contained within `s[start:end]`. Optional arguments *start* and *end* are interpreted as in slice notation. Return `-1` on failure.

`str.`**`rindex`**(*sub*[, *start*[, *end*]])
Like `rfind()` but raises `ValueError` when the substring *sub* is not found.

`str.`**`rjust`**(*width*[, *fillchar*])
Return the string right justified in a string of length *width*. Padding is done using the specified *fillchar* (default is an ASCII space). The original string is returned if *width* is less than or equal to `len(s)`.

`str.`**`rpartition`**(*sep*)
Split the string at the last occurrence of *sep*, and return a 3-tuple containing the part before the separator, the separator itself, and the part after the separator. If the separator is not found, return a 3-tuple containing two empty strings, followed by the string itself.

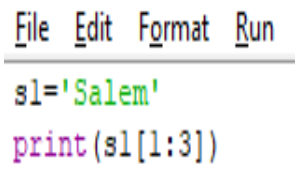
`str.`**`rsplit`**(*sep=None*, *maxsplit=-1*)
Return a list of the words in the string, using *sep* as the delimiter string. If *maxsplit* is given, at most *maxsplit* splits are done, the *rightmost* ones. If *sep* is not specified or `None`, any whitespace string is a separator. Except for splitting from the right, `rsplit()` behaves like `split()` which is described in detail below.

Қиып алу

Сондай-ақ, қос нүкте операторын пайдаланып кез келген үздіксіз жол бөлігін қарауға болады :
Екінші Сан кесудің соңын көрсетеді, бірақ оған кірмейді. Кесу келесідей оқылады: "бастап және дейін, бірақ қосылмайды"
Егер екінші Сан жолдың ұзындығынан асып кетсе, кесінді жолдың соңына дейін алынады

```
>>> s = 'Monty Python'
>>> print(s[0:4])
Mont
```

```
>>> print(s[6:7])
P
>>> print(s[6:20])
Python
```



In кілт сөзін бір жолдың екіншісінің ішінде бар жоғын тексеру үшін де қолдануға болады
In бар өрнек-шын (шын) немесе жалған (жалған) мәнін қайтаратын логикалық өрнек. If операторымен өрнектерде қолдануға болады.

```
>>> fruit = 'banana'
>>> 'n' in fruit
True
>>> 'm' in fruit
False
>>> 'nan' in fruit
True
>>> if 'a' in fruit :
...     print('Найдено!')
...
Найдено!
>>>
```