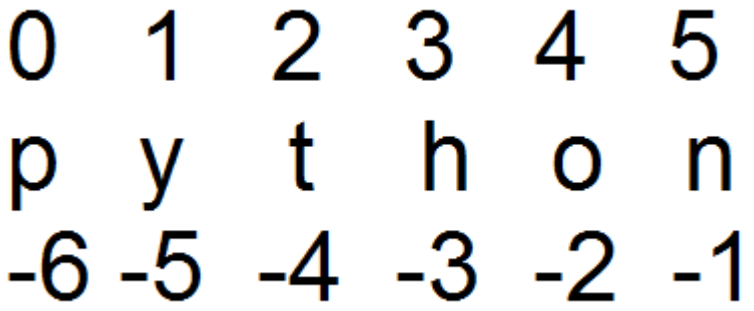
8 ЖОЛДАРМЕН ЖҰМЫС ІСТЕУ

8.1 Негізгі анықтамалар

Айта кету керек, біз бұрын Python бағдарламалау тілінде жолдармен жұмыс жасаудың алғашқы дағдыларын алдық. Деректерді енгізуге шақыру немесе бағдарламада жауап беру, сондай-ақ жолды аудару, кесте, жолды пішімдеу сияқты сұрақтар қарастырылады. Бұл тақырыпта біз жолдармен жұмыс істеуге арналған функциялар мен әдістермен танысамыз.
Python-дағы жол-бұл **str** класының нысаны. Бұрын тақырыптарда сандық деректермен жұмыс істеу үшін типтерді келтіру функциялары қолданылды, мысалы, **int**, бұл келесі мысалда көрсетілгендей:

```
ind=int(input(" \n Введите индекс элемента "))
```

Суретте жолдар нөлден индекстелетіні көрсетілген, яғни егер **stroke="python"** операторы болса, онда жолдағы бірінші таңба нөлге тең болады және оған сілтеме **stroka[0]** ретінде жазылады. Сонымен қатар, Python-дағы жол элементтеріне, теріс индекстерді көрсету арқылы қол жеткізуге болады, мысалы, **print(stroka[-6])** операторы **p** таңбасын экранға шығарады.



Сурет 103 – Жолдарды индекстеу

Бағдарламалау кезінде бір таңбаны екінші таңбамен салыстырмас бұрын оны 128 әріптік-цифрлық таңбаларды кодтауды қолдайтын ASCII (American Standard Code for Information Interchange ақпарат алмасудың американдық стандарттық коды) кестесі арқылы санға айналдыру қажет екені белгілі.
Нөлден басталатын базалық кестенің алғашқы 32 коды аппараттық жасаушыларға (ең алдымен, компьютерлер мен баспа құрылғыларын өндірушілерге) беріледі. Бұл аймақта кез-келген тілдік таңбаларға сәйкес келмейтін басқару кодтары деп аталады және сәйкесінше бұл кодтар экранда да, басып шығару құрылғысында да көрсетілмейді, бірақ біз тізімге енгізетін басқару функцияларын кестеде көрсетеміз.

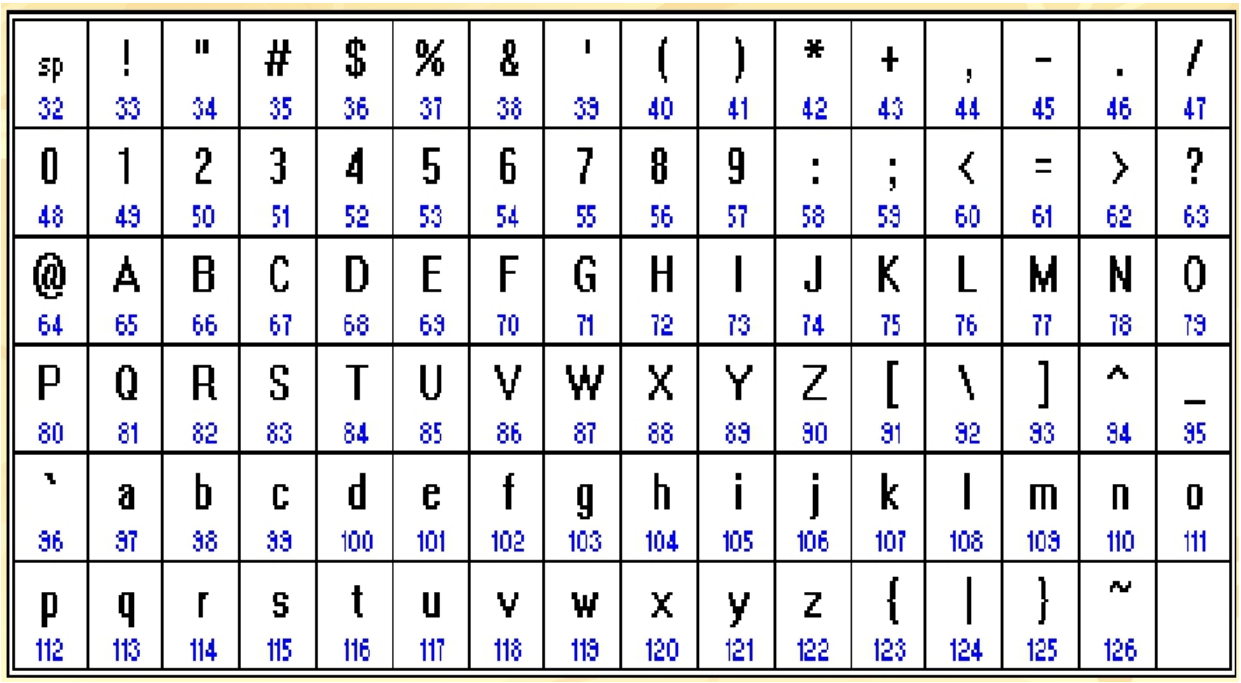
Кесте 7 - Нөлден басталатын негізгі кестенің алғашқы 32 коды

Символ	Код	Клавиши	Значение

nul	0	Ctrl + @	Нуль
soh	1	Ctrl + A	Начало заголовка
stx	2	Ctrl + B	Начало текста
etx	3	Ctrl + C	Конец текста
eot	4	Ctrl + D	Конец передачи
enq	5	Ctrl + E	Запрос
ack	6	Ctrl + F	Подтверждение
bel	7	Ctrl + G	Сигнал (звонок)
bs	8	Ctrl + H	Забой (шаг назад)
ht	9	Ctrl + I	Горизонтальная табуляция
lf	10	Ctrl + J	Перевод строки
vt	11	Ctrl + K	Вертикальная табуляция
ff	12	Ctrl + L	Новая страница
cr	13	Ctrl + M	Возврат каретки
so	14	Ctrl + N	Выключить сдвиг
si	15	Ctrl + O	Включить сдвиг
dle	16	Ctrl + P	Ключ связи данных
dc1	17	Ctrl + Q	Управление устройством 1
dc2	18	Ctrl + R	Управление устройством 2
dc3	19	Ctrl + S	Управление устройством 3
dc4	20	Ctrl + T	Управление устройством 4
nak	21	Ctrl + U	Отрицательное подтверждение
syn	22	Ctrl + V	Синхронизация
etb	23	Ctrl + W	Конец передаваемого блока
can	24	Ctrl + X	Отказ
em	25	Ctrl + Y	Конец среды
sub	26	Ctrl + Z	Замена
esc	27	Ctrl + [	Ключ
fs	28	Ctrl + \	Разделитель файлов
gs	29	Ctrl +]	Разделитель группы
rs	30	Ctrl + ^	Разделитель записей
us	31	Ctrl + _	Разделитель модулей

Кестелерді кодтаудың ұлттық стандарттарына кодтық кестенің халықаралық бөлігі өзгеріссіз енеді, ал екінші жартысында ұлттық алфавиттердің кодтары, жалған графикалық белгілер және кейбір математикалық белгілер бар.
32 кодтан 127 кодқа дейін ағылшын алфавитінің таңбаларының кодтары, тыныс белгілері, сандар, арифметикалық амалдар және кейбір қосалқы белгілер орналастырылған. Негізгі ASCII кодтау кестесі 8-ші кестеде көрсетілген.

Кесте 8 - Негізгі ASCII кодтау кестесі



sp 32	! 33	" 34	# 35	$ 36	% 37	& 38	' 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47
0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63
@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79
P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	\ 92	] 93	^ 94	_ 95
` 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111
p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	{ 123	| 124	} 125	~ 126	

Сонымен, жолдармен жұмыс жасау кезінде **"A"** символы **"a"** символымен бірдей емес екенін білу керек. Жолдың ұзындығы тек компьютердің жедел жадының көлемімен шектеледі және жол элементіне жүгіну үшін оның индексін тік жақшада көрсету жеткілікті. Сол сияқты, тізім элементіне жүгіну бұрын қарастырылған.
Python-дағы жолдар өзгермейтін дәйектілік болып табылады және **for** операторымен циклды қолдана отырып өңделеді. Индекске жүгіну арқылы жол таңбасын өзгерту мүмкін емес екенін түсіну маңызды.

```
stroka="Python"
stroka[0]="p"  #недопустимый оператор
```

Жолды өңдеу кезінде оған төмендегі тізімде көрсетілгендей тікелей циклде жүгінуге болады.

```
stroka="python"
for i in stroka:
    print(i, end=" ")
```

Бұл кодтың нәтижесі **p y t h o n** сөзін экранға шығару болады.

8.2 Символдармен жұмыс істеуге арналған функциялар

Жолдармен жұмыс істеуге арналған негізгі функцияларды қарастырайық.
1. ***Len () функциясы.***
Жолдармен жұмыс істеу кезінде **Len()** функциясы пайдалы болуы мүмкін, оның мақсаты жолдың ұзындығын анықтау болып табылады. **Print** операторында біз жолдың әр элементіне сілтеме жасаймыз, жолдың атын

атаймыз және әр элементтің индексін тік жақшаға аламыз. Әрине, төмендегі кодтың нәтижесі **P y t h o n** сөзін экранға шығару болады.

```
stroka="Python"
for i in range(len(stroka)):
   print(stroka[i], end=" ")
```

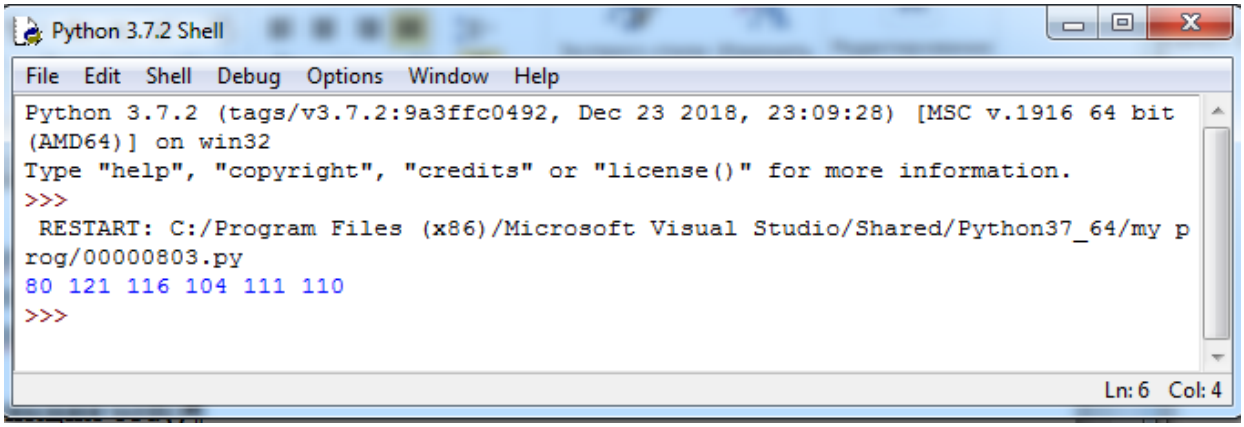
```
P y t h o n
```

2. Ord() функциясы

Тағы бір функция - **ord(),** оның синтаксисі **ord("таңба")** көрсетілген таңбаның кодын қайтарады. Мысалы, төмендегі мысалда жолдың әр символына қолданған **ord()** функциясы **"Python"** жолының ASCII таңбалардың кодын қайтарады.

```
stroka="Python"
for i in range(len(stroka)):
   print(ord(stroka[i]), end=" ")
```

Бағдарлама жұмысының нәтижесі 104-ші суретте көрсетілген.



Сурет 104 – Python сөзінің ASCII таңбалық кодтары шығарылды

3.Chr() функциясы

Chr(Сан) синтаксисі болатын функцияның әрекеті **ord()** функциясына тікелей қарама-қарсы, атап айтқанда таңбаның ASCII коды бойынша таңбаны қайтару. Осылайша, **ord ()** функциясына қолданылған **chr()** функциясы, ол өз кезегінде ASCII таңбалық кодтарын қайтарады, төмендегі көрсетілген кодтағы **"Python"** жолының бастапқы түріне қайтарады.

```
stroka="Python"
for i in range(len(stroka)):
   print(chr(ord(stroka[i])), end=" ")
```

```
P y t h o n
```

8.3 Жолдармен жұмыс істеу әдістері

Жолдармен жұмыс істеуге арналған негізгі әдістерді қарастырайық.

1. Upper() әдісі .

Stroka.upper() әдіс синтаксисі. Жолдың барлық таңбаларын жоғарғы регистрге түрлендіреді. Мысалы,

```
stroka="PythOn"
newstr=stroka.upper()
print(newstr)
```

```
PYTHON
```

2. Lower() әдісі.

Stroka.lower() әдіс синтаксисі.. Жолдың барлық таңбаларын төменгі регистрге түрлендіреді. Мысалы,

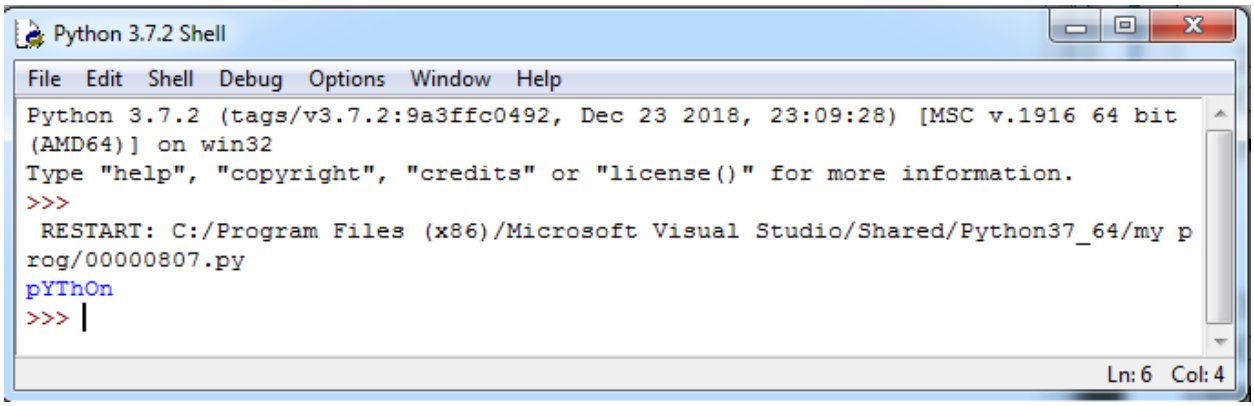
```
stroka="PytHon"
newstr=stroka.lower()
print(newstr)
```

```
python
```

3. Swapcase()әдісі .

Stroka. swapcase() әдіс синтаксисі. 105 – ші суретте көрсетілгендей, кіші әріптермен жазылған жолдың барлық таңбаларын жоғарғы және керісінше түрлендіреді. Мысалы,

```
stroka="PytHoN"
newstr=stroka.swapcase()
print(newstr)
```

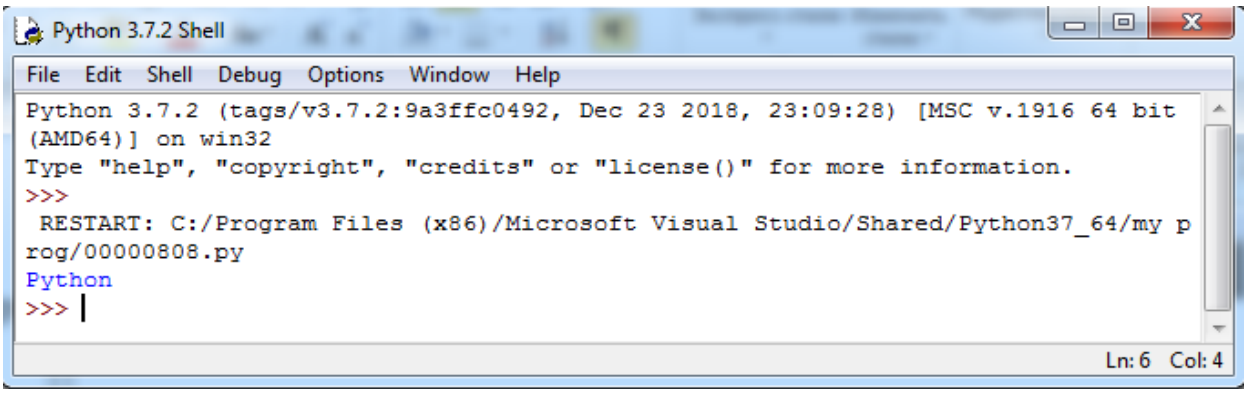


Сурет 105 – Swapcase() әдісінің жұмысы

4. Capitalize()әдісі .

Stroka.capitalize() әдіс синтаксисі. Жолдағы бірінші әріпті жоғарғы регистрге, ал қалғанын төменгі регистрге түрлендіреді. Мысалы,

```
stroka="pyTHoN"
newstr=stroka.capitalize()
print(newstr)
```

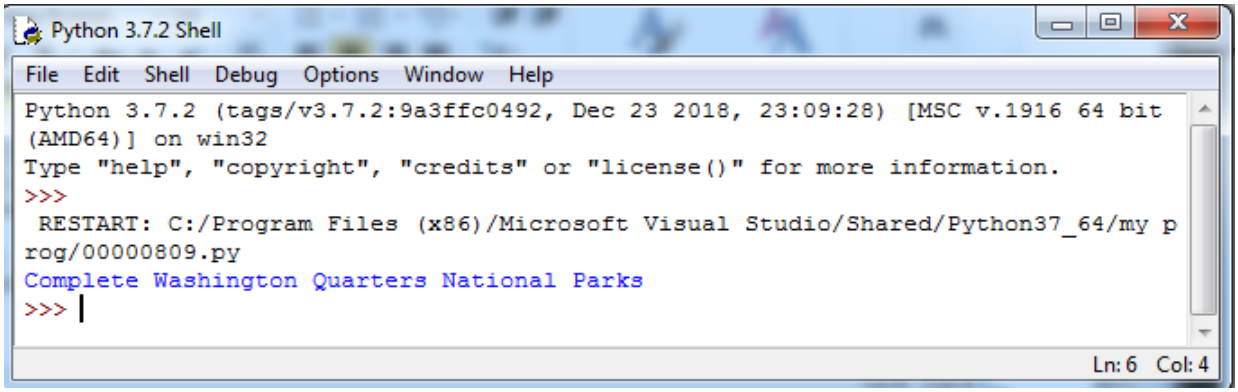


Сурет 106 –Capitalize() әдісінің жұмысы

5. Title() әдici .

Stroka.title() әдіс синтаксисы. 107-ші суретте көрсетілгендей жолдағы барлық бірінші әріптерді жоғарғы регистрге, ал қалғандары төменгі регистрге түрлендіреді. Мысалы,

```
stroka="complete washingtoN quarTers national paRks"
newstr=stroka.title()
print(newstr)
```



Сурет 107 –Title() әдісінің жұмысы

6. Startswith() әдici .

Stroka.startswith(podstroka) әдіс синтаксисі. **Stroka** жолының **podstroka**(жол үзіндісінен) көрсетілген ішкі жолдан басталатынын тексереді. Мысалы, Бұл мысалда "толық washingtoN quarTers national paRks" жолының басында "com pl" ішкі жолының болу шарты тексеріледі. Бұл кодтың нәтижесі **true** мәні болады.

```
stroka="complete washingtoN quarTers national paRks"
podstr="compl"
n=stroka.startswith(podstr)
print(n)
```

7. Endswith() әдici .

Әдіс синтаксисі: **stroka.endswith(podstroka). podstroka** ішкі жолымен көрсетілген **stroka** жолының аяқталғанын тексереді. Мысалы, "complete washingtoN quarTers national paRks " жолының соңында " rks " ішкі жолының

болу шарты тексеріледі. Бұл кодтың нәтижесі **False**(жалған) болады. Бұл мәтінді регистрге енгізудің сезімталдығын тағы бір рет көрсетеді, өйткені біздің жолымыз "rks" емес "Rks "ретінде аяқталады.

```
stroka="complete washingtoN quarTers national paRks"
podstr="rks"
n=stroka.endswith(podstr)
print(n)
```

8. Replace()әдісі .

Әдіс синтаксисі: **stroka.replace(old, new)**, мұнда **old** - ауыстыру үшін ішкі жол, **new** - жаңа ішкі жол. Әдіс **stroka** жолында **old** ішкі жолын жаңа ішкі жолмен табады және ауыстырады. Мысалы, осы мысалда stroka=stroka.replace("washingtoN quarTers national paRks ","Washington quarters National parks") келесі жолды "Complete Washington quarters National parks" қайтарады

```
stroka="Complete washingtoN quarTers national paRks"
stroka=stroka.replace("washingtoN quarTers national paRks", "Washington quarters National parks")
print(stroka)
```

9. Rfind()әдісі .

Әдіс синтаксисі: **stroke.rfind (podstr),** мұнда **podstr** - ішкі жол. Әдіс ішкі жолдың соңғы позициясының орнын жолға қайтарады. Егер **ішкі жол табылмаса**, онда -1 мәні қайтарылады. **Podstr** параметрінен кейін бастапқы позицияны және ішкі жолды іздейтін жолдағы соңғы позицияларды көрсетуге болады. Бұл параметрлер міндетті емес, егер бастапқы позиция болмаса, іздеу жолдың басынан бастап жүзеге асырылады. Сонымен, **төмендегі мысалда "on**" ішкі жолының **соңғы кірісі 17** позициясында жазылады, өйткені біз іздеуде **8 таңбаны бастапқы позиция ретінде**, ал **соңғыны 30** ретінде көрсеттік. **Егер мысалда бастапқы және соңғы позициялар көрсетілмесе**, онда нәтиже **33** болады. Жол, нөлден индекстеледі, бос орындар мен тыныс белгілері де таңбалар болып табылады.

```
stroka="Complete Washington quarters National parks"
stroka=stroka.rfind("on", 8, 30)
print(stroka)
```

10 Find()әдісі .

Әдіс синтаксисі: **stroke.flnd (podstr),** мұнда **podstr -** ішкі жол. Алдыңғы әдіске қарағанда, **find ()** әдісі жолдың бірінші ішкі жол пзозициясының орнын қайтарады. **Егер ішкі жол табылмаса**, онда -1 мәні қайтарылады. Тиісінше, егер сіз алдыңғы бағдарламаны өзгертсеңіз және **rfind()** әдісінің орнына **find ()** әдісін қолдансаңыз, сонымен қатар іздеудегі бастапқы және соңғы позицияларды алып тастасаңыз, онда " Complete Washington quarters National parks " жолына "**on**" ішкі жолының бірінші енгізуі 17-ші позицияда басталады.

```
stroka="Complete Washington quarters National parks"
stroka=stroka.find("on")
print(stroka)
```

11. Count() әдісі.

Әдіс синтаксисі: **stroke.count(podstr),** мұнда **podstr** - ішкі жол. Бұл әдіс **substr** ішкі жолының пайда болу санын **stroka** жолына қайтарады. Осылайша, төмендегі мысалда **prin**t операторы экранға шығарған жауап **2** саны болады, өйткені "on" ішкі жолы " Complete Washington quarters National parks " жолына екі рет енеді.

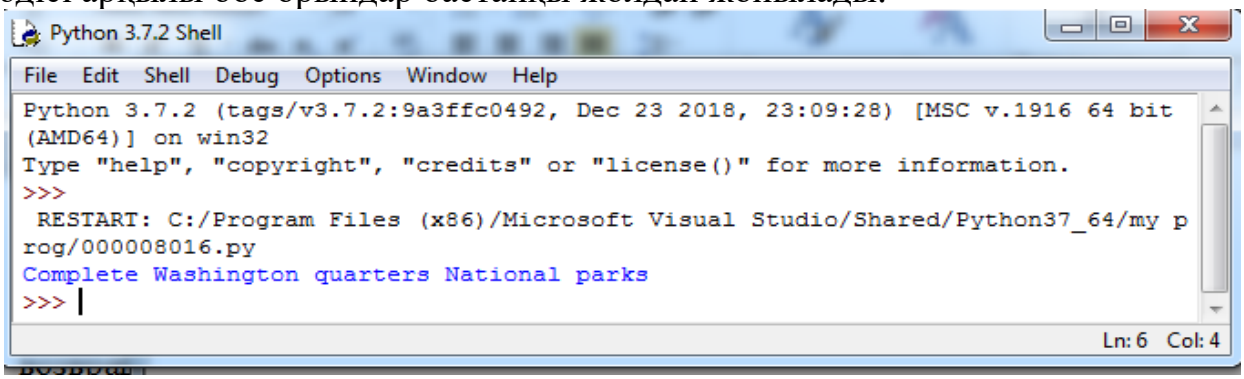
```
stroka="Complete Washington quarters National parks"
stroka=stroka.count("on")
print(stroka)
```

12. Strip() әдісі

Әдіс синтаксисі: **stroka.strip().** Бұл әдіс **stroka** жолындағы бастапқы және соңғы бос орындарды жояды. Төмендегі мысалда бастапқы жолда жолдың басында және соңында бос орындар бар.

```
stroka="      Complete Washington quarters National parks     "
stroka=stroka.strip()
print(stroka)
```

108-ші суретте бағдарламаның орындалу нәтижесі көрсетілген, онда **strip()** әдісі арқылы бос орындар бастапқы жолдан жойылады.



Сурет 108 –strip() әдісінің жұмысы

13. Lstrip() и Rstrip()әдicmepi .

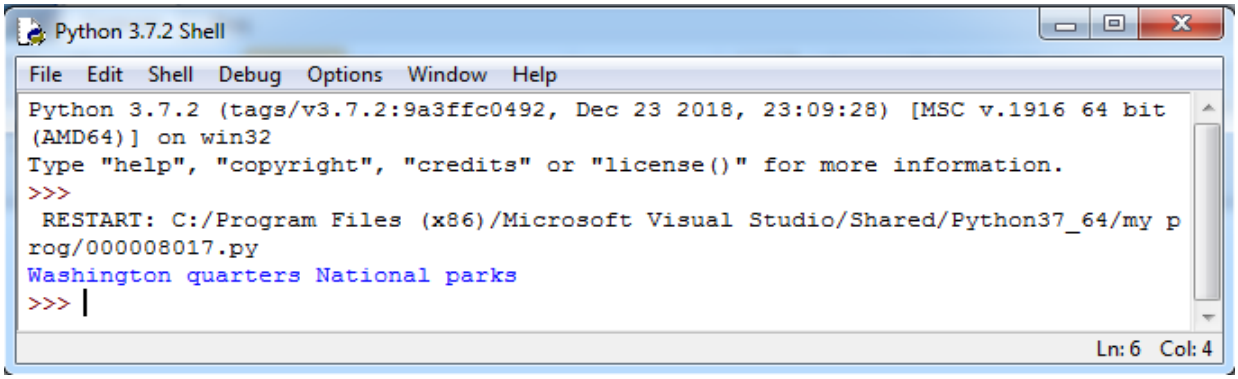
Олардың синтаксисі мен әрекеті қарастырылған **strip()** әдісіне ұқсас, айырмашылығы - strip() әдісі бастапқы жолдың басынан сол жақтағы бос орын таңбаларын жояды, ал **rstrip()** әдісі бастапқы жолдың соңында.

Strip(), strip() және **lstrip()** әдістерін қолдана отырып, бос орындарды ғана емес, жоюға болады. Сонымен, егер әдістердің бірінің параметрі ретінде таңбаны немесе таңбалар тізбегін көрсетсеңіз, онда ол(олар) жойылады. Мысалы, келесі мысалда біз **strip()** әдісіндегі параметр ретінде " Complete"

таңбаларын көрсеттік. Тиісінше, әдіс бастапқы жолды қайтарады, бірақ көрсетілген таңбаларсыз.

```
stroka="Complete Washington quarters National parks     "
stroka=stroka.strip("Complete ")
print(stroka)
```

Ұқсас жағдай 109-суретте көрсетілген.



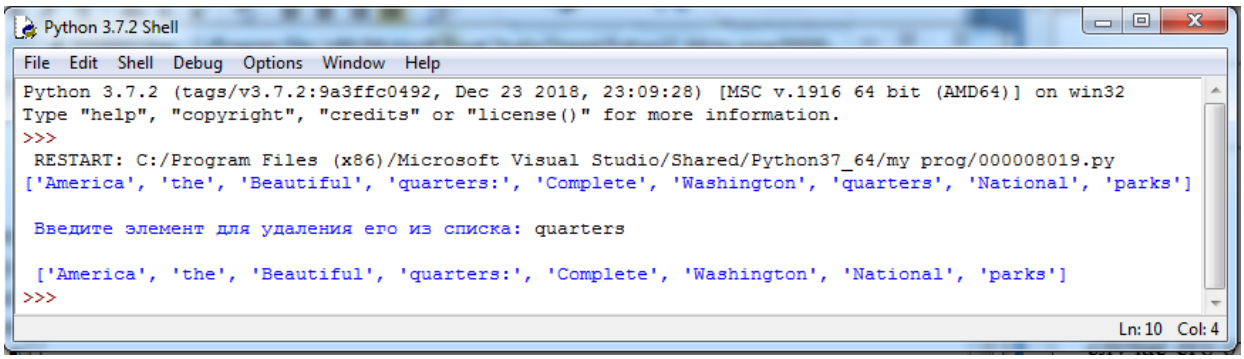
Сурет 108 –strip() әдісінің параметрмен жұмысы

14. Split() әдісі.

Әдіс синтаксисі: **stroka.split().** Әдіс жолды ішкі жолдарға бөледі және оларды тізімге қосады. Егер бөлгіш параметр ретінде болса, онда жол көрсетілген бөлгішке сәйкес ішкі жолдарға бөлінеді. Ол болмаған жағдайда бос орын бөлгіш болып саналады. Төмендегі тізімде көрсетілген кодта **split()** әдісімен бастапқы жол тізімге айналдырылған, бөлгіш бос орын болып табылады. Бұдан әрі тізімнің үстінде тізімнің **remove()** әдісі орындалады, ол пайдаланушы пернетақтадан енгізген тізім элементін жояды.

```
stroka="America the Beautiful quarters: Complete Washington quarters National parks"
spisok=stroka.split(" ")
print(spisok)
element=input("\n Введите элемент для удаления его из списка: ")
for i in spisok:
    if element in spisok:
        spisok.remove(element)
print("\n",spisok)
```

Бағдарлама жұмысының нәтижесі 109-суретте көрсетілген. Айта кету керек, тізімді құру кезінде бөлгіш бос орын болды. Тиісінше, бастапқы тізімде **"quarters"** тізімінің элементі оған **"quarters:"**тізімінің элементі сияқты бір рет кіреді. Жойылатын элементті сұраған кезде біз **"quarters"** енгіздік, сондықтан **"quarters:"** тізімінің элементі тізімде қалды және тек **"quarters"** тізімінен жойылды.



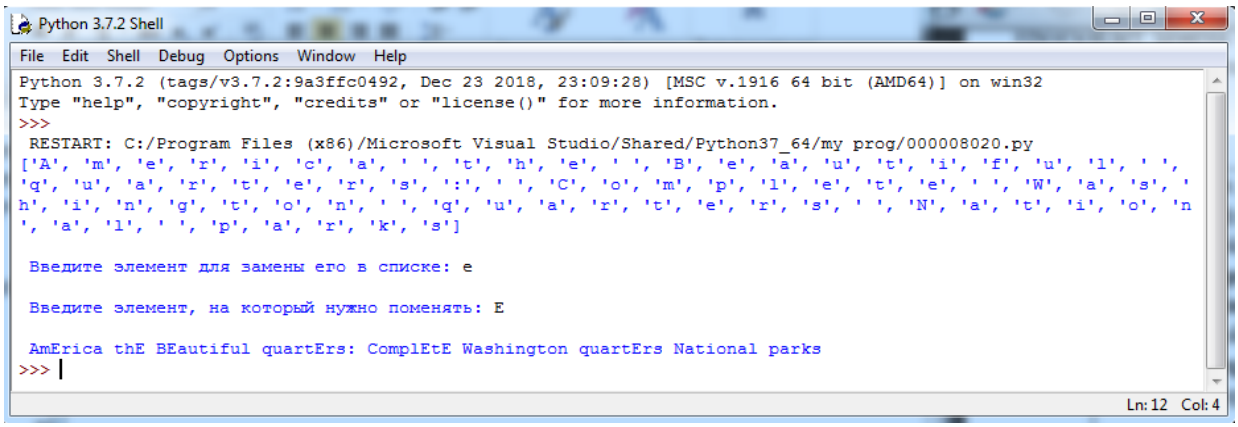
Сурет 109 – Тізімнен **"quarters"** элементін жою үшін **Split()** әдісінің тізімдік **remove()** әдісімен жұмысы

15. Join()әдісі .

Әдіс синтаксисі: **бөлгіш.join (spisok),** онда **spisok** - тізім немесе кортеж. Жолды тізімге түрлендірудің тағы бір тәсілі-**list()** функциясын қолдану, ал кері түрлендіру **]join()** әдісімен жүзеге асырылады. Мұндай түрлендірулер жол элементін индекс бойынша өзгерту үшін қажет, өйткені жолдар бастапқыда өзгермейді.

Төмендегі мысалда **list()** функциясын қолдана отырып, бастапқы жолдан тізім жасалады. Бастапқы тізім элементін пернетақтадан енгізілген элементпен ауыстыруға тырысайық және ол үшін **input()** функциясын қолдана отырып, тиісті сұраныстарды орындаймыз. Біз **len()** функциясын қолдана отырып, тізімнің ұзындығын есептейміз және **for** операторының көмегімен тізімді қарауды ұйымдастырамыз. Егер тізімнің келесі элементі енгізілген элементке тең болса, онда элемент тізімінің бастапқы **element**-ті ауыстырылады **element1**-ге. Содан кейін, **join()** әдісін қолдана отырып, тізімді жолға айналдырып, оны экранға шығарыңыз (110-сурет).

```
stroka="America the Beautiful quarters: Complete Washington quarters National parks"
spisok=list(stroka)
print(spisok)
element=input("\n Введите элемент для замены его в списке: ")
element1=input("\n Введите элемент, на который нужно поменять: ")
n=len(spisok)
for i in range(0, n):
    if spisok[i]==element:
        spisok[i]=element1
stroka1="".join(spisok)
print("\n",stroka1)
```



Сурет 110 – "e" тізімінің элементін "E" - ге ауыстыру