5196CH08

باب-8

اسٹرنگ
(STRINGS)

> ’’کمپیوٹر نوٹ بک کے بارے میں بڑی بات
> یہ ہے کہ آپ اس میں کتنا ہی مواد ٹھونس دیں نہ
> یہ بڑا ہوگا اور نہ بھاری‘‘
> —بل گیٹس
> (Bill Gates)

8.1 تعارف

ہم باب 5 میں پڑھ چکے ہیں کہ تسلسل (Sequence) آئٹموں کا مرتب مجموعہ ہے اور ہر آئٹم کی انڈیکسنگ ایک صحیح عدد کے ذریعے ہوتی ہے۔ پائتھن میں استعمال ہونے والے مندرجہ ذیل تسلسل ڈیٹا ٹائپ بھی باب 5 میں مختصراً متعارف کرائے گئے ہیں۔

- اسٹرنگ (Strings)
- لسٹ (List)
- ٹپل (Tuple)

باب 5 میں ایک اور ڈیٹا ٹائپ بھی متعارف کرایا گیا ہے جسے ’ڈکشنری‘ کہتے ہیں جو میپنگ کے زمرے کے تحت آتی ہے۔ اس باب میں ہم اسٹرنگ کا تفصیلی مطالعہ کریں گے۔ لسٹ کا احاطہ باب 9 میں کیا جائے گا جب کہ ٹپل اور ڈکشنری باب 10 میں زیر بحث رہیں گے۔

8.2 اسٹرنگ (STRINGS)

اسٹرنگ ایک ایسا تسلسل ہے جو ایک یا ایک سے زیادہ یونیکوڈ (UNICODE) کیریکٹر پر مشتمل ہے۔ یہاں کیریکٹر کوئی حرف، ہندسہ، وہائٹ اسپیس (Whitespace) یا کوئی علامت ہوسکتا ہے۔ اسٹرنگ کی تشکیل ایک یا ایک سے زیادہ کیریکٹر کو اکہرے، دوہرے یا تہرے واوین کے اندر لکھ کر کی جاتی ہے۔

مثال 8.1

```
>>> str1 = 'Hello World!'
>>> str2 = "Hello World!"
>>> str3 = """Hello World!"""
>>> str4 = '''Hello World!''
```

str1, str2, str3, str4 سبھی اسٹرنگ متغیرات ہیں جن کی قدر (ویلیو) ایک ہی ہے یعنی 'Hello World!'۔ str3 اور str4 میں اسٹور کی گئی قدروں کو تہرے واوین کا استعمال کرکے متعدد سطروں تک وسعت دی جاسکتی ہے۔

```
>>> str3 = """Hello World!
        welcome to the world of Python"""
>>> str4 = '''Hello World!
        welcome to the world of Python'''
```

اس باب میں

- « اسٹرنگ کا تعارف
- « اسٹرنگ آپریشن
- « اسٹرنگ کا اعادہ
- « اسٹرنگ میتھڈ اور بلٹ ان فنکشن
- « اسٹرنگ کو استعمال کرنا

پائتھن میں کیریکٹر ڈیٹا ٹائپ نہیں ہوتا۔ لمبائی ایک والی اسٹرنگ کو کیریکٹر تصور کیا جاتا ہے۔

8.2.1 اسٹرنگ میں کیریکٹر تک رسائی حاصل کرنا

اسٹرنگ (string) کے ہر ایک انفرادی کیریکٹر تک ایک تکنیک کی مدد سے رسائی حاصل کی جاسکتی ہے۔ اس تکنیک کو انڈیکسنگ (Indexing) کہتے ہیں۔ انڈیکس اس کیریکٹر کا تعین کرتا ہے جس تک رسائی مطلوب ہے اور اسے بڑے بریکٹ ([]) میں لکھا جاتا ہے۔ اسٹرنگ میں پہلے کیریکٹر (بائیں طرف سے) کا انڈیکس 0 اور آخری کیریکٹر کا انڈیکس n-1 ہوتا ہے جہاں n اسٹرنگ کی لمبائی ہے۔ اگر ہم انڈیکس کی قدر اس رینج سے تجاوز کرجاتی ہے تو IndexError ظاہر ہوجاتی ہے۔ انڈیکس ایک صحیح عدد (مثبت، صفر یا منفی) ہونا چاہیے۔

```
#initializes a string str1
>>> str1 = 'Hello World!'
#gives the first character of str1
>>> str1[0]
 'H'
#gives seventh character of str1
>>> str1[6]
'W'
#gives last character of str1
>>> str1[11]
'!'
#gives error as index is out of range
>>> str1[15]
IndexError: string index out of range
```

انڈیکس ایک عبارت بھی ہوسکتا ہے جس میں متغیرات (Variable) اور آپریٹر شامل ہوسکتے ہیں لیکن عبارت کا حاصل ایک صحیح عدد ہونا چاہیے۔

```
#an expression resulting in an integer index
#so gives 6th character of str1
>>> str1[2+4]
 'W'
#gives error as index must be an integer
>>> str1[1.5]
TypeError: string indices must be integers
```

پائتھن میں انڈیکس کی قدر منفی بھی ہوسکتی ہے۔ منفی انڈیکس کا استعمال اس صورت میں کیا جاتا ہے جب ہمیں اسٹرنگ کے کیریکٹر کو دائیں سے بائیں طرف ایکسس کرنا ہو۔ دائیں طرف سے شروع کرتے ہوئے پہلے کیریکٹر کا انڈیکس 1- اور آخری کیریکٹر کا n- ہوتا ہے جہاں n اسٹرنگ کی لمبائی ہے۔ جدول 8.1 میں اسٹرنگ 'Hello World!' میں کیریکٹر کی انڈیکسنگ (منفی اور مثبت دونوں) کو دکھایا گیا ہے۔

```
>>> str1[-1] #gives first character from right
'!'
>>> str1[-12]#gives last character from right
'H'
```

جدول 8.1 اسٹرنگ ‘Hello World!’ میں کیریکٹر کی انڈیکسنگ

مثبت انڈیکس	0	1	2	3	4	5	6	7	8	9	10	11
اسٹرنگ	H	e	l	l	o		W	o	r	l	d	!
منفی انڈیکس	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

پائتھن میں ان بلٹ فنکشن len() پیرامیٹر کے طور پر پاس کی گئی اسٹرنگ کی لمبائی کو ظاہر کرتا ہے۔

مثال کے طور پر اسٹرنگ ‘Hello World!’ = str1 کی لمبائی 12 ہے۔

```
#gives the length of the string str1
>>> len(str1)
12
#length of the string is assigned to n
>>> n = len(str1)
>>> print(n)
12
#gives the last character of the string
>>> str1[n-1]
'!'
#gives the first character of the string
>>> str1[-n]
'H'
```

8.2.2 اسٹرنگ نا قابل تغیر (String is Immutable) ہوتی ہے

اسٹرنگ ایک نا قابل تغیر ڈیٹا ٹائپ ہے۔ اس کا مطلب یہ ہے کہ اسٹرنگ کی تشکیل کے بعد اس کے مواد کو تبدیل نہیں کیا جاسکتا ہے۔ ایسا کرنے کی کوشش کا نتیجہ غلطی ظاہر ہونے کی صورت میں برآمد ہوگا۔

```
>>> str1 = "Hello World!"
#if we try to replace character 'e' with 'a'
>>> str1[1] = 'a'
TypeError: 'str' object does not support item
assignment
```

8.3 اسٹرنگ پر انجام دیے جانے والے عمل (STRING OPERATIONS)

جیسا کہ ہم جانتے ہیں ایک اسٹرنگ کیریکٹر کا تسلسل ہے۔ پائتھن میں اسٹرنگ ڈیٹا ٹائپ پر مختلف عملوں (Operations) کا اطلاق کیا جاسکتا ہے مثلاً کنکیٹینیشن، تکرار، رکنیت (ممبر شپ) اور سلائسنگ۔ ذیل میں ان عملوں کی وضاحت مناسب مثالوں کے ساتھ کی گئی ہے۔

8 3 1 کنکیٹنیشن (Concatenation)

کنکیٹنیشن کا مطلب ہے ’جوڑنا‘۔ پائتھن میں ہم کنکیٹنیشن آپریٹر (جمع کا نشان) کا استعمال کرکے دو اسٹرنگ کو یکجا کر سکتے ہیں، اسے + کی علامت سے ظاہر کیا جاتا ہے۔

```
>>> str1 = 'Hello'      #First string
>>> str2 = 'World!'      #Second string
>>> str1 + str2          #Concatenated strings
'HelloWorld!'
                    #str1 and str2 remain same
>>> str1            #after this operation.
'Hello'
>>> str2
'World!'
```

8.3.2 تکرار (Repetition)

پائتھن ہمیں تکراری آپریٹر کا استعمال کرکے دی ہوئی اسٹرنگ کو دہرانے میں مدد کرتا ہے جسے علامت * سے ظاہر کیا جاتا ہے۔

```
#assign string 'Hello' to str1
>>> str1 = 'Hello'
#repeat the value of str1 2 times
>>> str1 * 2
'HelloHello'
#repeat the value of str1 5 times
>>> str1 * 5
'HelloHelloHelloHelloHello'
```

نوٹ: تکراری آپریٹر کا استعمال کرنے کے بعد بھی str1 اپنی اصل حالت میں موجود ہے۔

8.3.3 رکنیت (Membership)

پائتھن میں دو ممبرشپ آپریٹر ہوتے ہیں ایک 'in' اور دوسرا 'not in'۔ "in" آپریٹر دو اسٹرنگ کو لیتا ہے اور اگر پہلی اسٹرنگ دوسری اسٹرنگ میں ذیلی اسٹرنگ کے طور پر موجود ہوتی ہے تو آپریٹر True ظاہر کرتا ہے بصورت دیگر False ظاہر کر دیتا ہے۔

```
>>> str1 = 'Hello World!'
>>> 'W' in str1
True
>>> 'Wor' in str1
True
>>> 'My' in str1
False
```

"not in" آپریٹر بھی دو اسٹرنگ کو لیتا ہے اور اگر پہلی اسٹرنگ دوسری اسٹرنگ میں ذیلی اسٹرنگ کے طور پر

موجود نہیں ہے تو آپریٹر True ظاہر کرتا ہے بصورت دیگر False ظاہر کر دیتا ہے۔

```
>>> str1 = 'Hello World!'
>>> 'My' not in str1
True
>>> 'Hello' not in str1
False
```

8.3.4 سلائسنگ (Slicing)

پائتھن میں اسٹرنگ یا ذیلی اسٹرنگ کے کسی حصے کو ایکسس کرنے لیے ہم ایک طریقہ بروئے کار لاتے ہیں جسے سلائسنگ (Slicing) کہتے ہیں۔ اس کام کو ایک انڈیکس رینج متعین کر کے انجام دیا جا سکتا ہے۔ اسٹرنگ str1 دی ہوئی ہے، سلائس آپریشن str1[n:m]، اسٹرنگ str1 کے انڈیکس n (بشمول n) سے شروع ہونے والے اور m (m شامل نہیں ہے) پر ختم ہونے والے حصے کو ظاہر کر دیتا ہے۔ بالفاظ دیگر ہم کہہ سکتے ہیں کہ str1[n:m]، str1[n] سے شروع ہو کر str1[m-1] تک سبھی کیریکٹر کو ظاہر کر دیتا ہے۔ ذیلی اسٹرنگ میں موجود کیریکٹر کی تعداد ہمیشہ دونوں انڈیکس m اور n کے فرق کے مساوی ہوگی یعنی (m-n)

```
>>> str1 = 'Hello World!'
#gives substring starting from index 1 to 4
>>> str1[1:5]
'ello'
#gives substring starting from 7 to 9
>>> str1[7:10]
'orl'
#index that is too big is truncated down to
#the end of the string
>>> str1[3:20]
'lo World!'
#first index > second index results in an
#empty '' string
 >>> str1[7:2]
```

اگر پہلے انڈیکس کا ذکر نہیں کیا گیا ہے تو سلائسنگ 0 انڈیکس سے شروع ہوتی ہے۔

```
#gives substring from index 0 to 4
>>> str1[:5]
'Hello'
```

اگر دوسرے انڈیکس کا ذکر نہیں ہے تو سلائسنگ اسٹرنگ کے اختتام تک ہوتی ہے۔

```
#gives substring from index 6 to end
>>> str1[6:]
'World!'
```

سلائسنگ کے عمل کے لیے تیسرا انڈیکس بھی لیا جا سکتا ہے جو اسٹیپ سائز (Step size) کا تعین کرتا ہے۔ مثال کے طور پر str1[n:m:k] کا مطلب ہے کہ اسٹرنگ str1 میں n سے شروع کر کے اور m-1 پر ختم کرتے ہوئے ہر ایک K^{th} کیریکٹر علاحدہ کیا جاتا ہے۔ اگر اسٹیپ سائز متعین نہ کیا جائے تو اس کی قدر ایک ہوتی ہے۔

```
>>> str1[0:10:2]
'HloWr'
>>> str1[0:10:3]
'HlWl'
```

سلائسنگ کے لیے منفی انڈیکس کا استعمال بھی کیا جا سکتا ہے۔

```
 #characters at index -6,-5,-4,-3 and -2 are
 #sliced
>>> str1[-6:-1]
'World'
```

اگر ہم دونوں انڈیکس کو نظر انداز کر دیں اور اسٹیپ سائز -1 رکھیں تو

```
#str1 string is obtained in the reverse order
>>> str1[::-1]
'!dlroW olleH'
```

8.4 اسٹرنگ کا اعادہ (Traversing a String)

ہم For لوپ یا while لوپ کا استعمال کر کے اسٹرنگ کے ہر ایک کیریکٹر کو ایکسس کر سکتے ہیں اسٹرنگ کا اعادہ کر سکتے ہیں۔

(A) **For** لوپ کا استعمال کر کے اسٹرنگ کا اعادہ

```
>>> str1 = 'Hello World!'
>>> for ch in str1:
          print(ch,end = '')
Hello World!         #output of for loop
```

مندرجہ بالا کوڈ میں، لوپ کی ابتدا اسٹرنگ str1 کے پہلے کیریکٹر سے ہوتی ہے اور جب آخری کیریکٹر کو ایکسس کر لیا جاتا ہے تو یہ لوپ خود بہ خود اختتام پذیر ہو جاتا ہے۔

(B) **While** لوپ کا استعمال کر کے اسٹرنگ کا اعادہ

```
>>> str1 = 'Hello World!'
>>> index = 0
#len(): a function to get length of string
>>> while  index < len(str1):
          print(str1[index],end = '')
          index += 1

Hello World!            #output of while loop
```

یہاں while لوپ اس وقت تک جاری رہتا ہے جب تک index < len(str1) کا نتیجہ True ہے، جہاں انڈیکس 0 سے len(str1) -1 تک تبدیل ہوتا ہے۔

8.5 اسٹرنگ میتھڈز اور بلٹ ان فنکشن

(STING METHODS AND BUILT-IN FUNCTIONS)

پائتھن میں ایسے متعدد بلٹ ان فنکشن ہوتے ہیں جنھیں ہم اسٹرنگ میں استعمال کر سکتے ہیں۔ اسٹرنگ میں ردو بدل کے لیے عام طور سے استعمال کیے جانے والے کچھ بلٹ ان فنکشن جدول 8.2 میں دیے گئے ہیں۔

جدول 8.2 لسٹ میں ردوبدل کے لیے بلٹ ان فنکشن

میتھڈ	وضاحت	مثال

len()	دی ہوئی اسٹرنگ کی لمبائی کو ظاہر کرتا ہے۔	`>>> str1 = 'Hello World!'`
		`>>> len(str1)`
		`12`

title()	ایسی اسٹرنگ کو ظاہر کرتا ہے جس کے ہر ایک لفظ کا پہلا حرف بڑا اور باقی حروف چھوٹے ہوتے ہیں۔	`>>> str1 = 'hello WORLD!'`
		`>>> str1.title()`
		`'Hello World!'`

lower()	چھوٹے حروف پر مشتمل اسٹرنگ کو تبدیل کر کے بڑے حروف والی اسٹرنگ کی شکل میں ظاہر کرتا ہے۔	`>>> str1 = 'hello WORLD!'`
		`>>> str1.lower()`
		`'hello world!'`

upper()		`>>> str1 = 'hello WORLD!'`
		`>>> str1.upper()`
		`'HELLO WORLD!'`

count(str, sta end)	دی ہوئی اسٹرنگ میں ذیلی اسٹرنگ str کے وقوع پذیر ہونے کی تعداد کو ظاہر کرتا ہے۔ اگر ہم ابتدائی اور اختتامی انڈیکس کا ذکر نہیں کرتے ہیں تو تلاش کا عمل 0 سے شروع ہو کر اسٹرنگ کی لمبائی پر ختم ہوتا ہے۔	`>>> str1 = 'Hello World! Hello Hello'`
		`>>> str1.count('Hello',12,25)`
		`2`
		`>>> str1.count('Hello')`
		`3`

find(str,start end)	دی ہوئی اسٹرنگ میں وقوع پذیر ذیلی اسٹرنگ str کے انڈیکس کے پہلے وقوع کو ظاہر کرتا ہے۔ اگر ہم ابتدا اور اختتام کا ذکر نہیں کرتے ہیں تو تلاش کا عمل 0 سے شروع ہو کر اسٹرنگ کی لمبائی پر ختم ہوتا ہے۔ اگر دی ہوئی اسٹرنگ میں ذیلی اسٹرنگ موجود نہیں ہے تو فنکشن 1- ظاہر کرتا ہے۔	`>>> str1 = 'Hello World! Hello Hello'`
		`>>> str1.find('Hello',10,20)`
		`13`
		`>>> str1.find('Hello',15,25)`
		`19`
		`>>> str1.find('Hello')`
		`0`
		`>>> str1.find('Hee')`
		`-1`

index(sir, start, end)	find() کی طرح ہی ہے البتہ، اگر دی ہوئی اسٹرنگ میں ذیلی اسٹرنگ موجود نہیں ہے تو ValueError ظاہر ہوجاتی ہے۔	`>>> str1 = 'Hello World! Hello Hello'`
		`>>> str1.index('Hello')`
		`0`
		`>>> str1.index('Hee')`
		`ValueError: substring not found`

endswith()	اگر دی ہوئی اسٹرنگ فراہم کردہ ذیلی اسٹرنگ پر ختم ہوتی ہے تو یہ True ظاہر کرتا ہے بصورت دیگر False ظاہر کرتا ہے۔	`>>> str1 = 'Hello World!'`
		`>>> str1.endswith('World!')`
		`True`
		`>>> str1.endswith('!')`
		`True`
		`>>> str1.endswith('lde')`
		`False`

startswith()	اگر دی ہوئی اسٹرنگ فراہم کردہ ذیلی اسٹرنگ سے شروع ہوتی ہے تو یہ True ظاہر کرتا ہے بصورت دیگر False ظاہر کرتا ہے۔	`>>> str1 = 'Hello World!'`
		`>>> str1.startswith('He')`
		`True`
		`>>> str1.startswith('Hee')`
		`False`

isslnum()	اگر دی ہوئی اسٹرنگ کے کیریکٹر حروف تہجی یا اعداد ہیں تو یہ True ظاہر کرتا ہے۔ اگر خالی جگہ یا مخصوص علامات دی ہوئی اسٹرنگ کا حصہ ہیں یا اسٹرنگ خالی ہے تو یہ False ظاہر کرتا ہے۔	`>>> str1 = 'HelloWorld'`
		`>>> str1.isalnum()`
		`True`
		`>>> str1 = 'HelloWorld2'`
		`>>> str1.isalnum()`
		`True`
		`>>> str1 = 'HelloWorld!!'`
		`>>> str1.isalnum()`
		`False`

islower()	اگر اسٹرنگ غیر خالی ہے اور سبھی حروف تہجی چھوٹے ہیں یا کم از کم ایک کیریکٹر چھوٹا حرف تہجی ہے اور باقی کیریکٹر حروف تہجی نہیں ہیں تو یہ True ظاہر کرتا ہے۔	`>>> str1 = 'hello world!'`
		`>>> str1.islower()`
		`True`
		`>>> str1 = 'hello 1234'`
		`>>> str1.islower()`
		`True`
		`>>> str1 = 'hello ??'`
		`>>> str1.islower()`
		`True`
		`>>> str1 = '1234'`
		`>>> str1.islower()`
		`False`
		`>>> str1 = 'Hello World!'`
		`>>> str1.islower()`
		`False`

isupper()	اگر اسٹرنگ غیر خالی ہے اور سبھی حروف تہجی بڑے ہیں یا کم از کم ایک کیریکٹر بڑا حرف تہجی ہے اور باقی کیریکٹر حروف تہجی نہیں ہیں تو یہ True ظاہر کرتا ہے۔	`>>> str1 = 'HELLO WORLD!'`
		`>>> str1.isupper()`
		`True`
		`>>> str1 = 'HELLO 1234'`
		`>>> str1.isupper()`
		`True`
		`>>> str1 = 'HELLO ??'`
		`>>> str1.isupper()`
		`True`
		`>>> str1 = '1234'`
		`>>> str1.isupper()`
		`False`
		`>>> str1 = 'Hello World!'`
		`>>> str1.isupper()`
		`False`

isspace()	اگر اسٹرنگ غیر خالی ہے اور سبھی کیریکٹر خالی جگہیں (مثلاً ٹیب، نئی سطر، کیرج ریٹرن) ہیں تو یہ True ظاہر کرتا ہے۔	`>>> str1 = 'HELLO WORLD!'`
		`>>> str1.isupper()`
		`True`
		`>>> str1 = 'HELLO 1234'`
		`>>> str1.isupper()`
		`True`
		`>>> str1 = 'HELLO ??'`
		`>>> str1.isupper()`
		`True`
		`>>> str1 = '1234'`
		`>>> str1.isupper()`
		`False`
		`>>> str1 = 'Hello World!'`
		`>>> str1.isupper()`
		`False`

istitle()	اگر اسٹرنگ غیر خالی ہے اور اسٹرنگ میں ہر ایک لفظ کا پہلا حرف بڑا اور باقی چھوٹے ہیں تو یہ True ظاہر کرتا ہے۔	`>>> str1 = '      \n    \t \r'`
		`>>> str1.isspace()`
		`True`
		`>>> str1 = 'Hello           \n'`
		`>>> str1.isspace()`
		`False`

rstrip()	اسٹرنگ کے صرف بائیں جانب موجود خالی جگہوں کو ہٹانے کے بعد اسٹرنگ کو ظاہر کرتا ہے۔	`>>> str1 = 'Hello World!'`
		`>>> str1.istitle()`
		`True`
		`>>> str1 = 'hello World!'`
		`>>> str1.istitle()`
		`False`

strip()	اسٹرنگ کے صرف دائیں جانب موجود خالی جگہوں کو ہٹانے کے بعد اسٹرنگ کو ظاہر کرتا ہے۔	`>>> str1 = '          Hello World!`
		`'`
		`>>> str1.lstrip()`
		`'Hello World!`

replace
(oldstr,
newstr)
	اسٹرنگ کے دائیں اور بائیں دونوں جانب موجود خالی جگہوں کو ہٹانے کے بعد اسٹرنگ کو ظاہر کرتا ہے۔ پرانی اسٹرنگ جہاں جہاں موجود ہے اس کو نئی اسٹرنگ سے بدل دیتا ہے۔	>>> str1 = 'Hello World!'
		>>> str1.replace('o','*')
		'Hell* W*rld!'
		>>> str1 = 'Hello World!'
		>>> str1.replace('World','Country')
		'Hello Country!'
		>>> str1 = 'Hello World! Hello'
		>>> str1.replace('Hello','Bye')
		'Bye World! Bye'

join()	اسٹرنگ کے کیریکٹر کو کسی فاصل (Separator) کی مدد سے باہم متحد کر کے ظاہر کرتا ہے۔	>>> str1 = ('HelloWorld!')
		>>> str2 = '-' #separator
		>>> str2.join(str1)
		'H-e-l-l-o-W-o-r-l-d-!'

partition())	دی ہوئی اسٹرنگ کو ذیلی اسٹرنگ (فاصل) کے پہلی مرتبہ وقوع پذیر ہونے کے مقام پر منقسم کر دیتا ہے اور تین حصوں میں منقسم اسٹرنگ کو ظاہر کرتا ہے۔
	1۔ فاصل سے پہلے والی ذیلی اسٹرنگ
	2۔ فاصل
	3۔ فاصل کے بعد والی ذیلی اسٹرنگ
	اگر اسٹرنگ میں فاصل موجود نہیں ہے تو یہ پوری اسٹرنگ اور دو خالی اسٹرنگ کو ظاہر کر دیتا ہے۔
		>>> str1 = 'India is a Great Country'
		>>> str1.partition('is')
		('India ', 'is', ' a Great
		Country')
		>>> str1.partition('are')
		('India is a Great Country',' ','
		')

split()	ان الفاظ کی فہرست کو ظاہر کرتا ہے جن کی حد بندی متعینہ ذیلی اسٹرنگ کے ذریعے کی گئی ہو۔ اگر فاصل (Dilimiter) دیا ہوا نہیں ہے تو الفاظ کو خالی جگہ کے ذریعے علاحدہ کیا جائے گا۔	>>> str1 = 'India is a Great Country'
		>>> str1.split()
		['India','is','a','Great',
		'Country']
		>>> str1 = 'India is a Great
		Country'
		>>> str1.split('a')
		['Indi', ' is ', ' Gre', 't
		Country']

8.6 اسٹرنگ کو استعمال کرنا (HANDLING STRINGS)

اس سیکشن میں ہم پائتھن میں اسٹرنگ پر مختلف عملوں کو انجام دینے کے لیے استعمال کنندہ کے ذریعے تعریف شدہ فنکشن کے بارے میں پڑھیں گے۔

پروگرام 8-1 یہ شمار کرنے کے لیے کہ کوئی کیریکٹر (آرگیومینٹ کے طور پر پاس کیا گیا) دی ہوئی اسٹرنگ میں کتنی مرتبہ موجود ہے، استعمال کنندہ کے ذریعے تعریف شدہ فنکشن پر مشتمل ایک پروگرام لکھیے۔

```
#Program 8-1
#Function to count the number of times a character occurs in a
#string
def charCount(ch,st):
    count = 0
    for character in st:
        if character == ch:
            count += 1
    return count
#end of function

st = input("Enter a string: ")
ch = input("Enter the character to be searched: ")
count = charCount(ch,st)
print("Number of times character",ch,"occurs in the string
is:",count)
```

نتیجہ:

```
Enter a string: Today is a Holiday
Enter the character to be searched: a
Number of times character a occurs in the string is: 3
```

پروگرام 8-2 پیرامیٹر کے طور پر اسٹرنگ کے ساتھ استعمال کنندہ کے ذریعے تعریف شدہ فنکشن پر مشتمل ایک پروگرام لکھیے جو اسٹرنگ میں موجود سبھی مصوتوں کو * سے بدل دیتا ہے۔

```
#Program 8-2
#Function to replace all vowels in the string with  '*'
def replaceVowel(st):
    #create an empty string
    newstr = ''
    for character in st:
         #check if next character is a vowel
        if character in 'aeiouAEIOU':
```

```
               #Replace vowel with *
               newstr += '*'
          else:
               newstr += character
     return newstr
#end of function
st = input("Enter a String: ")
st1 = replaceVowel(st)
print("The original String is:",st)
print("The modified String is:",st1)
```

نتیجہ:

```
Enter a String: Hello World
The original String is: Hello World
The modified String is: H*ll* W*rld
```

پروگرام 8-3 ایک پروگرام لکھیے جس میں استعمال کنندہ سے ایک اسٹرنگ کو ان پٹ کرائیے اور اسے نئی اسٹرنگ کی تشکیل کیے بغیر رجعتی ترتیب میں پرنٹ کیجیے۔

```
#Program 8-3
#Program to display string in reverse order
st = input("Enter a string: ")
for i in range(-1,-len(st)-1,-1):
    print(st[i],end='')
```

نتیجہ:

```
Enter a string: Hello World
dlroW olleH
```

پروگرام 8-4 ایک پروگرام لکھیے جو پیرامیٹر کے طور پر پاس کی گئی اسٹرنگ کی ترتیب کو الٹ دیتا ہے اور اسے ایک نئی اسٹرنگ کی شکل میں اسٹور کرتا ہے۔ اسٹرنگ کی ترتیب کو الٹنے کے لیے استعمال کنندہ کے ذریعے تعریف شدہ فنکشن لکھیے۔

```
#Program 8-4
#Function to reverse a string
def reverseString(st):
     newstr = ''        #create a new string
     length = len(st)
```

```
    for i in range(-1,-length-1,-1):
        newstr += st[i]
    return newstr
#end of function
st = input("Enter a String: ")
st1 = reverseString(st)
print("The original String is:",st)
print("The reversed String is:",st1)
```

نتیجہ:

```
Enter a String: Hello World
The original String is: Hello World
The reversed String is: dlroW olleH
```

پروگرام 8-5 استعمال کنندہ کے ذریعے تعریف شدہ فنکشن پر مشتمل ایک پروگرام لکھیے جس میں اس بات کی جانچ کیجیے کہ آیا اسٹرنگ پیلنڈروم (palindrome) ہے یا نہیں۔ (کوئی اسٹرنگ اس وقت پیلنڈروم کہلاتی ہے اگر اسے بائیں سے دائیں اور دائیں سے بائیں پڑھا جائے تو اسٹرنگ وہی رہتی ہے۔ مثال کے طور پر Kanak ایک پیلنڈروم ہے۔)

```
#Program 8-5
#Function to check if a string is palindrome or not
def checkPalin(st):
    i = 0
    j = len(st) - 1
    while(i <= j):
        if(st[i] != st[j]):
            return False
        i += 1
        j -= 1
    return True
#end of function
st = input("Enter a String: ")
result = checkPalin(st)
if result == True:
    print("The given string",st,"is a palindrome")
else:
    print("The given string",st,"is not a
    palindrome")
```

نتیجہ 1:

```
Enter a String: kanak
The given string kanak is a palindrome
```

نتیجہ 2:

```
Enter a String: computer
The given string computer is not a palindrome
```

خلاصہ

- اسٹرنگ کیریکٹر کا ایک تسلسل ہے جنھیں اکہرے، دوہرے یا تہرے واوین کے اندر لکھا جاتا ہے۔
- اسٹرنگ کے اندر موجود ہر ایک انفرادی کیریکٹر کو ایکسس کرنے کے لیے انڈیکسنگ کا استعمال کیا جاتا ہے۔
- اسٹرنگ میں پہلے کیریکٹر کا انڈیکس 0 اور آخری کیریکٹر کا انڈیکس n-1 ہوتا ہے جہاں n اسٹرنگ کی لمبائی ہے۔ منفی انڈیکسنگ کی رینج n- سے 1- تک ہوتی ہے۔
- پائتھن میں اسٹرنگ ایک ناقابل تغیر ڈیٹا ٹائپ ہے۔ اس کا مطلب یہ ہے کہ اسٹرنگ کی تشکیل کے بعد اس کے مواد کو تبدیل نہیں کیا جا سکتا ہے۔
- ممبرشپ آپریٹر "in" دو اسٹرنگ کو لیتا ہے اور اگر پہلی اسٹرنگ دوسری اسٹرنگ میں ذیلی اسٹرنگ کے طور پر موجود ہوتی ہے تو آپریٹر True ظاہر کرتا ہے بصورت دیگر False ظاہر کر دیتا ہے۔ ممبرشپ آپریٹر "not in" مذکورہ بالا عمل کے برعکس کام کرتا ہے۔
- اسٹرنگ کے کسی ایک حصے کو ایکسس کرنے کا طریقہ سلائسنگ (Slicing) کہلاتا ہے۔ اس کام کو ایک انڈیکس رینج متعین کر کے انجام دیا جا سکتا ہے۔ سلائس آپریشن str1[n:m]، اسٹرنگ str1 کے انڈیکس n (بشمول n) سے شروع ہونے والے اور m (m شامل نہیں ہے) پر ختم ہونے والے حصے کو ظاہر کر دیتا ہے۔
- اسٹرنگ کے ہر ایک کیریکٹر کو For لوپ یا while لوپ کا استعمال کر کے ایکسس کر سکتے ہیں۔
- پائتھن میں ایسے متعدد بلٹ ان فنکشن ہوتے ہیں جنھیں ہم اسٹرنگ میں استعمال کر سکتے ہیں۔

مشق

1۔ مندرجہ ذیل اسٹرنگ mySubject: پر غور کیجیے۔

```
mySubject = "Computer Science"
```

مندرجہ ذیل اسٹرنگ آپریشن کا نتیجہ کیا ہوگا:

```
   i.  print(mySubject[0:len(mySubject)])
  ii.  print(mySubject[-7:-1])
 iii.  print(mySubject[::2])
  iv.  print(mySubject[len(mySubject)-1])
   v.  print(2*mySubject)
  vi.  print(mySubject[::-2])
 vii.  print(mySubject[:3] + mySubject[3:])
viii.  print(mySubject.swapcase())
  ix.  print(mySubject.startswith('Comp'))
   x.  print(mySubject.isalpha())
```

2۔ مندرجہ ذیل اسٹرنگ myAddress: پر غور کیجیے:

```
myAddress = "WZ-1,New Ganga Nagar,New Delhi"
```

مندرجہ ذیل اسٹرنگ آپریشن کا نتیجہ کیا ہوگا:

```
   i.  print(myAddress.lower())
  ii.  print(myAddress.upper())
 iii.  print(myAddress.count('New'))
  iv.  print(myAddress.find('New'))
   v.  print(myAddress.rfind('New'))
  vi.  print(myAddress.split(','))
 vii.  print(myAddress.split(' '))
viii.  print(myAddress.replace('New','Old'))
  ix.  print(myAddress.partition(','))
   x.  print(myAddress.index('Agra'))
```

پروگرامنگ سوالات (PROGRAMMING PROBLEMS)

1۔ ایک پروگرام لکھیے جس میں استعمال کنندہ سے اینٹر کلید دبائے جانے تک متن کی سطر (سطریں) داخل کرنے کے لیے کہا گیا ہو۔ متن میں موجود کیریکٹر کی کل تعداد (جس میں وہائٹ اسپیس (White spaces) بھی شامل ہوں [، حروف تہجی کی کل تعداد، ہندسوں کی کل تعداد، مخصوص علامات کی کل تعداد اور الفاظ (یہ فرض کیجیے کہ ہر ایک لفظ دوسرے لفظ سے خالی جگہ کے ذریعے علاحدہ ہے) کی کل تعداد شمار کیجیے۔

2۔ ایک سے زیادہ الفاظ پر مشتمل اسٹرنگ کو ٹائٹل کیس (title case) اسٹرنگ میں تبدیل کرنے کے لیے استعمال کنندہ کے ذریعے تعریف شدہ فنکشن پر مشتمل پروگرام لکھیے جب کہ اسٹرنگ کو پیرامیٹر کے طور پر پاس کیا گیا ہو۔ (ٹائٹل کیس کا مطلب ہے کہ ہر ایک لفظ کا پہلا حرف بڑا ہوگا)

3۔ ایک فنکشن deleteChar() لکھیے جو دو ایسے پیرامیٹر کو قبول کرتا ہے جن میں سے ایک اسٹرنگ اور دوسرا کیریکٹر ہے۔ یہ فنکشن اس کیریکٹر کو اسٹرنگ میں موجود سبھی جگہوں سے حذف کر کے نئی اسٹرنگ کی تشکیل کرتا ہے اور اس نئی اسٹرنگ کو ظاہر کرتا ہے۔

4۔ ایک ایسی اسٹرنگ کو داخل (Input) کیجیے جس میں کچھ ہندسے ہوں۔ اس اسٹرنگ میں موجود سبھی ہندسوں کے حاصل جمع کو ظاہر کرنے کے لیے ایک فنکشن لکھیے۔

5۔ ایک فنکشن لکھیے جو کسی جملے کو ایک ان پُٹ پیرامیٹر کے طور پر لیتا ہے۔ جملے میں ہر ایک لفظ دوسرے لفظ سے خالی جگہ کے ذریعے علاحدہ ہے۔ یہ فنکشن ہر ایک خالی جگہ کو علامت خط (Hyphen) سے بدل دیتا ہے اور ترمیم شدہ جملے کو ظاہر کر دیتا ہے۔