

Tuple Manipulation

सीबीएसई पाठ्यक्रम पर आधारित
कक्षा -11

अध्याय - 8

द्वारा:

संजीव भदौरिया

स्नातकोत्तर शिक्षक (संगणक विज्ञान)

के० वि० बाराबंकी (लखनऊ संभाग)

परिचय

- Python में tuple भी एक प्रकार का पात्र (container) होता है जिसमें किसी भी प्रकार का मानो (values) की सूची रखी जा सकती है।
- Tuple एक अपरिवर्तनीय (immutable) data type होता है अर्थात् आप tuple के किसी भी value को बदल नहीं सकते हैं।
- Tuple एक प्रकार से string और list जैसी ही sequence होती है बस सिर्फ अंतर इतना होता है की list की values बदली जा सकती हैं (mutable) और tuple या string की नहीं (immutable)।
- इस अध्याय में हम tuple के manipulation के बारे में पढ़ेंगे, जिसमें हम tuple बनाना, उसको प्रयोग करना और tuple पर कुछ क्रियाएं (operation) built in functions के द्वारा सीखेंगे।

Tuple को बनाना


- Python में tuple को बनाने के लिए “()” कोष्ठक का प्रयोग किया जाता है |


()	empty tuple
(1, 2, 3)	integers का tuple
(1, 2.5, 3.7, 7)	numbers का tuple
('a', 'b', 'c')	characters का tuple
('a', 1, 'b', 3.5, 'zero')	mixed values का tuple
('one', 'two', 'three', 'four')	string का tuple

Tuple एक प्रकार के immutable sequence होते हैं जिनके किसी भी मान में हम परिवर्तन नहीं कर सकते।


Tuple को बनाना

tuple बनाने के लिए निम्न उदाहरण पर ध्यान दें -

- Empty tuple: 

```
>>> t=()
>>> t
()
```
- Single element tuple: 

```
>>> t=(1)
>>> t
1
```
- Long tuple:

```
>>> t=(0,1,4,9,16,25,36,49,64,81,100,121)
>>> t
(0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121)
```
- Nested tuple: 

```
>>> t=(1,2,3,(4,5))
>>> t
(1, 2, 3, (4, 5))
```

Tuple को बनाना

tuple को अन्य sequences से बनाने के लिए निम्न उदाहरण पर ध्यान दें इसके लिए tuple() फंक्शन का प्रयोग करते हैं।-

String से tuple बनाना

```
>>> t=tuple("Hello")
>>> t
('H', 'e', 'l', 'l', 'o')
```

List से tuple बनाना

```
>>> L=['a', 'e', 'i', 'o', 'u']
>>> T=tuple(L)
>>> T
('a', 'e', 'i', 'o', 'u')
```

Input लेकर tuple बनाना

```
>>> t1=tuple(input("Enter element"))
Enter element123456
>>> t1
('1', '2', '3', '4', '5', '6')
```

ये सभी character type के element बन गए हैं। यदि इन्हें विभिन्न प्रकार के elements बनाना है तो निम्न statement लिखना होगा।

```
Tuple=eval(input("Enter elements"))
```

```
>>> t1=eval(input("Enter the elements"))
Enter the elements (2,4.5,"hello")
>>> t1
(2, 4.5, 'hello')
```

Tuple को Access करना

- Python में tuple को access करने के लिए आप वे सभी कार्य कर सकते हैं जो की हमने list के साथ किये थे | list के जैसे ही हम tuple के प्रत्येक element को access कर सकते हैं |
- **List के साथ समानता:** list के जैसे tuple में भी index होता है | tuple और tuple में mutability को छोड़कर सारे गुण समान हैं |

Forward index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Tuple	R	E	S	P	O	N	S	I	B	I	L	I	T	Y
Backward index	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- List की लम्बाई पता करने के लिए len () फंक्शन का प्रयोग करते हैं

```
>>> t=tuple("Hello")
>>> len(t)
5
```

Tuple को Access करना

- Indexing और Slicing:

- `T[i]` index `i` पर उपस्थित item को return करता है |
- `T[i:j]` ये एक नया tuple return करता है जिसमे `T` के `i` से `j` index तक के item होते हैं |
- `T[i:j:n]` ये एक नया tuple return करता है जिसमे `T` के `i` से `j` index तक के `n`वें (n^{th}) item होते हैं |

```
>>> T=(1,2,3,4,5,6,7,8,9,10)
>>> T[1:10:3]
(2, 5, 8)
```

- Membership operator:

- Tuple में membership operator “in” तथा “not in” ठीक वैसे ही कार्य करते हैं जैसे की list में करते हैं| (कृपया इसके list वाले अध्याय को देखें)

- Concatenation और Replication operators:

- Tuple के साथ + operator दूसरे tuple को पहले tuple के बाद में जोड़ देता है| जबकि * tuple के elements को repeat कर देता है|

Tuple को Access करना

- Individual elements को access करना (उदहारण देखें)-

```
>>> L=['a', 'e', 'i', 'o', 'u']
>>> L[0]
'a'
>>> L[3]
'o'
```

- Tuple को travers करने के लिए निम्न code लिखना होगा –

```
for <item> in <tuple>:
    #प्रत्येक item को process करिए
```

```
T=tuple("Python")
print(T, end="")
print()
for a in T:
    print(a)
```

```
('P', 'y', 't', 'h', 'o', 'n')
P
y
t
h
o
n
```

OUTPUT

Tuple Operations

- Tuple को join करना
 - जोड़ने के लिए + के साथ दोनों tuple होने चाहिए।

```
>>> tp1=(1,3,5)
>>> tp2=(6,7,8)
>>> tp1+tp2
(1, 3, 5, 6, 7, 8)
>>> tp3=tp1+tp2
>>> tp3
(1, 3, 5, 6, 7, 8)
```

- Tuple को join करने में निम्न में error आयेगी।

- Tuple + number करने में
- Tuple + complex number करने में
- Tuple + string करने में
- Tuple + list करने में
- Tuple + (5) करने में भी error आयेगी क्योंकि single वैल्यू के साथ tuple को वैल्यू ही माना जाता है न की tuple ।

- Tuple को Replicate करना

```
>>> tp1=(1,3,5)
>>> tp1*3
(1, 3, 5, 1, 3, 5, 1, 3, 5)
>>> tp1
(1, 3, 5)
>>> tp2=tp1*3
>>> tp2
(1, 3, 5, 1, 3, 5, 1, 3, 5)
```

Tuple की Slicing

```
>>> tpl=(10,12,14,20,22,24,30,32,34)
```

```
>>> seq=tpl[3:-3]
```

```
>>> seq
```

```
(20, 22, 24)
```

Upper limit चाहे जो हो tuple पूरी list के अंतिम item तक ही दिखायेगा

```
>>> tpl[3:30]
```

```
(20, 22, 24, 30, 32, 34)
```

```
>>> tpl[-15:7]
```

```
(10, 12, 14, 20, 22, 24, 30)
```

```
>>> tpl[0:10:2]
```

हर दूसरा item दिखायेगा |

```
(10, 14, 22, 30, 34)
```

```
>>> tpl[0:10:3]
```

हर तीसरा item दिखायेगा |

```
(10, 20, 30)
```

```
>>> tpl[::3]
```

```
(10, 20, 30)
```

Tuple की तुलना (Comparison)

```
>>> a=(2,3)
>>> b=(2,3)
>>> a==b
True
>>> c=('2','3')
>>> a==c
False
>>> a>b
False
>>> d=(2.0,3.0)
>>> d>a
False
>>> d==a
True
>>> e=(2,3,4)
>>> a<e
True
```

इस उदाहरण से आप तुलनात्मक प्रकृति का अंदाजा लगा सकते हैं ।

Tuple को unpack करना

```
>>> t=(2,3,'A','B')
>>> w,x,y,z=t
>>> print(w)
2
>>> print(x)
3
>>> print(y)
A
>>> print(z)
B
```

इस उदाहरण से आप समझ सकते हैं कि tuple को unpack करना कितना आसान है और किस प्रकार इन्हें unpack किया जा सकता है।

Tuple को delete करना

जैसा की हम जानते हैं की tuple एक immutable type है तो इसके किसी individual element को हम delete नहीं कर सकते। किन्तु हम `del()` से सम्पूर्ण tuple को delete कर सकते हैं | निम्न उदाहरण को देखिये -

```
>>> t=(2,3,'A','B')
```

```
>>> del t[2]
```

एक item delete नहीं हो सकता इसलिए error आगई।

```
Traceback (most recent call la:
```

```
File "<pyshell#65>", line 1,
```

```
del t[2]
```

```
TypeError: 'tuple' object does:
```

```
>>> del t
```

```
>>> t
```

```
Traceback (most recent call la:
```

```
File "<pyshell#67>", line 1,
```

```
t
```

```
NameError: name 't' is not def:
```

```
....
```

पूरा tuple delete हो गया। इसके बाद जब t को print करते हैं तो error आती है।

Tuple Functions

```
>>> emp= ('Ram', 25000, 24, 'LKO')
```

```
>>> len(emp) ←
```

```
4
```

len() Function

```
>>> tpl=(10,12,14,16,18,20,22)
```

```
>>> max(tpl) ←
```

```
22
```

max() Function

```
>>> tpl2=("Karan","Zubin","Zara","Ana")
```

```
>>> max(tpl2)
```

```
'Zubin'
```

```
>>> min(tpl) ←
```

```
10
```

min() Function

```
>>> min(tpl2)
```

```
'Ana'
```

```
>>> tpl2.index("Zubin") ←
```

```
1
```

index() Function

```
>>> tpl3=(10,12,14,16,10,18,20,10,22)
```

```
>>> tpl3.count(10)
```

```
3
```

```
>>> t=tuple("Hello") ←
```

```
>>> t
```

```
('H', 'e', 'l', 'l', 'o')
```

tuple() Function

और अधिक पाठ्य-सामग्री हेतु निम्न लिंक पर क्लिक करें -

www.pythontrends.wordpress.com