PES UNIVERSITY

Introduction to Tuples

- At the end of this class, students will be able to-
  - Use the variable type – Tuple
  - Create and use Tuples using Tuple built – in functions

Tuples

A **tuple** is an *immutable* linear data structure. Thus, in contrast to lists, once a tuple is defined it cannot be altered. Otherwise, tuples and lists are essentially the same.

To distinguish tuples from lists, tuples are denoted by parentheses instead of square brackets as given below,

```
nums = (10, 20, 30)
student = ('John Smith', 48, 'Computer Science', 3.42)
```

Another difference of tuples and lists is that *tuples of one element must include a comma following the element*. Otherwise, the parenthesized element will not be made into a tuple, as shown below,

CORRECT	WRONG
>>> (1,) (1)	>>> (1) 1

An **empty tuple** is represented by a set of empty parentheses, ().

The elements of tuples are accessed the same as lists, with square brackets,

```
>>> nums[0]
10
```

```
>>> student[0]
'John Smith'
```

Any attempt to alter a tuple is invalid. Thus, delete, update, insert and append operations are not defined on tuples.

Properties of Tuples

- A tuple has zero or more elements
- The element of the tuple can be of any type.
- There is no restriction on the type of the element.
- A tuple has elements which could be of the same type(homogeneous) or of diferent types(heterogeneous)

- Each element of a tuple can be referred to by a index or a subscript
- An index is an integer Access to an element based on index or position takes the same time no matter where the element is in the tuple – random access
- Tuples are immutable. Once created, we cannot change the number of elements – no append, no insert, no remove, no delete.
- Elements of the tuple cannot be assigned.
- A tuple can not grow and cannot shrink. Size of the tuple can be found using the function len.

- Elements in the tuple can repeat
- Tuple is a sequence
- Tuple is also iterable - is eager and not lazy.
- Tuples can be nested. We can have tuple of tuples.
- Assignment of one tuple to another causes both to refer to the same tuple.
- tuples can be sliced. This creates a new tuple

Sequences

A **sequence** in Python is a linearly-ordered set of elements accessed by index number.

Lists, tuples and strings are all sequences. Strings, like tuples, are immutable, therefore they cannot be altered.

Operation		String s = 'hello' w = '!'	Tuple s = (1,2,3,4) w = (5,6)	List s = [1,2,3,4] w = [5,6]
Length	`len(s)`	`5`	`4`	`4`
Select	`s[0]`	`'h'`	`1`	`1`
Slice	`s[1:4]` `s[1:]`	`'ell'` `'ello'`	`(2, 3, 4)` `(2, 3 ,4)`	`[2 ,3, 4]` `[2, 3, 4]`
Count	`s.count('e')` `s.count(4)`	`1` `--`	`--` `1`	`--` `1`
Index	`s.index('e')` `s.index(3)`	`1` `--`	`--` `2`	`--` `2`
Membership	`'h' in s`	`True`	`False`	`False`
Concatenation	`s + w`	`'hello!'`	`(1, 2, 3, 4, 5, 6)`	`[1, 2, 3, 4, 5, 6]`
Minimum Value	`min(s)`	`'e'`	`1`	`1`
Maximum Value	`max(s)`	`'o'`	`4`	`4`
Sum	`sum(s)`	`n/a`	`10`	`10`

Operations	Operations

empty tuple
◦ t1 = ()
◦ t2 = tuple()
	b = ([12, 23], {34 : 45}, "56")
	print(b, len(b))
	b[0].append(67) #ok #
	b[0] = [78, 89] #no
	#del b[0] # no
	#b[0] += [100] # no ; assignment forbidden

a=(1) #int
a=(1,) #tuple	

a = (11, 33, 22, 44, 55) print(a[2]) # 22 print(a[2:4]) # (22, 44)	c = [11, 22, 33, 44] for i in c : print("one") # 4 times for i in [c] : print("two") # once for i in (c) : # not a tuple print("three") # 4 times for i in (c,) : # a tuple print("four") # once
#a[2] = 222 # NO #a.append(66) # Error	
a new tuple created a = a + (111, 222) print(a)	e = (11, 33, 11, 11, 44, 33) print(e.count(55)) # 0 print(e.index(44)) # 4
print((3, 4) * 2) # (3, 4, 3, 4) print((3 + 4) * 2) # 14 print((3 + 4,) * 2) # (7, 7)	a=1,2,3 x,y,z=a a,b=b,a

Summary

- Tuple is used as a means of structuring and accessing a collection of data.
- Tuples organize data in a linear sequence
- Tuples are immutable