**Exercise No: 6**

**Date: 11.10.2020**

**Aim:** Predicting python outputs for the given code

**Program:**

|  |  |  |
| --- | --- | --- |
|  | # Create a tuple, also called tuple packing.  numbers = 1, 2 | |
|  | print(numbers) **# Output: (1, 2)** | |
|  | # Create tuple with paranthesis. | |
|  | numbers = (1, 2, 3) | |
|  | print(numbers) **# Output: (1, 2, 3)** | |
|  | # Create an empty tuple. | |
|  | numbers = () | |
|  | print(numbers) **# Output: ()** | |
|  | # Create a tuple with one item. Note that the trailing comma is necessary | |
|  | numbers = 1, | |
|  | print(numbers) # Output: (1,) | |
|  | # Create a tuple with heterogenous items. | |
|  | random\_tuple = "Hey", (1, 2), 1, ["you"] | |
|  | print(random\_tuple) **# Output: ("Hey", (1, 2), 1, ["you"])** | |
|  | # Create tuple with tuple() constructor. | |
|  | numbers = tuple() | |
|  | print(numbers) **# Output: ()** | |
|  | numbers = tuple([1, 2]) # Takes any sequence as input | |
|  | print(numbers) **# Output: (1, 2)**  **#methods on tuples** | |
|  | | | # Get length of list by using len() method. |
|  | | | numbers = 5, 8, 8 |
|  | | | print(len(numbers)) # Output: 3 |
|  | | | # Get index of an element using the index() method. |
|  | | | numbers = 5, 8, 8 |
|  | | | print(numbers.index(8)) # Output: 1 |
|  | | | # Count occurences of an item in a tuple. |
|  | | | numbers = 5, 8, 8 |
|  | | | print(numbers.count(8)) # Output: 2  eggs =(‘hello’,42,0.5)  eggs[0] **#output: hello**  eggs[1:3] **#output: (42,0.5)**  len(eggs) **#output: 3** |

|  |  |
| --- | --- |
|  | **# Access elements of a tuple by indexing.**  str\_tuple = "hey", "there!", "how", "are", "you?" |
|  | print(str\_tuple[0]) **# Output: "hey"** |
|  | print(str\_tuple[len(str\_tuple) - 1]) **# Output: "you?"** |
|  | print(str\_tuple[-1]) **# Output: "you?"** |
|  |  |
|  | **# Slicing a tuple.** |
|  | str\_tuple = "hey", "there!", "how", "are", "you?" |
|  | print(str\_tuple[2:]) **# Output: ("how", "are", "you?")** |
|  | print(str\_tuple[:2]) **# Output: ("hey", "there!")** |
|  | print(str\_tuple[-3:]) **# Output: ("how", "are", "you?")** |
|  | print(str\_tuple[:-3]) **# Output: ("hey", "there!")** |
|  | print(str\_tuple[1:4]) **# Output: ("there!", "how", "are")** |
|  | # Get a copy of the tuple by slicing. |
|  | print(str\_tuple[:]) **# Output: ("hey", "there!", "how", "are", "you?")** |

|  |
| --- |
|  |
|  | **# Concatenate tuples.**  numbers = (1, 2) |
|  | strings = ("Hey", "there") |
|  | print(numbers + strings) **# Output: (1, 2, "Hey", "there")**  **#looping through tuple using ‘in’**  numbers = 1, 2  for number in numbers:  print(number)   |  | | --- | |  | |

|  |
| --- |
| **# Check if element is present in tuple**  numbers = 1, 2 |
| print(1 in numbers) **# Output: True** |
| print(5 in numbers) **# Output: False** |

|  |
| --- |
|  |
|  |
|  |
|  | # We are packing two items 1 and 2 into the tuple.  numbers =1, 2 |
|  |  |
|  | # Tuple sequence unpacking. |
|  | # Number of variables used has to be same as the number of items in the tuple. |
|  | # Unpacking the tuple and assigning its items to x and y. |
|  | x, y = numbers |
|  | # Note that this is also packing the args as a tuple which gets unpacked as the print  method's arguments. |
|  | print(x, y) **# Output # (1, 2)** |

**Result:** The above program has been successfully verified