Practical No. 9 : Write python program to perform following operations on tuple: Create, Access, Print, Delete & Convert tuple into list and vice-versa

- ## Practical related questions

1. **Define empty tuple. Write syntax to create empty tuple.**

➔

A tuple is an ordered, immutable collection of elements. An empty tuple is a tuple that contains no elements.

Syntax to create an empty tuple:
empty_tuple = ()

Example:

```
empty_tuple = ()
print("Empty tuple:", empty_tuple)
```

Output:
Empty tuple: ()

2. **Write syntax to copy specific elements existing tuple into new tuple.**

➔

Syntax to Copy Specific Elements from an Existing Tuple into a New Tuple:
To copy specific elements from an existing tuple into a new tuple, you can use slicing.

Slicing -
Slicing allows you to select a range of elements from the tuple and store them in a new tuple.

Example:

```
existing_tuple = (10, 20, 30, 40, 50)
new_tuple = existing_tuple[1:4]
print("New tuple:", new_tuple)
```

Output:

New tuple: (20, 30, 40)

3. Compare tuple with list (Any 4 points).

➔

Tuple	List
Immutable (Cannot be changed after creation).	Mutable (Can be changed after creation)
Defined with parentheses ()	Defined with square brackets [ ]
Faster than lists due to immutability	Slower compared to tuples
Used for fixed collections of data	Used for collections that may change during execution

4. Create a tuple and find the minimum and maximum number from it.

➔

```
my_tuple = (10, 20, 5, 40, 30)

min_num = min(my_tuple)
max_num = max(my_tuple)

print("Minimum number:", min_num)
print("Maximum number:", max_num)
```

Output:

```
Minimum number: 5
Maximum number: 40
```

5. Write a Python program to find the repeated items of a tuple.

→

```
my_tuple = (10, 20, 30, 20, 40, 30, 50, 20)
repeated_items = []
for item in my_tuple:
    if my_tuple.count(item) > 1 and item not in repeated_items:
        repeated_items.append(item)
print("Repeated items:", repeated_items)
```

Output:

Repeated items: [20, 30]

6. Print the number in words for Example: 1234 => One Two Three Four.

→

```
num_dict = {0: 'Zero', 1: 'One', 2: 'Two', 3: 'Three', 4: 'Four',
            5: 'Five', 6: 'Six', 7: 'Seven', 8: 'Eight', 9: 'Nine'}
num = 1234
for digit in str(num):
    print(num_dict[int(digit)], end=' ')
```

Output:

One Two Three Four