Python Tuples Exercises

In [ ]:
```
#Q1 Create myTuple tuple with the follwoing values ("NPower","JDA","Tuesday",30,3
```

In [ ]:
```
#Q2 What is the type of myTuple
```

In [ ]:
```
#Q3 What is the length of myTuple
```

In [ ]:
```
#Q4 print the values in each index #Use regular indexing
```

In [ ]:
```
#Q5 print the values in each index #Use negative indexing
```

In [ ]:
```
#Q6 what is the type of each value
```

In [ ]:
```
#Q7 unpack myTuple in the follwoeing variables name,program,dayName,month,day,yea
# print the variables
```

In [ ]:
```
#Q8 unpack myTuple2 in the follwoeing variables name,program,dayName.
# What will happen to variables (name,program,dayName) and (month,day,year)
```

In [ ]:
```
# Note the following
Tuple1=("Jerry",2,89) #This is a tuple with 3 elements
Tuple2=("Ulan",)#This is a tuple with 1 element
test="Leul" #This is a VARIABLE with string value

a,b,c=Tuple1
print("Type a",type(a))
print(a,b,c)

d=Tuple2
print(type(d))
print(d)

e=test
print(e)
```

In [ ]:

```
#Tuples are immutable
#we can always make the testTuple variable reference a new tuple in the memory
#and hold a different information

testTuple=(1,2,3)
print(testTuple)

testTuple=(4,5,6)
print(testTuple)

#But we can't change or edit a value for the existing tuple

testTuple[0]=6 #ERROR 'tuple' object does not support item assignment
```

In [ ]:

```
#Q9 Reverse myTuple, output should looks like (2021,3,30,"Tuesday","JDA","NPower'
```

In [ ]:

```
#Q10 Create nestedTuple=(("Coursera","course",6),("week",(2,"Lists","Tuple")))
```

In [ ]:

```
#Q11 What is the output of nestedTuple[1:2]
```

In [ ]:

```
#Q12 print each element in the nestedTuple
```

In [ ]:

```
#Q13 Access (2,"Lists","Tuple") from nestedTuple
```

In [ ]:

```
#Q14 Access the value "Lists" from nestedTuple
```

In [ ]:

```
#Q15 Access the value "Tuple" from nestedTuple
```

In [ ]:

```
#Q16 Access the value "course" from nestedTuple
```

In [ ]:

```
#Q17 Concatenate myTuple with nestedTuple
```

In [ ]:

```
#Q18 add your name to the tuple
```

In [ ]:

```
#Q19 check whether Coursera exists within  myTuple

# NOTE "in" doesn't work properly with nested tuples
```

In [ ]:

```
#Q20 check whether 55 exists within testTuple
testTuple=(1,2,33,55,6,55)
```

In [ ]:

```
#Q21 Find the index of JDA in myTuple


# Find the index of 'Coursera' in myTuple
# NOTE index doesn't work properly with nested tuples # Wrong output
```

In [ ]:

```
#Q22 print index 8 from myTuple
```

In [ ]:

```
#Q23 Get the 4th element from the begining of myTuple and 4th element from last
```

In [ ]:

```
#Q24 Find how many times 55 appeared in testTuple [Hint: Use method count()]
```