In [30]:
```
#Q1 Create myTuple tuple with the follwoing values ("NPower","JDA","Tuesday",30,3,2021)
myTuple=("NPower","JDA","Tuesday",30,3,2021)
print(myTuple)
```

```
('NPower', 'JDA', 'Tuesday', 30, 3, 2021)
```

In [3]:
```
#Q2 What is the type of myTuple
type(myTuple)
```

Out[3]:
```
tuple
```

In [4]:
```
#Q3 What is the length of myTuple
len(myTuple)
```

Out[4]:
```
6
```

In [8]:
```
#Q4 print the values in each index #Use regular indexing
print(myTuple[0],myTuple[1],myTuple[2],myTuple[3],myTuple[4],myTuple[5])
```

```
NPower JDA Tuesday 30 3 2021
```

In [9]:
```
#Q5 print the values in each index #Use negative indexing
print(myTuple[-6],myTuple[-5],myTuple[-4],myTuple[-3],myTuple[-2],myTuple[-1])
```

```
NPower JDA Tuesday 30 3 2021
```

In [10]:
```
#Q6 what is the type of each value
print(type(myTuple[0]),type(myTuple[1]),type(myTuple[2]),type(myTuple[3]),type(myTuple[4]),type(myTuple[5]))
```

```
<class 'str'> <class 'str'> <class 'str'> <class 'int'> <class 'int'> <class 'int'>
```

In [7]:
```
#Q7 unpack myTuple in the follwoeing variables name,program,dayName,month,day,year accordingly
# print the variables
(name,program,dayName,month,day,year)=myTuple
print(name,program,dayName,month,day,year)
```

```
NPower JDA Tuesday 30 3 2021
```

In [8]:
```
#Q8 unpack myTuple2 in the follwoeing variablesname,program,dayName.
# What will happen to variables (name,program,dayName) and (month,day,year)
myTuple2=("Jerry",2,89)
(name,program,dayName)=myTuple2
print(name,program,dayName)
print(month,day,year)
```

```
Jerry 2 89
30 3 2021
```

In [12]:
```
# 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)
```

```
Type a <class 'str'>
Jerry 2 89
<class 'str'>
Ulan
Leul
```

In [ ]:
```
#Tuples are immutable
#we can always make the testTuple variable reference a new tuple in the memory
#and holding 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 [17]:
```
#Q9 Reverse myTuple, output should looks like ("NPower","JDA","Tuesday",30,3,2021)
print(myTuple[::-1])
print(tuple(reversed(myTuple)))
```

```
(2021, 3, 30, 'Tuesday', 'JDA', 'NPower')
(2021, 3, 30, 'Tuesday', 'JDA', 'NPower')
```

In [16]:
```
#Q10 Create nestedTuple=(("Coursera","course",6),("week",(2,"Lists","Tuple")))
nestedTuple=(("Coursera","course",6),("week",(2,"Lists","Tuple")))
print(nestedTuple)
```

```
(('Coursera', 'course', 6), ('week', (2, 'Lists', 'Tuple')))
```

In [20]:
```
#Q11 What is the output of nestedTuple[1:2]
nestedTuple[1:2]
```

Out[20]:
```
(('week', (2, 'Lists', 'Tuple')),)
```

In [25]:
```
#Q12 print each element in the nestedTuple
print(nestedTuple[0][0],nestedTuple[0][1],nestedTuple[0][2])
print(nestedTuple[1][0])
print(nestedTuple[1][1][0],nestedTuple[1][1][1],nestedTuple[1][1][2])
```

```
Coursera course 6
week
2 Lists Tuple
```

In [26]:
```
#Q13 Access (2,"Lists","Tuple") from nestedTuple
print(nestedTuple[1][1][0],nestedTuple[1][1][1],nestedTuple[1][1][2])
```

```
2 Lists Tuple
```

In [27]:
```
#Q14 Access "Lists" from nestedTuple
nestedTuple[1][1][1]
```

Out[27]:
```
'Lists'
```

In [28]:
```
#Q15 Access "Tuple" from nestedTuple
nestedTuple[1][1][2]
```

Out[28]:
```
'Tuple'
```

In [29]:
```
#Q16 Access "course" from nestedTuple
nestedTuple[0][1]
```

Out[29]:
```
'course'
```

In [32]:
```
#Q17 Concatenate myTuple with nestedTuple
myTuple=myTuple+nestedTuple
print(myTuple)
```

```
('NPower', 'JDA', 'Tuesday', 30, 3, 2021, ('Coursera', 'course', 6), ('week', (2, 'Lists', 'Tuple')))
```

In [33]:
```
#Q18 add your name to the tuple
myTuple=myTuple+("Sowji",)
print(myTuple)
```

```
('NPower', 'JDA', 'Tuesday', 30, 3, 2021, ('Coursera', 'course', 6), ('week', (2, 'Lists', 'Tuple')), 'Sowji')
```

In [21]:
```
#Q19 check whether Coursera exists within a myTuple

# NOTE in doesn't work properly with nested tuples # Wrong output
"Coursera" in myTuple
```

Out[21]:
```
False
```

In [35]:
```
#Q20 check whether an element exists within a testTuple
2021 in myTuple
```

Out[35]:
```
True
```

In [40]:
```
#Q21 Find the index of JDA in myTuple
myTuple.index("JDA")


# Find the index of 'Coursera' in myTuple
myTuple.index('Coursera')

# NOTE index doesn't work properly with nested tuples # Wrong output
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-40-07d68b738e36> in <module>
      4
      5 # Find the index of 'Coursera' in myTuple
----> 6 myTuple.index('Coursera')
      7
      8 # NOTE index doesn't work properly with nested tuples # Wrong output

ValueError: tuple.index(x): x not in tuple
```

In [37]:
```
#Q22 print index 8 from myTuple
print(myTuple[8])
```

```
Sowji
```

In [38]:
```
#Q23 Get the 4th element and 4th element from last of a myTuple
print(myTuple[3])
print(myTuple[-4])
```

```
30
2021
```

In [39]:
```
#Q24 Find how many times 27 appeared in the tuple [Hint: Use method count()]
myTuple.count(27)
```

Out[39]:
```
0
```

In [ ]:
```

```