In [2]:

```
#create a tuple from multiple data types
tuple1 = ("tuple", False, 3.2, 1)
```

In [3]:

```
print(tuple1)
```

```
('tuple', False, 3.2, 1)
```

In [4]:

```
a = ['tuple','dict','try']
b = [2.3,4.5,6.7]
c = [1,2,3]
d = [True, False, True]
```

In [5]:

```
tuple2 = (a,b,c,d)
```

In [6]:

```
print(tuple2)
```

```
(['tuple', 'dict', 'try'], [2.3, 4.5, 6.7], [1, 2, 3], [True, False, Tru
e])
```

In [1]:

```
# Create a list of length 3 and Create a list of length 4, For each element in the firs
t list,
# Display the corresponding index element of the second list
```

In [6]:

```
#Solution
# Create a list of length 3:
header = ['data science', 'Blue mix', 'algorithms']

# Create a list of length 4:
match = ['Red hat', 'Blue mixx','Green gold','Orange Ai']
```

In [7]:

```
# For each element in the first list,
for header, match in zip(header, match):
    # Display the corresponding index element of the second list:
    print(header, 'has the following options:', match)
```

```
data science has the following options: Red hat
Blue mix has the following options: Blue mixx
algorithms has the following options: Green gold
```

In [11]:

```
# display an example of Nested For Loops Using List Comprehension
# Create a list of length 3:
headers = ['data science', 'Blue mix', 'algorithms']

# Create a list of length 4:
matches = ['Red hat', 'Blue mixx','Green gold','Orange Ai']
```

In [12]:

```
#Solution
[(header, match) for header in headers for match in matches ]
```

Out[12]:

```
[('data science', 'Red hat'),
 ('data science', 'Blue mixx'),
 ('data science', 'Green gold'),
 ('data science', 'Orange Ai'),
 ('Blue mix', 'Red hat'),
 ('Blue mix', 'Blue mixx'),
 ('Blue mix', 'Green gold'),
 ('Blue mix', 'Orange Ai'),
 ('algorithms', 'Red hat'),
 ('algorithms', 'Blue mixx'),
 ('algorithms', 'Green gold'),
 ('algorithms', 'Orange Ai')]
```

In [13]:

```
#write a program for breaking and exiting out of loop
# Create a list:
armies = ['Red Army', 'Blue Army', 'Green Army']
```

In [14]:

```
for army in armies:
    print(army)
    if army == 'Blue Army':
        print('Blue Army Found! Stopping.')
        break
```

```
Red Army
Blue Army
Blue Army Found! Stopping.
```

In [15]:

```
#A loop will exit when completed, but using an else statement we can add an action at t
he conclusion of the loop if it hasn't been exited earlier.
```

In [16]:

```
for army in armies:
    print(army)
    if army == 'Orange Army':
        break
else:
    print('Looped Through The Whole List, No Orange Army Found')
```

```
Red Army
Blue Army
Green Army
Looped Through The Whole List, No Orange Army Found
```

In [ ]:

```

```