1. What is the output of the following program?

In [1]:
```
str1 = '{2}, {1} and {0}'.format('a', 'b', 'c')
str2 = '{0}{1}{0}'.format('abra', 'cad')
print(str1, str2)
```

```
c, b and a abracadabra
```

The format() method formats the specified value(s) and insert them inside the string's placeholder.\ We can pass a tuple as paraeter and numbers wrapped in curly brackets '{}' in the string represents index of the tuple elements.

2. What is the output of the following program?

In [2]:
```
line1 = "And Then There Were None"
line2 = "Famous In Love"
line3 = "Famous Were The Kol And Klaus"
line4 = line1 + line2 + line3
print(str.find(line1, 'Were'), str.count((line4), 'And'))
```

```
15 2
```

The find() method finds the first occurrence of the specified value. \ The count() method returns the number of times a specified value appears in the string.

3. What is the output of the following program?

In [3]:
```
my_list = ['hello', 'world']
for i in my_list:
    my_list.append(i.upper())
print(my_list)
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-3-5ed1ebc3a0f5> in <module>
      1 my_list = ['hello', 'world']
      2 for i in my_list:
----> 3     my_list.append(i.upper())
      4 print(my_list)

KeyboardInterrupt:
```

This would go on forever as it is iterating over a continuously expanding list. Here with every iteration the list gets one more element appended which adds one more iteration to the for loop.

4. What is the output of the following program?

In [7]:
```
i = 1
while True:
    if i % 7 == 0:
        break
    print(i)
    i += 1
```

```
1
2
3
4
5
6
```

This while loop goes on until the if statement becomes true. That is why on the 7th iteration when i = 7, it finds that 7 % 7 is 0 and breaks the loop.

5. What is the output of the following program?

In [8]:
```
x = 123
for i in x:
    print(i)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-8-512506850552> in <module>
      1 x = 123
----> 2 for i in x:
      3     print(i)

TypeError: 'int' object is not iterable
```

Python cannot iterate over int objects.

6. What is the output of the following program?

In [9]:
```
for i in [1, 2, 3, 4][::-1]:
    print (i)
```

```
4
3
2
1
```

[::-1] basically reverses the original list. That is why the elements of the original list are being in reverse order.

7. Write down any 5 built-in methods in Python.

1. upper() - used for setting all characters in uppercase in a string
2. lower() - used for setting all characters in lowercase in a string
3. title() - used for setting all words in title case in a string
4. find() - used for searching the string for a specified value and returns the position of where it was found
5. append() - used for adding an element at the end of the list

In [13]:
```
"Mir".upper()
```

Out[13]:
```
'MIR'
```

In [14]:
```
"MiR".lower()
```

Out[14]:
```
'mir'
```

In [15]:
```
"mIr ahMed".title()
```

Out[15]:
```
'Mir Ahmed'
```

In [17]:
```
"mIr ahMed".find('I')
```

Out[17]:
```
1
```

In [19]:
```
my_list = ["Mir"]
my_list.append("Ahmed")
print(my_list)
```

```
['Mir', 'Ahmed']
```

8. Is tuple immutable? Justify your answer with an example.

In [21]:
```
my_tuple = ('Mir', 'ahmed')
my_tuple[1] = 'Ahmed'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-21-1a7b0053427c> in <module>
      1 my_tuple = ('Mir', 'ahmed')
----> 2 my_tuple[1] = 'Ahmed'

TypeError: 'tuple' object does not support item assignment
```

Tuple does not allow any alteration to the elements, hence there are no methods for tuple execpt count() and index().

9. List 5 differences between list and tuple.

1. List is mutable and tuple is immutable
2. List has many built in methods where tuple does not.
3. List consumes more memory than tuple.
4. List is preferred when elements can change or new elements are being added over time, whereas tuple is preferred when elements do not change over time.
5. Implication of iterating over list elements is more time consuming than over tuple elements.

10. Which of the following is false statement in Python?

In [25]:
```
int(144)==144
```

Out[25]:
```
True
```

In [27]:
```
int('144')==144
```

Out[27]:
```
True
```

In [28]:
```
int(144.0)==144
```

Out[28]:
```
True
```

None of the above is false statement.

11. What will be the output of the following code?

In [29]:
```
print(type(1/2))
```

```
<class 'float'>
```

12. What will be the output of the following Python code snippet?

In [31]:
```
print('for'.isidentifier())
```

```
True
```

The isidentifier() method returns True if the string is a valid identifier, otherwise False.\ \ A string is considered a valid identifier if it only contains alphanumeric letters (a-z) and (0-9), or underscores (_). A valid identifier cannot start with a number, or contain any spaces.

13. What will be the output of the following code?

In [32]:
```
veggies = ['carrot', 'broccoli', 'potato', 'asparagus']
veggies.insert(veggies.index('broccoli'), 'celery')
print(veggies)
```

```
['carrot', 'celery', 'broccoli', 'potato', 'asparagus']
```

veggies.index('broccoli') returns 1 (index of the element) and veggies.insert() will insert the element 'celery' on position (index) 1.