Assignment - 3 Python

Exercises

Answer the questions or complete the tasks outlined in bold below, use the specific method described if applicable.

What is 7 to the power of 4?

In [1]:
```
7**4
```

Out[1]:
```
2401
```

Split this string:

```
s = "Hi there Sam!"
```

into a list.

In [2]:
```
s="Hi there Sam!"
result=s.split(" ")
result
```

Out[2]:
```
['Hi', 'there', 'Sam!']
```

In [3]:
```
result[2]='dad!'
result
```

Out[3]:
```
['Hi', 'there', 'dad!']
```

Given the variables:

```
planet = "Earth"
diameter = 12742
```

Use .format() to print the following string:

```
The diameter of Earth is 12742 kilometers.
```

In [4]:
```
output=("The diameter of {planet} is {diameter} kilometrs.").format(planet = "Earth", di
```

In [5]:
```
output
```

Out[5]:
```
'The diameter of Earth is 12742 kilometrs.'
```

Given this nested list, use indexing to grab the word "hello"

In [6]:
```
lst = [1,2,[3,4],[5,[100,200,['hello']],23,11],1,7]
```

In [7]:
```
lst[3][1][2][0]
```

Out[7]:
```
'hello'
```

Given this nest dictionary grab the word "hello". Be prepared, this will be annoying/tricky

In [8]:
```
d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}
```

In [9]:
```
d['k1'][3]['tricky'][3]['target'][3]
```

Out[9]:
```
'hello'
```

What is the main difference between a tuple and a list?

In [10]:
```
# list is mutable, tuple is immutabe
#list can be represented as [],tuples can be represented as ()
#list allows duplicate elements,tuples doesn't allows duplicate elements
#list can be created use list keyword ,tuple can be created use tuple keyword
```

Create a function that grabs the email website domain from a string in the form:

```
user@domain.com
```

So for example, passing "user@domain.com" would return: domain.com

In [11]:
```
def email(input):
  return input.split('@')[1]
```

In [12]:
```
email("user@domain.com")
```

Out[12]:
```
'domain.com'
```

Create a basic function that returns True if the word 'dog' is contained in the input string. Don't worry about edge cases like a punctuation being attached to the word dog, but do account for capitalization.

In [13]:
```
def dog_true(input):
  return 'dog' in input.lower().split()
```

In [14]:
```
dog_true('Dog make us feel less alone')
```

Out[14]:
```
True
```

Create a function that counts the number of times the word "dog" occurs in a string. Again ignore edge cases.

In [15]:
```
def Dog_count(input):
    c= 0
    for x in input.lower().split():
        if x == 'dog':
            c += 1
    return c
```

In [16]:
```
Dog_count('A dog is the perfect exercise partner,so i love dog')
```

Out[16]: 2

Problem

You are driving a little too fast, and a police officer stops you. Write a function to return one of 3 possible results: "No ticket", "Small ticket", or "Big Ticket". If your speed is 60 or less, the result is "No Ticket". If speed is between 61 and 80 inclusive, the result is "Small Ticket". If speed is 81 or more, the result is "Big Ticket". Unless it is your birthday (encoded as a boolean value in the parameters of the function) -- on your birthday, your speed can be 5 higher in all cases.

In [17]:
```
def caught_speeding(speed, is_birthday):

    if is_birthday:
        speeding = speed - 5
    else:
        speeding = speed

    if speeding > 80:
        return 'Big Ticket'
    elif speeding > 60:
        return 'Small Ticket'
    else:
        return 'No Ticket'
```

In [18]:
```
caught_speeding(85,False)
```

Out[18]: 'Big Ticket'

In [19]:
```
caught_speeding(85,True)
```

Out[19]: 'Small Ticket'

In [20]:
```
caught_speeding(65,True)
```

Out[20]: 'No Ticket'

Create an employee list with basic salary values(at least 5 values for 5 employees) and using a for loop retreive each employee salary and calculate total salary expenditure.

In [21]:
```
employee=[15000,300456,25000,100000,550000]
total=0
print("employees list:")
for i in employee:
  print(i)
  total=total+i
print('total salary expenditure:')
print(total)
```

```
employees list:
15000
300456
25000
100000
550000
total salary expenditure:
990456
```

Create two dictionaries in Python:

First one to contain fields as Empid, Empname, Basicpay

Second dictionary to contain fields as DeptName, DeptId.

Combine both dictionaries.

In [22]:
```
def Merge_dict(First, Second):
    for i in Second.keys():
        First[i]=Second[i]
    return First
First={'Empid':1, 'Empname':'Ram', 'Basicpay':'11 lpa'}
Second={'DeptName':'CSE','DeptId':123}
dict1= Merge_dict(First,Second)
dict1
```

Out[22]:
```
{'Empid': 1,
 'Empname': 'Ram',
 'Basicpay': '11 lpa',
 'DeptName': 'CSE',
 'DeptId': 123}
```