Estimate The Crop Yield Using Data Analytics

ASSIGNMENT-3

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? **

```
7**4
```

```
2401
```

** Split this string:**

```
s = "Hi there PRANAVI!"
```

into a list.

```
s = "Hi there PRANAVI!"
```

```
s = s.split(" ") print(s)
```

```
['Hi', 'there', 'Pranavi!']
```

** Given the variables:**

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

** Use .format() to print the following string: **

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

```
planet = "Earth"
diameter = 12742
print("The diameter of {} is {} kilometers.".format(planet,diameter))
```

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

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

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

```
lst[3][1][2][0]
```

```
{"type":"string"}
```

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

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

```
d["k1"][3]["tricky"][3]["target"][3]
```

```
{"type":"string"}
```

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

```
print("List and Tuple in Python are the classes of Python Data
Structures. The list is dynamic, whereas the tuple has static
characteristics. This means that lists can be modified whereas tuples
cannot be modified, the tuple is faster than the list because of
static in nature")
```

```
List and Tuple in Python are the classes of Python Data Structures.
The list is dynamic, whereas the tuple has static characteristics.
This means that lists can be modified whereas tuples cannot be
modified, the tuple is faster than the list because of static in
nature
```

** 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

```
def email(email):
  return email.split("@")[1]

email_var = "user@domain.com"
print(email(email_var))
```

```
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. **

```
def word_found(word):
  return True if "dog" in str(word) else False



word = "Don't worry about edge cases like a punctuation being attached
to the word dog, but do account for capitalization"
word_found(word)
```

```
True
```

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

```
def word_found_counts(word):
  return word.count("dog")
```

```
word = "Don't worry about edge cases like a punctuation being attached
to the word dog, but do account for capitalization dogs"
word_found_counts(word)
```

```
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.

```
def caught_speeding(speed, is_birthday):

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

    if speeding <= 80 :
        return 'Small Ticket'
    elif speeding <= 60:
        return 'No Ticket'
    else:
        return 'Big Ticket'

print(caught_speeding(81,True))
```

```
Big Ticket
```

```
print(caught_speeding(80,False))
```

```
Small 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.

```
employee = [["srisowmiya",600000],["sona",200000],["sriram",300000],
["suriyaraja",400000],["pranavi",500000]]
total = 0
for i in employee:

  total += i[1]

print(total)
```

```
2000000
```

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.

```
basic_emp = {
    "Empid" : 1,
    "Empname": "Pranavi",
    "Basicpay" : 100000.00
}

second_dict = {
    "DeptName" : "computer science",
    "DeptId" : 2001
}


combine = {
    "Empid" : basic_emp["Empid"],
    "Empname": basic_emp["Empname"],
    "Basicpay" : basic_emp["Basicpay"],
    "DeptName" : second_dict["DeptName"],
    "DeptId" : second_dict["DeptId"]
}

print(combine)
```

```
{'Empid': 1, 'Empname': 'Pranavi', 'Basicpay': 100000.0, 'DeptName':
'computer science', 'DeptId': 2001}
```