

Python Fundamentals Showcase

```
resp_iter = self.stub.GetStatuses(empty_pb2.Empty())
statuses = {}
sync for data in resp_iter:
    status = Status(
        status_id=data.id, name=status_name)
    statuses[status.name] = status
return statuses
```

TEMEL GÖREVLER & ÇÖZÜMLER

Presented By GÜLİZ SAMĞAR

Görev 1:

Verilen değerlerin veri yapılarını inceleyiniz.

```
x = 8  
y = 3.2  
z = 8j + 18  
  
a = "Hello World"  
  
b = True  
  
c = 23 < 22  
  
l = [1, 2, 3, 4]  
  
d = {"Name": "Jake",  
      "Age": 27,  
      "Address": "Downtown"}  
  
t = ("Machine Learning", "Data Science")  
  
s = {"Python", "Machine Learning", "Data Science"}
```

Kod Bloğu:

```
x= 8  
print(type(x))  
  
y= 3.2  
print(type(y))  
  
z = 8j + 18  
print(type(z))  
  
a = "Hello World"  
print(type(a))  
  
b = True  
print(type(b))  
  
c = 23 < 22  
print(type(c))  
  
l = [1, 2, 3, 4]  
print(type(l))  
  
d = {"Name": "Jake",  
      "Age": 27,  
      "Address": "Downtown"}  
print(type(d))  
  
t = ("Machine Learning", "Data Science")  
print(type(t))  
  
s = {"Python", "Machine Learning", "Data Science"}  
print(type(s))
```

Beklenen Çıktı:

```
'int'>  
'float'>  
'complex'>  
'str'>  
'bool'>  
'bool'>  
'list'>  
'dict'>  
'tuple'>  
'set'>
```

Görev 2: Verilen string ifadenin tüm harflerini büyük harfe çeviriniz. Virgül ve nokta yerine boşluk koyunuz, kelime kelime ayıriz.

```
text = "The goal is to turn data into information, and information into insight."
```

Kod Bloğu:

```
text = "The goal is to turn data into information, and information into insight."  
formatted_text= text.upper().replace(",","",).replace(".", " ").split()  
print(formatted_text)
```

Beklenen Çıktı:

```
['THE', 'GOAL', 'IS', 'TO', 'TURN', 'DATA', 'INTO', 'INFORMATION', 'AND', 'INFORMATION', 'INTO',  
'INSIGHT']
```

Görev 3:

Verilen listeye aşağıdaki adımları uygulayınız.

```
lst = ["D", "A", "T", "A", "S", "C", "I", "E", "N", "C", "E"]
```

Adım 1: Verilen listenin eleman sayısına bakınız.

Adım 2: Sıfırıncı ve onuncu indeksteki elemanları çağırınız.

Adım 3: Verilen liste üzerinden ["D", "A", "T", "A"] listesi oluşturunuz.

Adım 4: Sekizinci indeksteki elemanı siliniz.

Adım 5: Yeni bir eleman ekleyiniz.

Adım 6: Sekizinci indekse "N" elemanını tekrar ekleyiniz.

Kod Bloğu:

```
lst = ["D", "A", "T", "A", "S", "C", "I", "E", "N", "C", "E"]

# Adım 1:
len(lst)
print("Adım 1 - Eleman sayısı:", len(lst))

# Adım 2:
print("Adım 2 - 0. indeks:", lst[0])
print("      10. indeks:", lst[10])

# Adım 3:
data_list = lst[0:4]
print("Adım 3 - Yeni liste:", data_list)

# Adım 4:
removed_item = lst.pop(8)
print("Adım 4 - Silinen eleman:", removed_item)
print("      Güncel liste:", lst)

# Adım 5:
lst.append(100)
print("Adım 5 - Yeni eleman eklendi:", lst)

# Adım 6:
lst.insert(8, "N")
print("Adım 6 - Eleman tekrar eklendi:", lst)
```

Beklenen Çıktı:

```
Adım 1 - Eleman sayısı: 11
Adım 2 - 0. indeks: D
      10. indeks: E
Adım 3 - Yeni liste: ['D', 'A', 'T', 'A']
Adım 4 - Silinen eleman: N
      Güncel liste: ['D', 'A', 'T', 'A', 'S', 'C', 'I', 'E', 'C', 'E']
Adım 5 - Yeni eleman eklendi: ['D', 'A', 'T', 'A', 'S', 'C', 'I', 'E', 'C', 'E', 100]
Adım 6 - Eleman tekrar eklendi: ['D', 'A', 'T', 'A', 'S', 'C', 'I', 'E', 'N', 'C', 'E', 100]
```

Görev 4:

Verilen sözlük yapısına aşağıdaki adımları uygulayınız.

```
students = {"Christian": ["America", 18],  
            "Daisy": ["England", 12],  
            "Antonio": ["Spain", 22],  
            "Dante": ["Italy", 25],}
```

Adım 1: Key değerlerine erişiniz.

Adım 2: Value'lara erişiniz.

Adım 3: Daisy key'ine ait 12 değerini 13 olarak güncelleyiniz.

Adım 4: Key değeri Ahmet value değeri [Turkey,24] olan yeni bir değer ekleyiniz.

Adım 5: Antonio'yu dictionary'den siliniz.

Kod Bloğu:

```
students = {  
    "Christian": ["America", 18],  
    "Daisy": ["England", 12],  
    "Antonio": ["Spain", 22],  
    "Dante": ["Italy", 25],  
}  
  
# Adım 1:  
keys = students.keys()  
print("Adım 1 - Anahtarlar:", list(keys))  
  
# Adım 2:  
values = students.values()  
print("Adım 2 - Değerler:", list(values))  
  
# Adım 3:  
students["Daisy"][1] = 13  
print("Adım 3 - Güncel Daisy değeri:", students["Daisy"])  
  
# Adım 4:  
students["Ahmet"] = ["Turkey", 24]  
print("Adım 4 - Ahmet eklendi:", students)  
  
# Adım 5:  
students.pop("Antonio")  
print("Adım 5 - Antonio silindi:", students)
```

Beklenen Çıktı:

```
Adım 1 - Anahtarlar: ['Christian', 'Daisy', 'Antonio', 'Dante']  
Adım 2 - Değerler: [['America', 18], ['England', 12], ['Spain', 22], ['Italy', 25]]  
Adım 3 - Güncel Daisy değeri: ['England', 13]  
Adım 4 - Ahmet eklendi: {'Christian': ['America', 18], 'Daisy': ['England', 13], 'Antonio': ['Spain', 22], 'Dante': ['Italy', 25], 'Ahmet': ['Turkey', 24]}  
Adım 5 - Antonio silindi: {'Christian': ['America', 18], 'Daisy': ['England', 13], 'Dante': ['Italy', 25], 'Ahmet': ['Turkey', 24]}
```

Görev 5: Argüman olarak bir liste alan, listenin içerisindeki tek ve çift sayıları ayrı listelere atayan ve bu listeleri return eden fonksiyon yazınız.

Kod Blogu:

```
def tek_cift_ayir(list):
    """Girilen listenin tek ve çift elemanlarını ayırır."""
    ciftler=[]
    tekler=[]
    for index in list:
        if index %2 == 0:
            ciftler.append(index)
        else:
            tekler.append(index)
    return ciftler,tekler

sayi_listesi = [2,13,18,93,22]
cift, tek =  tek_cift_ayir(sayı_listesi)
print("Çiftler:", cift)
print("Tekler:", tek)
```

Beklenen Çıktı:

```
Çiftler: [2, 18, 22]
Tekler: [13, 93]
```

Görev 6: Aşağıda verilen listede mühendislik ve tıp fakültelerinde dereceye giren öğrencilerin isimleri bulunmaktadır. Sırasıyla ilk üç öğrenci mühendislik fakültesinin başarı sırasını temsil ederken, son üç öğrenci de tıp fakültesi öğrenci sırasına aittir. Enumerate kullanarak öğrenci derecelerini fakülte özelinde yazdırınız.

```
ogrenciler = ["Ali", "Veli", "Ayşe", "Talat", "Zeynep", "Ece"]
```

Kod Bloğu:

```
ogrenciler = ["Ali", "Veli", "Ayşe", "Talat", "Zeynep", "Ece"]

for index, ogrenci in enumerate(ogrenciler):
    if index < 3:
        index += 1
        print("Mühendislik Fakültesi", index, ". öğrenci: ", ogrenci)
    else:
        index -= 2
        print("Tıp Fakültesi", index, ". öğrenci: ", ogrenci)
```

Beklenen Çıktı:

```
Mühendislik Fakültesi 2 . öğrenci: Veli
Mühendislik Fakültesi 3 . öğrenci: Ayşe
Tıp Fakültesi 1 . öğrenci: Talat
Tıp Fakültesi 2 . öğrenci: Zeynep
Tıp Fakültesi 3 . öğrenci: Ece
```

Görev 7: Aşağıda 3 adet liste verilmiştir. Listelerde sırası ile bir dersin kodu, kredisi ve kontenjan bilgileri yer almaktadır. Zip kullanarak ders bilgilerini anlamlı bir şekilde bastırınız.

```
ders_kodu = ["CMP1005", "PSY1001", "HUK1005", "SEN2204"]
kredi = [3,4,2,4]
kontenjan = [30,75,150,25]
```

Kod Bloğu:

```
ders_kodu = ["CMP1005", "PSY1001", "HUK1005", "SEN2204"]
kredi = [3,4,2,4]
kontenjan = [30,75,150,25]

for kod, kredi, kont in zip(ders_kodu, kredi, kontenjan):
    print(f"Kredisi {kredi} olan {kod} kodlu dersin kontenjanı {kont} kişidir.")
```

Beklenen Çıktı:

```
Kredisi 3 olan CMP1005 kodlu dersin kontenjanı 30 kişidir.
Kredisi 4 olan PSY1001 kodlu dersin kontenjanı 75 kişidir.
Kredisi 2 olan HUK1005 kodlu dersin kontenjanı 150 kişidir.
Kredisi 4 olan SEN2204 kodlu dersin kontenjanı 25 kişidir.
```

Görev 8: Aşağıda 2 adet set verilmiştir. Eğer 1. küme 2. kümeyi kapsıyor ise ortak elemanlarını, eğer kapsamıyor ise 2. kümenin 1. kümeden farkını yazdıracak fonksiyonu tanımlayınız.

```
kume1 = set(["data", "python"])
kume2 = set(["data", "function", "qcut", "lambda", "python", "mieuul"])
```

Kod Bloğu:

```
kume1 = set(["data", "python"])
kume2 = set(["data", "function", "qcut", "lambda", "python", "mieuul"])

def kume_islemi(set1, set2):
    """Kümeler arası kapsayıcılık kontrolü yapar."""
    if set1.issuperset(set2):
        print("Ortak elemanlar:", set1.intersection(set2))
    else:
        print("Farklı elemanlar:", set2.difference(set1))

kume_islemi(kume1, kume2)
```

Beklenen Çıktı:

```
Farklı elemanlar: {'lambda', 'mieuul', 'function', 'qcut'}
```