**Ek. Mini Projeler**

Kitaptaki 230 dan fazla irili ufaklı örnek programların her biri aynı zamanda bir mini projedir. Bu bölümde bu örnek programlar haricinde, günlük yaşamda veya iş mülakatlarında işinize yarayacak küçük hacimli 19 projeye de ayrıca yer verdik.

**Proje-1. PDF Dilimleyici:** Bir PDF dosyasından belli sayfaları alıp (PDF dosyasını dilimleyip) yeni bir PDF dosyası oluşturan “PDF\_Splitter.py” isimli bir programı kodlayınız.

**Çözüm-1.**

#PyPDF2 modülü kurulu değilse öncesinde **pip install PyPDF2** komutu ile kurulmalıdır.

#PyPDF2 modukünün PdfFileReader, PdfFileWriter fonksiyonları import edilir.

from PyPDF2 import PdfFileReader, PdfFileWriter

#Dilimlenecek PDF dosyası yolu ile birlikte yazılır

pdf\_oku = PdfFileReader("HerkesicinPython\_BolumSoruCevap.pdf")

pdf\_yaz = PdfFileWriter()

pages = [3,5,7,9] # PDF dosyasından split edilecek sayfalar

for p in pages:

pdf\_yaz.addPage(pdf\_oku.getPage(p-1)) # Sayfaları al ve yeni pdf'e ekle

with open("yeni\_pdf.pdf", 'wb') as f:

pdf\_yaz.write(f) # pdf\_yaz nesnesinini içeriği f dosyasına yaz

print(f"yeni {f.name} dosyası başarı ile oluşturuldu.")

**Not.** PyPDF2 yerine PyPDF4 modülü de kullanılabilir.

**Proje-2. PDF Birleştirici:** İki PDF dosyasını birleştirip (merge edip) yeni bir PDF dosyası oluşturan “PDF\_Merger.py” isimli bir programı kodlayınız.

**Çözüm-2.**

#PyPDF2 modülü kurulu değilse öncesinde **pip install PyPDF2** komutu ile kurulmalıdır.

#PyPDF2 modülünün ilgili fonksiyonları import edilir.

from PyPDF2 import PdfFileReader, PdfFileWriter, PdfFileMerger

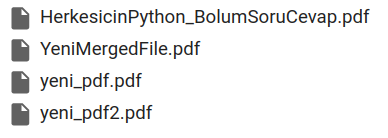
mergeFile = PdfFileMerger()

mergeFile.append(PdfFileReader('yeni\_pdf.pdf', 'rb'))

mergeFile.append(PdfFileReader('yeni\_pdf2.pdf', 'rb'))

mergeFile.write("YeniMergedFile.pdf")

**Not.** Bu iki program sonucunda çalışma klasörü altında aşağıdaki dosyaların oluşması gerekir;



-Resim: Ek1\_proje1.png-

**Proje-3. Password Üreteci:** N elemanlı bir password üreteci (password generator) programını kodlayınız.

**Çözüm-3.**

import string

import random

# Password üretecimiz en az bir büyük harf, bir rakam ve özel karakter # içermeli, gerisi küçük harflerden oluşmalıdır.

N = 8 # karakter sayısı, istenirse dışardan girilebilir.

# programın çalıştırıldığında hep aynı password üretmesi isteniyorsa

random.seed(1) #eklenir

buyuk\_harf = random.sample(string.ascii\_uppercase,1)

ozel\_karakter = random.sample(string.punctuation,1)

rakam = random.sample(string.digits,1)

kucuk\_harf = random.sample(string.ascii\_lowercase,(N-3))

pas\_list = kucuk\_harf + buyuk\_harf + ozel\_karakter + rakam

# şimdi listeyi karıştıralım

random.shuffle(pas\_list)

print("Üretilen password..:" , "".join(pas\_list))

**Proje-4. Metin Seslendirme:** Verilen bir dua metnini sesli okuyan programı kodlayınız.

**Çözüm.**

Salli-Barik dualarını seslendirelim;

**salli\_duasi = Allâhümme salli alâ Muhammedin ve alâ âli Muhammed. Kemâ salleyte alâ İbrahime ve alâ âli İbrahim. İnneke hamidün mecîd.**

### **barik\_duasi : Allâhümme bârik alâ Muhammedin ve alâ âli Muhammed. Kemâ bârekte alâ İbrahime ve alâ âli İbrahim. İnneke hamidün mecîd.**

İki dua da sadece değişen metinleri **dua() i**simli fonksiyona parametre olarak verirsek kod tekrarından kurtulabiliriz. Buna göre;

**Not.** Öncelikle “pip install text\_to\_speech” komutu ile text\_to\_speech modülü kurulmalıdır.

from text\_to\_speech import speak

def dua(s="salli", ss="salleyte"):

dua\_metni = f"Allâhümme {s} alâ Muhammedin ve alâ âli Muhammed. Kemâ {ss} alâ İbrahime ve alâ âli İbrahim. İnneke hamidün mecîd."

return dua\_metni

# main program

print(dua())

speak(dua(), "tr", save=True, file="salli.mp3", speak=True)

speak(dua(s="bârik", ss="bârekte"), "tr", save=True, file="barik.mp3", speak=True)

**Not.** speak() fonksiyonunun "tr" parametresi değiştirilerek farklı dillerde okumalar yapılabilir. Örneğin, İngilizce için "en", Arapça için "ar" yazılabilir.

**Proje-5. İngilizce metni düzeltme:** Verilen bir ingilizce metindeki hatalı kelimeleri düzeltip doğrusunu ekrana yazan programı kodlayınız.

**Çözüm.**

#textblob modülü kurulu değilse öncesinde **pip install textblob** komutu ile kurulmalıdır.

from textblob import TextBlob

**def** Hecele(text):

return list(text.split())

metin = input("İngilizce metni giriniz..:")

kelime\_list = Hecele(metin)

duzeltilmis\_list = []

for i in kelime\_list:

duzeltilmis\_list.append(TextBlob(i))

print("Kelime listesi..:", kelime\_list)

print("Düzeltilmiş liste.:")

for i in duzeltilmis\_list:

print(i.correct(), end=" ")

Programın örnek çıktısı;

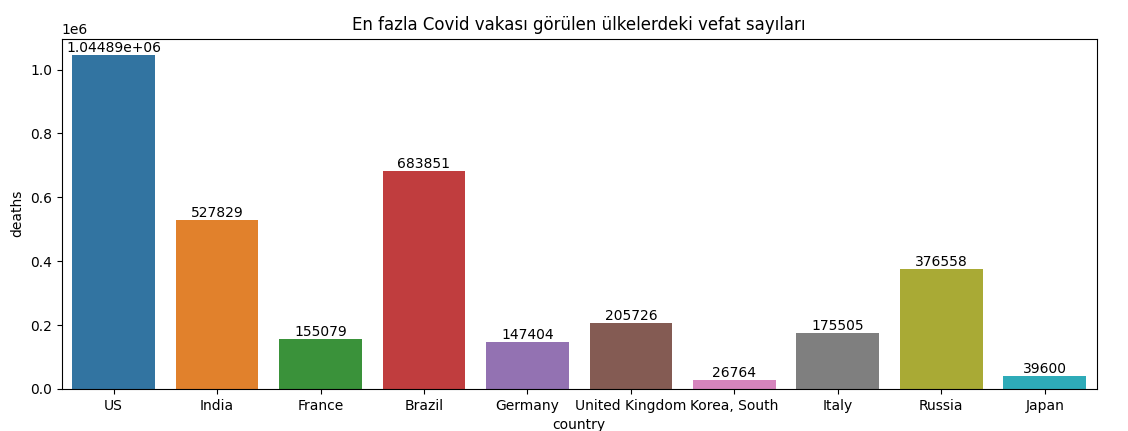
İngilizce metni giriniz..:Python is an eesy to laern, powerfull programmig languag.

Kelime listesi..: ['Python', 'is', 'an', 'eesy', 'to', 'laern,', 'powerfull', 'programmig', 'languag.']

Düzeltilmiş liste.:

Python is an easy to learn, powerful programming language.

**Proje-6. Covid verileri:** En fazla vaka sayısının görüldüğü ilk 10 ülkedeki ölüm sayılarını listeleyen ve çubuk grafiği şeklinde ekranda gösteren programı kodlayınız.



-Resim: Ek\_proje6\_covid.png-

**Çözüm.**

Bu programda hazır ‘covid’ modülüne ait veriler kullanılmıştır.

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**from covid import Covid**

**import pandas as pd**

**cvd = Covid() *# varsayılan veri tabanı seçildi***

**df = pd.DataFrame(cvd.get\_data())**

**top10=df.head(10) *# güncel ilk 10 ülkeyi listele***

**print(top10, sep = None)**

**ulke\_ad = top10.country *# ülke adları***

**vaka\_sa = top10.confirmed *# vaka sayıları***

**vefat\_sa = top10.deaths *# vefat sayıları***

**s\_plot = sns.barplot(x=ulke\_ad, y=vefat\_sa, data=top10)**

**plt.bar\_label(s\_plot.containers[0]) *# vefat sayısını yaz***

**plt.title("En fazla Covid vakası görülen ülkelerdeki vefat sayıları")**

**plt.show()**

**Proje-7: Grafik tabanlı İngilizce-Türkçe Sözlük:** Aşağıdaki gibi “sozluk.csv” isimli sözlük dosyasında tutulan verileri her bir sütunu ayrı listbox da gösteren bir programı kodlayınız.

| **sozluk.csv dosyası** | **Programın ekran çıktısı** |
| --- | --- |
|  |  |

-Resim: Ek\_proje7.png-

**Çözüm.**

**import csv**

**import tkinter as tk**

**root = tk.Tk()**

**sozluk = {} *# dict veri yapısı***

**root.title("Sözlük Programı")**

**with open('/home/bull/py\_projects/sozluk.csv') as f:**

**reader = csv.reader(f, delimiter=',')**

**for eng, tur in reader:**

**sozluk[eng] = tur *# dosyadan oku, sozluk değişkenine at***

**print(sozluk)**

***#listbox'lar tanımlanıyor***

**list1 = tk.Listbox(root)**

**list2 = tk.Listbox(root)**

**list1.grid(row=0, column=0)**

**list2.grid(row=0, column=1)**

***#sözlük yapısındaki key, value değerlerini ayrı listelere ekle***

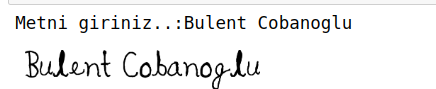
**for key, value in sozluk.items():**

**list1.insert('end', key)**

**list2.insert('end', value)**

**tk.mainloop()**

**Proje-8: Girilen metni el yazısına dönüştürelim:** Aşağıdaki gibi girilen bir metni el yazısına dönüştüren programı kodlayınız. (Not. Türkçe karakter girmeyiniz…)

****

-Resim: Ek\_proje8.png-

**Çözüm.**

Öncesinde “python3 -m pip install pywhatkit” komut satırı ile **pywhatkit** modülünü kurmayı unutmayınız!

**import pywhatkit as kit**

**from PIL import Image**

**text = input("Metni giriniz..:")**

***#Girilen metni el yazısına dönüştür, resim olarak kaydet***

**kit.text\_to\_handwriting(text, save\_to="text\_el.png")**

***#resmi aç***

**Image.open("text\_el.png")**

**Proje-9. Resim dosyasını ASCII karakter setine dönüştürme:** Bir resim dosyasını (png uzantılı) ASCII karakter setine dönüştüren programı kodlayınız.

****

-Resim: Ek\_proje9.png-

**Çözüm.**

**from PIL import Image**

**Image.open("panda.png")**

**import pywhatkit as kit**

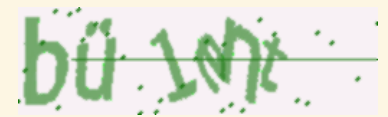
**kit.image\_to\_ascii\_art("panda.png", "asc\_at")**

**asc\_dosya = open("asc\_at.txt", "r")**

**print(asc\_dosya.read())**

**Not.** **pywhatkit** modülü (<https://pypi.org/project/pywhatkit/>) ile aşağıdaki işlemleri gerçekleştirebilirsiniz;

* WhatsApp gruplarına mesaj ve resim gönderebilirsiniz;
* Bir resmi ASCII karakter setine dönüştürebilirsiniz;
* Bir yazıyı (stringi) el yazısına dönüştürebilirsiniz;
* YouTube videolarını oynatabilirsiniz



**Proje-10. CAPTCHA Üreteci:** Sözcük listesindeki kelimelerden herhangi birini seçerek CAPTCHA görseline dönüştüren bir programı kodlayınız. (-Resim: Ek\_proje10.png-)

**Çözüm.**

**CAPTCHA (C**ompletely **A**utomated **P**ublic[**T**uring test](https://tr.wikipedia.org/wiki/Turing_testi) to tell **C**omputers and **H**umans **A**part**)** [Carnegie Mellon](https://tr.wikipedia.org/w/index.php?title=Carnegie_Mellon_School_of_Computer_Science&action=edit&redlink=1) Üniversitesi tarafından geliştirilen özellikle web ortamında [bilgisayar](https://tr.wikipedia.org/wiki/Bilgisayar) ile insan davranışlarının ayırt edilmesini hedefleyen bir projedir. Projenin amacı insan tarafından okunabilecek ancak bilgisayar programları tarafından okunamayacak bir sözcük oluşturmaktır.

* Öncesinde “python3 -m ***pip install captcha***” komut satırı ile **captcha** modülünü kurmayı unutmayınız!

*# CAPTCHA Üreteci*

import random

from captcha.image import ImageCaptcha

liste = ["bülent", "python", "bzaman", "1453Fatih"]

image = ImageCaptcha(width = 250, height = 75)

*#listeden rastgele birini seç*

cpth = random.choice(liste)

data = image.generate(cpth)

image.write(cpth, 'out.png')

*#oluşturulan resmi ekranda göster*

from PIL import Image

Image.open('out.png')

**Proje-11. En büyük ikinci elemanı bulma:** Sırasız halde verilen bir tamsayı listesindeki en büyük ikinci elemanı bulan programı kodlayınız.

**Çözüm.**

Bu ve benzeri sıralama problemleri yazılımcı mülakatlarında sıklıkla sorulan sorulardan, bu tip sorulardaki temel yaklaşımımız öncelikle verileri sıralamak sonrasında istenen elemana erişmek şeklinde olmalıdır.

liste = [3,5,8,44,6,1,34,12,39,65]

s\_list = sorted(liste) *# Sırala*

print(s\_list[-2]) *# Sondan ikinci elemanı al*

**Proje-12. Bir listenin ilk ve son elemanını yer değiştirme:** Sırasız halde verilen bir listenin ilk ve son elemanlarını yer değiştiren programı kodlayınız.

**Çözüm.**

Yapmamız gereken, verilen listenin yeni bir kopyasını oluşturduktan sonra ilk, yani 0. indisli elemanı ile son, yani -1. indisli elemanı yer değiştirmektir.

liste = ['one', 3, 4, 5, 6, 8, 44, 9, 2, 1]

new\_list = liste

*#yeni listenin ilk (o. indisli) elemanı ile son (-1.indisli) elemanı yer değiştirdi.*

new\_list[0], new\_list[-1] = new\_list[-1], new\_list[0]

print(new\_list)

**Proje-13. Bir string içerisindeki özel karakterleri (**punctuations**) silme:** Bir string içerisindeki özel (noktalama) karakterleri silen/kaldıran programı yazınız.

**Çözüm.**

Python dili string işlemlerini, **string**  modülüne (*built-in bir modüldür*) ait aşağıdaki string sabitleri kullanarak kolaylıkla yapabilirsiniz;

* **ascii\_letters:** Tüm ASCII karakterleri ([ascii\_lowercase](https://docs.python.org/3/library/string.html#string.ascii_lowercase) ve [ascii\_uppercase](https://docs.python.org/3/library/string.html#string.ascii_uppercase)) içeren bir string döndürür.
* **ascii\_lowercase:** Küçük harfleri 'abcdefghijklmnopqrstuvwxyz' içeren bir string döndürür.
* **ascii\_uppercase:** Büyük harfleri 'ABCDEFGHIJKLMNOPQRSTUVWXYZ' içeren bir string döndürür.
* **digits:** Rakamları '0123456789' içeren bir string döndürür.
* **hexdigits:** Onaltılık (hexadecimal) sayı sistemindeki rakamları '0123456789abcdefABCDEF' içeren bir string döndürür.
* **octdigits:** Onaltılık (hexadecimal) sayı sistemindeki rakamları '01234567' içeren bir string döndürür.
* **punctuation:** Noktalama (özel) karakterleri **‘**!"#$%&'()\*+,-./:;<=>?@[\]^\_`{|}~.’ içeren bir string döndürü.
* **printable:** Tüm ekrana basılabilen karakterleri ([digits](https://docs.python.org/3/library/string.html#string.digits), [ascii\_letters](https://docs.python.org/3/library/string.html#string.ascii_letters), [punctuation](https://docs.python.org/3/library/string.html#string.punctuation) ve [whitespace](https://docs.python.org/3/library/string.html#string.whitespace)) içeren bir string döndürür.
* **whitespace:** Ekrana basılamayan ‘space, tab, linefeed, return, formfeed ( \t\n\r\x0b\x0c) gibi’ boşluk karakterlerini içeren bir string döndürür.

Bu bilgiler ışığında bizden istenen programı aşağıdaki gibi kodlayabiliriz;

import string

my\_str = "python dersi! çok kolay, değil- mi?"

new\_str = ""

for ch in my\_str:

if ch not in string.**punctuation:**

new\_str = new\_str + ch

print(new\_str)

Programın ekran çıktısında aşağıdaki gibi punctation karakterlerinin silindiği görülecektir;

python dersi çok kolay değil mi

**Proje-14. Dijital Kronometre:** İç içe döngüleri ve time.sleep() metodunu kullanarak dijital bir kronometreyi ‘Saat:Dakika:Saniye - **00:01:58**’ şeklinde ekranda gösteren programı yazınız.

**Çözüm.**

**import time**

**for saat in range(24):**

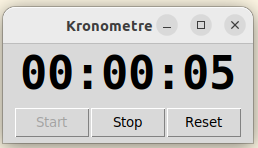
**for dakika in range(60):**

**for saniye in range(60):**

*#Aynı ekrana satır başı yaparak yazdırmak için end='\r' kullanıldı.*

**print(f"{saat:0>2d}:{dakika:0>2d}:{saniye:0>2d}", end='\r')**

**time.sleep(1)** *#1 saniye aralıklarla yaz*



**Not.** Bu programın GUI tabanlı farklı bir sürümünü [**https://github.com/bcobanoglu/HerkesIcinPython/blob/master/kronometre.py**](https://github.com/bcobanoglu/HerkesIcinPython/blob/master/kronometre.py)adresinden indirebilirsiniz. (Resim: Ek1\_proje\_14.png)

**Proje-15: FIFO tabanlı hayvan barınağı tasarımı:** Niksar hayvan barınağında sadece kedi ve köpekler barınmaktadır. Ve bu barınak "**İlk Giren İlk Çıkar** - **FIFO**" esasına göre çalışmaktadır. Yani insanlar barınaktaki hayvanların "en yaşlısını" (varış veya ekleme zamanına göre) sahiplenebilir. Köpek yada kedi tercih edecekler de yine yalnızca o türdeki en yaşlı hayvanı seçebilirler. İstedikleri hayvanı seçemezler. Bu sistemi sürdürmek için uygun veri yapısını oluşturup, aşağıdaki fonksiyonları uygulayın.

- **enqueue()** işlevi; Barınağa(listeye) herhangi bir hayvanı ekler.

- **dequeue\_cat()** işlevi; Barınaktan (listeden) en yaşlı kediyi kaldırır.

- **dequeue\_dog()** işlevi; Barınaktan (listeden) en yaşlı köpeği kaldırır.

**Çözüm.** Bu problemi aşağıdaki gibi listeleri kuyruk veri yapısında tasarlayarak çözebiliriz. Kuyruğa eleman eklemek için “**append**”, kuyruktan eleman çıkarmak için ise "**popleft**" metotları kullanılmaktadır.

Python dilinde kuyruk veri yapısı tanımlamak için “**deque**” sınıfına ait metotlar (**popleft()** gibi) kullanılabilir. Bunun için ‘**from collections import deque’** komut satırını program başına eklemek gerekir.

**from collections import deque**

**cat\_list = deque([]) *#boş kedi kuyruğu***

**dog\_list = deque([]) *#boş köpek kuyruğu***

***#Kuyruğa elemanlar ekleyen fonksiyon***

**def enqueue(hayvan, cins):**

**if cins=="dog":**

**dog\_list.append(hayvan)**

**print(f"Barınağa {hayvan} eklendi")**

**if cins=="cat":**

**cat\_list.append(hayvan)**

**print(f"Barınağa {hayvan} eklendi")**

**def dequeue\_cat():**

**print("Barınaktan çıkarılan kedi.:",cat\_list.popleft())**

**def dequeue\_dog():**

**print("Barınaktan çıkarılan köpek.:", dog\_list.popleft())**

**enqueue("Karabaş", "dog")**

**enqueue("Behlül", "cat")**

**enqueue("Çomar", "dog")**

**enqueue("Pati", "cat")**

**enqueue("Limon", "cat")**

***# Barınaktan en yaşlı kediyi çıkaralım***

**dequeue\_cat()**

***# Barınaktan en yaşlı köpeği çıkaralım***

**dequeue\_dog()**

**print (f"Barınağın Son hali.:\nKediler..:{cat\_list} \nKöpekler..:{dog\_list}")**

**Programın ekran çıktısı;**

**Barınağa Karabaş eklendi**

**Barınağa Behlül eklendi**

**Barınağa Çomar eklendi**

**Barınağa Pati eklendi**

**Barınağa Limon eklendi**

**Barınaktan çıkarılan kedi.: Behlül**

**Barınaktan çıkarılan köpek.: Karabaş**

**Barınağın Son hali.:**

**Kediler..:deque(['Pati', 'Limon'])**

**Köpekler..:deque(['Çomar'])**

**Proje-16: Sıkıştırılmış (Zipli) Dosyayı Açma:**

**Çözüm.**

Sıkıştırılmış bir dosyayı (**"HerkesIcinPython.zip" gibi**) açmak ve içindeki dosya isimlerini ekrana yazdırmak için aşağıdaki programı kullanabilirsiniz;

**from zipfile import ZipFile**

**with ZipFile("HerkesIcinPython.zip", "r") as z:**

**z.extractall() *#dosyayı aç***

**print(z.namelist()) *#dosya isimlerini yaz***

**Not.** zipfile hakkında detaylı bilgi için <https://docs.python.org/3/library/zipfile.html> adresi ziyaret edilebilir. Örneğin birden fazla dosyayı sıkıştırmak için komut isteminden(terminalden) aşağıdaki gibi komut verilebilir;

**$** python3 -m **zipfile -c** zipli\_dosya.zip dosya1.txt dosya2.txt

**Proje-17: Hava durumunu öğrenme:** Girilen bir şehrin hava durumunu veren programı kodlayınız.

**Çözüm.**

**import requests**

**city = input('Şehir adı..:') *# şehir adını gir***

***# city = 'sakarya'***

**print("\*\*\* " + city + " hava durumu \*\*\*")**

***#Hava raporunu alacağımız url adresi;***

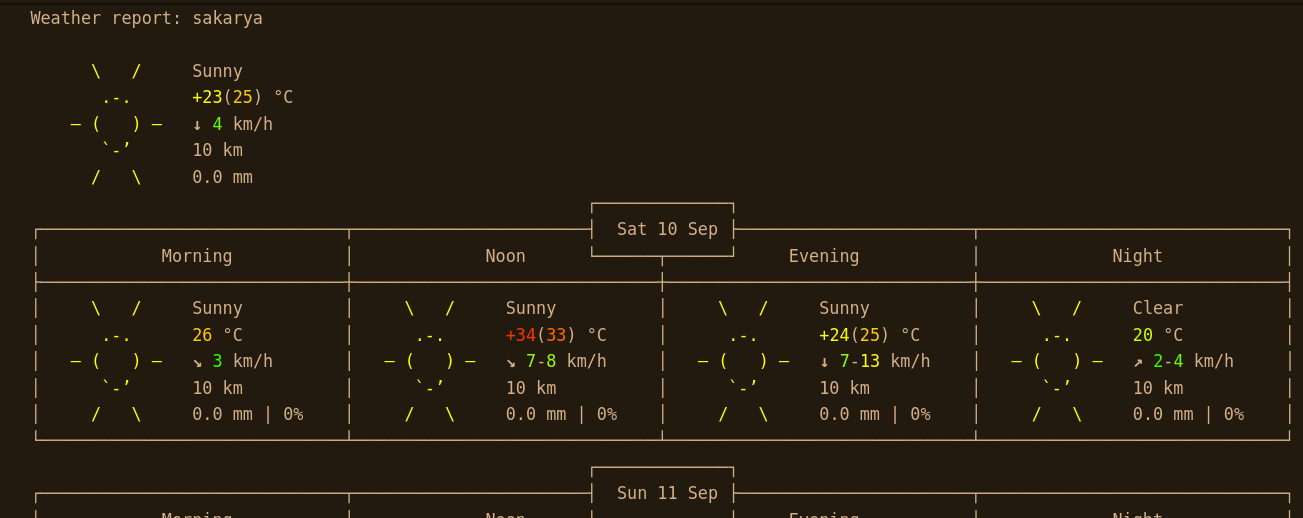
**url = 'https://wttr.in/{}'.format(city)**

**durum = requests.get(url) *# web adresinden veriyi çek***

**print(durum.text) *# text formatında yazdır***

**Açıklama:** Python dilinde herhangi bir API’ye istek göndermek ve onlarla çalışmak için “requests” modülü kullanılabilir.Örneğimizdeki gibi requests modülünü kullanarak bir web adresi içeriğini ‘**requests.get(url)**’ komut satırı ile alabiliriz. Bu modül kurulu değilse öncesinde “python3 -m *pip install* requests” komut satırı ile kurmak gerektiğini unutmayınız!

Programın örnek ekran çıktısı;

****

(Resim: Ek1\_proje\_17.png)

**Proje-18. Seçilen bir klasör içerisindeki ‘.py’ uzantılı dosyaları ekrana yazdırma:** Windows tarzı bir diyalog kutusu ile seçilen klasör altındaki belli dosyaları (.py uzantılı dosyaları) listeleyen programı kodlayınız.

**Çözüm. Öncesinde kurulu değilse PySimpleGUI modülü kurulmalıdır. glob modülü ile eşleşen tüm yol adları listelenir.**

**import PySimpleGUI as gui**

**gui.theme('Dark Blue 3') *# Pencere teması***

**layout = [[gui.Text('Klasör ve Dosya Seç')],**

**[gui.Text('Klasörler',size=(15,1)), gui.InputText(), gui.FolderBrowse()],**

**[gui.OK(), gui.Cancel()]]**

**window = gui.Window('Klasör ve Dosya Adı', layout)**

**olay, yol = window.read()**

**window.close()**

**print(yol[0]) *# seçilen klasörü ekrana yazdır***

**import glob**

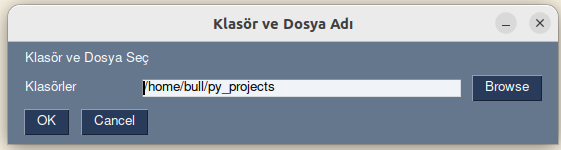
***#klasör yoluna aramak istediğimiz dosya uzantısını ekledik***

**dosya\_yolu = f'{yol[0]}/\*.py'**

**files = glob.glob(dosya\_yolu) *# Eşleşen dosyaları al***

**print(files)**

Programın ekran çıktısında önce diyalog penceresinden ‘Browse’ tuşu ile klasör seçilir, sonra ‘OK’ tuşuna basıldıktan sonra ekranda dosya isimleri listelenir.

**(Ek1\_proje\_18.png)**

**/home/bull/py\_projects**

**['/home/bull/py\_projects/turtle1.py', '/home/bull/py\_projects/text\_correct.py', '/home/bull/py\_projects/kronometre.py', '/home/bull/py\_projects/hangman.py',**

**…**

**]**

**Proje-19: TDK tabanlı Adam Asmaca Oyunu:** TDK sözlüğünden rastgele bir kelime seçerek adam asmaca oyunu oynayan programı kodlayınız.

**Çözüm.** Örnek 19.1 deki adam asmaca oyunundan farklı olarak bu programda aynı zamanda büyük hacimli (TDK sözlüğü gibi) bir txt dosyasından veri çekmeyi, iç içe listeleri **flatten** ile düzleştirmeyi, sample() fonksiyonu ile bir listeden rastgele eleman çekmeyi, üç tırnak''' ile çok satırlı (alt alta) string tanımlamayı göreceğiz.

import random *# gerekli modüller import ediliyor*

from nltk import flatten

import pandas as pd

*# tdk kelime listesini tutan sözlük dosyası*

url = "https://raw.githubusercontent.com/bcobanoglu/HerkesIcinPython/master/tdk\_kelime\_list.txt"

df = pd.read\_csv(url)

liste = flatten(df.sample(10).values.tolist()) *# tdk sözlüğünden 10 kelime al*

print(liste) *# flatten():iç içe listeyi tek listeye dönüştürür*

kelime = random.choice(liste) *# birini seç*

*#her yanlış harfte ekranda gösterilecek 7 elemanlı hangman listesi*

hangman = ['''

+----+

o |

-|- |

/ \ |

\_===\_

''']

dogruHarf = [] *# doğru harf listesi*

yanlisHarf = [] *# yanlış harf (hatalı durum) listesi*

hak = len(kelime) *# seçilen kelimenin karakter uzunluğu kadar hak verilir*

while hak>0: *# hak sıfırdan büyük olduğu sürece*

out="" *# çıktı değişkeni*

for h in kelime: *# kelimenin harfleri üzerinde dolaşalım*

if h in dogruHarf: *# doğru harf ise*

out = out + h *# çıktıya o harfi ekle*

else: *# yanlış harf ise*

out = out + "\_" *# çıktıya \_ ekle*

if out == kelime: *# çıktı, seçilen kelimeye eşitse*

break *# döngüden çık*

print ("Kelime: ", out) *# Mevcut çıktıyı yaz*

*#print (hangman[hak-1]) # listedeki görseli yaz*

girHarf = input() *# harf gir ve girilen harfi karşılaştır*

if girHarf in dogruHarf or girHarf in yanlisHarf:

print ("zaten ", girHarf, "girilmişti!")

elif girHarf in kelime:

print ("Doğru")

dogruHarf.append(girHarf) *# dogruHarf listesine ekle*

else: *# girilen harf kelimede yoksa*

print ("Yanlış")

hak = hak-1 *# hakkı 1 azalt*

yanlisHarf.append(girHarf) *#yanlisHarf listesine ekle*

if hak !=0: *# hak bitmeden çıkıldı ise Tebrikler*

print ("Tebrikler, Evet:", kelime)

else:

print ("Maalesef kelimemiz:", kelime)

print(hangman[0]) *# listedeki ilk elemanı göster*

**Not.** Bu programın kaynak kodunu <https://github.com/bcobanoglu/HerkesIcinPython/blob/master/hangman.py> adresinden indirebilirsiniz.