This is the template for the image recognition exercise.
Some **general instructions**, read these carefully:

- The final assignment is returned as a clear and understandable *report*
  - define shortly the concepts and explain the phases you use
  - use the Markdown feature of the notebook for larger explanations
- return your output as a *working* Jupyter notebook
- name your file as Exercise_MLPR2023_Partx_uuid.jpynb
  - use the uuid code determined below
  - use this same code for each part of the assignment
- write easily readable code with comments
  - if you exploit code from web, provide a reference
- it is ok to discuss with a friend about the assignment. But it is not ok to copy someone's work. Everyone should submit their own implementation
  - in case of identical submissions, both submissions are failed

Deadlines:

- Part 1: Mon 6.2 at 23:59
- Part 2: Mon 20.2 at 23:59
- Part 3: Mon 6.3 at 23:59

No extensions for the deadlines

- after each deadline, example results are given, and it is not possible to submit anymore

If you encounter problems, Google first and if you can’t find an answer, ask for help

- Moodle area for questions
- pekavir@utu.fi
- teacher available for questions
  - Monday 30.1 at 14:00-15:00 room 407B Honka (Agora 4th floor)
  - Monday 13.2 at 14:00-15:00 room 407B Honka (Agora 4th floor)
  - Thursday 2.3 at lecture 10:15-12:00

Grading

The exercise covers a part of the grading in this course. The course exam has 5 questions, 6 points of each. Exercise gives 6 points, i.e. the total score is 36 points.

From the template below, you can see how many exercise points can be acquired from each task. Exam points are given according to the table below:

7 exercise points: 1 exam point
8 exercise points: 2 exam points

9 exercise points: 3 exam points
10 exercise points: 4 exam points
11 exercise points: 5 exam points
12 exercise points: 6 exam points

To pass the exercise, you need at least 7 exercise points, and at least 1 exercise point from each Part.
Each student will grade one submission from a peer and their own submission. After each Part deadline, example results are given. Study them carefully and perform the grading according to the given instructions. Mean value from the peer grading and self-grading is used for the final points.

In [ ]:
```
# import uuid
# # Run this cell only once and save the code. Use the same id code for e
# # Printing random id using uuid1()
# print ("The id code is: ",end="")
# print (uuid.uuid1())
```

The id code is: bc75471a-a096-11ed-8824-3dc691ec23e7

Part 1

In [ ]:
```

```

Read the original research article:

İ. Çınar and M. Koklu. Identification of rice varieties using machine learning algorithms. Journal of Agricultural Sciences, 28(2):307–325, 2022. doi: 10.15832/ankutbd.862482.

https://dergipark.org.tr/en/download/article-file/1513632

Introduction (1 p)

Will be written in Part 3

Preparations of the data (1 p)

Make three folders in your working folder: "notebooks", "data" and "training_data". Save this notebook in "notebooks" folder.

Perform preparations for the data

- import all the packages needed for this notebook in one cell
- import the images. Data can be found from (downloading starts as you press the link) https://www.muratkoklu.com/datasets/vtdhnd09.php
    - save the data folders "Arborio", "Basmati" and "Jasmine" in "data" folder

- take a random sample of 100 images from Arborio, Basmati and Jasmine rice species (i.e. 300 images in total)
- determine the contour of each rice (you can use e.g. *findContours* from OpenCV)
- plot one example image of each rice species, including the contour

Feature extraction (2 p)

Gather the feature data

Color features (15)
- Calculate the following color features for each image, including only the pixels within the contour (you can use e.g. *pointPolygonTest* from OpenCV) - Mean for each RGB color channel - Variance for each RGB color channel - Skewness for each RGB color channel - Kurtosis for each RGB color channel - Entropy for each RGB color channel
Dimension features (6)

- Fit an ellipse to the contour points (you can use e.g. *fitEllipse* from OpenCV)
- Plot one example image of each rice species including the fitted ellipse
- Calculate the following features for each image (for details, see the original article)
  - the major axis length the ellipse
  - the minor axis length of the ellipse
  - area inside the contour (you can use e.g. *contourArea* from OpenCV)
  - perimeter of the contour (you can use e.g. *arcLength* from OpenCV)
  - roundness
  - aspect ratio

Gather all the features in one array or dataframe: one data point in one row, including all feature values in columns.
For each data point, include also information of the original image and the label (rice species). Save the data in "training_data" folder.

Part # 1

In [ ]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import matplotlib.image as mimg
import os, os.path as path
from scipy.stats import kurtosis, skew, entropy

import cv2
import math
```

In [ ]:
```
data_path = "../data"
training_data_path = "../training_data/"
imgs = "../imgs/"
```

```
if not path.exists(data_path):
    os.mkdir(data_path)
if not path.exists(training_data_path):
    os.mkdir(training_data_path)
if not path.exists(imgs):
    os.mkdir(imgs)
```

In [ ]:
```
def random_data(data_path):
    data = []
    for folder in os.listdir(data_path):
        data.extend([{"path": path.join(data_path, folder, file), "label"
                    for file in np.random.choice(os.listdir(f'{data_path}
    df = pd.DataFrame(data)
    return df
```

In [ ]:
```
rice_df = random_data(data_path)
```

In [ ]:
```
rice_df.sample(10)
```

Out[ ]:

	path	label
134	../data/Jasmine/Jasmine (7106).jpg	Jasmine
202	../data/Arborio/Arborio (5515).jpg	Arborio
91	../data/Basmati/basmati (2441).jpg	Basmati
184	../data/Jasmine/Jasmine (11063).jpg	Jasmine
30	../data/Basmati/basmati (3083).jpg	Basmati
275	../data/Arborio/Arborio (4534).jpg	Arborio
295	../data/Arborio/Arborio (13732).jpg	Arborio
23	../data/Basmati/basmati (9581).jpg	Basmati
49	../data/Basmati/basmati (7531).jpg	Basmati
137	../data/Jasmine/Jasmine (2254).jpg	Jasmine

Color Features

In [ ]:
```
for index, row in rice_df.iterrows():

    image = cv2.imread(row['path'])
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    ret, thresh = cv2.threshold(gray_image, 125, 255, 0)
    contours, hierarchy = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHA

    R,G,B = [], [], []
    for x in range(image.shape[0]):
        for y in range(image.shape[1]):
            # Check if the pixel falls inside the contour
            if cv2.pointPolygonTest(contours[0], (y,x), False) > 0:
                R.append(image[y,x,0])
                G.append(image[y,x,1])
                B.append(image[y,x,2])

    R = np.array(R)
```

```
    G = np.array(G)
    B = np.array(B)

    rice_df.at[index, 'RGB_R_mean'] = np.mean(R)
    rice_df.at[index, 'RGB_G_mean'] = np.mean(G)
    rice_df.at[index, 'RGB_B_mean'] = np.mean(B)

    rice_df.at[index, 'RGB_R_kurtosis'] = kurtosis(R)
    rice_df.at[index, 'RGB_G_kurtosis'] = kurtosis(G)
    rice_df.at[index, 'RGB_B_kurtosis'] = kurtosis(B)

    rice_df.at[index, 'RGB_R_skew'] = skew(R)
    rice_df.at[index, 'RGB_G_skew'] = skew(G)
    rice_df.at[index, 'RGB_B_skew'] = skew(B)

    rice_df.at[index, 'RGB_R_entropy'] = entropy(R)
    rice_df.at[index, 'RGB_G_entropy'] = entropy(G)
    rice_df.at[index, 'RGB_B_entropy'] = entropy(B)

    rice_df.at[index, 'RGB_R_var'] = np.var(R)
    rice_df.at[index, 'RGB_G_var'] = np.var(G)
    rice_df.at[index, 'RGB_B_var'] = np.var(B)
```

Creating Contours and Dimension Features

In [ ]:

```
for index, row in rice_df.iterrows():
    img_path = row["path"]

    img = cv2.imread(img_path)

    img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    ret, thresh = cv2.threshold(img_gray, 127, 255, 0)

    contours, hierarchy = cv2.findContours(
        thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    cv2.drawContours(img, contours, -1, (0, 255, 0), 3)

    ellipse = cv2.fitEllipse(contours[0])

    cv2.ellipse(img, ellipse, (0, 0, 255), 3)

    (xc, yc), (d1, d2), angle = ellipse

    major_axis_ellipse = max(d1, d2)
    minor_axis_ellipse = min(d1, d2)

    moments = cv2.moments(contours[0])

    cx = moments['m10']/moments['m00']
    cy = moments['m01']/moments['m00']

    area = cv2.contourArea(contours[0])

    x,y,w,h = cv2.boundingRect(contours[0])
    aspect_ratio = float(w)/h
```

```
    perimeter = cv2.arcLength(contours[0], True)

    roundness = (4 * area * math.pi) / perimeter ** 2

    img_path = os.path.join(imgs, *(img_path.split(os.path.sep)[3:]))
    cv2.imwrite(img_path, img)

    rice_df.at[index, 'contours'] = img_path
    rice_df.at[index, 'area'] = area
    rice_df.at[index, 'perimeter'] = perimeter
    rice_df.at[index, 'major_axis_ellipse'] = major_axis_ellipse
    rice_df.at[index, 'minor_axis_ellipse'] = minor_axis_ellipse
    rice_df.at[index, 'cx'] = cx
    rice_df.at[index, 'cy'] = cy
    rice_df.at[index, 'aspect_ratio'] = aspect_ratio
```

In [ ]:
```
rice_df.sample(10)
```

Out[ ]:

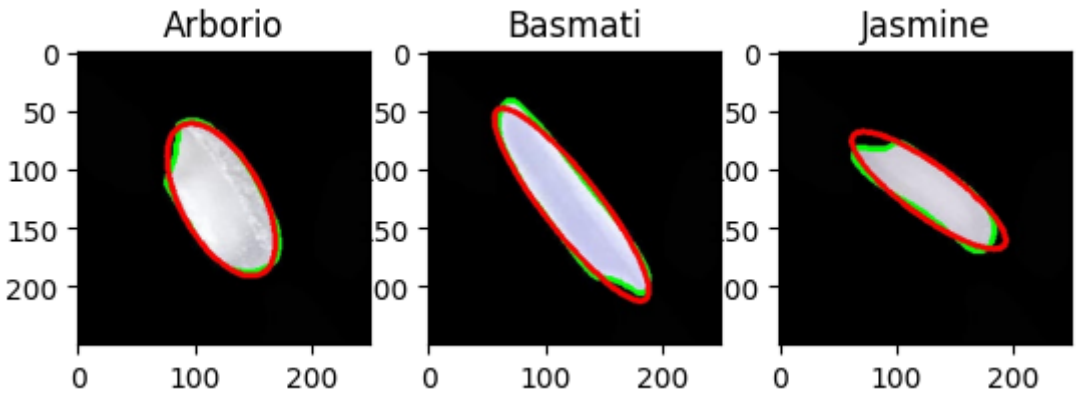
	path	label	RGB_R_mean	RGB_G_mean	RGB_B_mean	RGB_R_ku
233	../data/Arborio/Arborio (274).jpg	Arborio	162.521461	156.823795	157.736822	-1.2
20	../data/Basmati/basmati (9807).jpg	Basmati	85.461737	76.028309	76.202559	-1.5
30	../data/Basmati/basmati (3083).jpg	Basmati	206.346291	181.674498	181.470454	2.2
292	../data/Arborio/Arborio (7939).jpg	Arborio	163.584568	154.915176	155.219596	-1.2
151	../data/Jasmine/Jasmine (9062).jpg	Jasmine	93.876421	91.099560	90.954712	-1.8
18	../data/Basmati/basmati (314).jpg	Basmati	85.378627	84.668527	80.360212	-1.6
179	../data/Jasmine/Jasmine (2932).jpg	Jasmine	146.969605	139.302330	139.450659	-1.5
247	../data/Arborio/Arborio (2282).jpg	Arborio	129.777165	127.341456	126.946513	-1.9
287	../data/Arborio/Arborio (8522).jpg	Arborio	121.482222	118.891614	118.427783	-1.8
184	../data/Jasmine/Jasmine (11063).jpg	Jasmine	116.072690	110.546570	110.561629	-1.9

10 rows × 25 columns

Plots of Rice

In [ ]:
```
samples = rice_df.groupby('label').sample(1).reset_index()
```

In [ ]:
```
f, ax = plt.subplots(1,3)
for i, sample in samples.iterrows():
    img = mimg.imread(sample["contours"])
    ax[i].imshow(img)
    ax[i].set_title(sample["label"])
```



In [ ]:
```
rice_df.to_csv('../training_data/data.csv')
```