In [ ]:
```
from sklearn.ensemble import RandomForestClassifier
import pandas as pd
```

In [ ]:
```
df_train = pd.read_csv("./dataset/my_train_features.csv")
df_test = pd.read_csv("./dataset/my_train_features.csv")
```

In [ ]:
```
df_train.head()
```

Out[ ]:

	letter_slant	line_slant	letter_size	word_spacing	personality
0	backward	upperside	1307.2	small	Agreeableness
1	backward	upperside	932.0	small	Agreeableness
2	forward	upperside	891.4	small	Agreeableness
3	forward	upperside	279.6	small	Agreeableness
4	vertical	upperside	766.0	small	Agreeableness

In [ ]:
```
df_test.head()
```

Out[ ]:

	letter_slant	line_slant	letter_size	word_spacing	personality
0	backward	upperside	1307.2	small	Agreeableness
1	backward	upperside	932.0	small	Agreeableness
2	forward	upperside	891.4	small	Agreeableness
3	forward	upperside	279.6	small	Agreeableness
4	vertical	upperside	766.0	small	Agreeableness

In [ ]:
```
letter_slant_mapping = {'backward': -1, 'forward': 1, 'vertical': 0}
line_slant_mapping = {'lowerside': -1, 'baseline': 0, 'upperside': 1}
word_spacing_mapping = {'small': -1, 'medium': 0, 'large': 1}
```

In [ ]:
```
df_train["letter_slant"] = df_train["letter_slant"].map(letter_slant_mapping)
df_train["line_slant"] = df_train["line_slant"].map(line_slant_mapping)
df_train["word_spacing"] = df_train["word_spacing"].map(word_spacing_mapping)
df_test["letter_slant"] = df_test["letter_slant"].map(letter_slant_mapping)
df_test["line_slant"] = df_test["line_slant"].map(line_slant_mapping)
df_test["word_spacing"] = df_test["word_spacing"].map(word_spacing_mapping)
```

In [ ]:
```
df_train.head()
```

Out[ ]:

	letter_slant	line_slant	letter_size	word_spacing	personality
0	-1	1	1307.2	-1	Agreeableness
1	-1	1	932.0	-1	Agreeableness
2	1	1	891.4	-1	Agreeableness
3	1	1	279.6	-1	Agreeableness
4	0	1	766.0	-1	Agreeableness

In [ ]:
```
print(df_train.info())
print(df_test.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 177 entries, 0 to 176
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   letter_slant  177 non-null    int64  
 1   line_slant    177 non-null    int64  
 2   letter_size   177 non-null    float64
 3   word_spacing  177 non-null    int64  
 4   personality   177 non-null    object 
dtypes: float64(1), int64(3), object(1)
memory usage: 7.0+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 177 entries, 0 to 176
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   letter_slant  177 non-null    int64  
 1   line_slant    177 non-null    int64  
 2   letter_size   177 non-null    float64
 3   word_spacing  177 non-null    int64  
 4   personality   177 non-null    object 
dtypes: float64(1), int64(3), object(1)
memory usage: 7.0+ KB
None
```

In [ ]:
```
df_train.isnull().sum()
```

Out[ ]:
```
letter_slant    0
line_slant      0
letter_size     0
word_spacing    0
personality     0
dtype: int64
```

In [ ]:
```
df_test.isnull().sum()
```

```
Out[ ]: letter_slant    0
        line_slant      0
        letter_size     0
        word_spacing    0
        personality     0
        dtype: int64
```

```
In [ ]: df_train.dropna(inplace=True)
        df_test.dropna(inplace=True)
```

```
In [ ]: print(df_train.info())
        print(df_test.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 177 entries, 0 to 176
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   letter_slant  177 non-null    int64  
 1   line_slant    177 non-null    int64  
 2   letter_size   177 non-null    float64
 3   word_spacing  177 non-null    int64  
 4   personality   177 non-null    object 
dtypes: float64(1), int64(3), object(1)
memory usage: 7.0+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 177 entries, 0 to 176
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   letter_slant  177 non-null    int64  
 1   line_slant    177 non-null    int64  
 2   letter_size   177 non-null    float64
 3   word_spacing  177 non-null    int64  
 4   personality   177 non-null    object 
dtypes: float64(1), int64(3), object(1)
memory usage: 7.0+ KB
None
```

```
In [ ]: x_train = df_train.drop('personality', axis=1)
        y_train = df_train['personality']
        x_test = df_test.drop('personality', axis=1)
        y_test = df_test['personality']
```

```
In [ ]: rf = RandomForestClassifier(n_estimators=100, criterion="gini", random_state=42)
        rf_res = rf.fit(x_train, y_train)
```

```
In [ ]: y_pred = rf.predict(x_test)
        print(y_pred)
```

```
['Agreeableness' 'Agreeableness' 'Agreeableness' 'Agreeableness'
 'Agreeableness' 'Agreeableness' 'Agreeableness' 'Agreeableness'
 'Agreeableness' 'Agreeableness' 'Agreeableness' 'Agreeableness'
 'Agreeableness' 'Agreeableness' 'Agreeableness' 'Agreeableness'
 'Agreeableness' 'Agreeableness' 'Agreeableness' 'Agreeableness'
 'Agreeableness' 'Agreeableness' 'Agreeableness' 'Agreeableness'
 'Agreeableness' 'Agreeableness' 'Agreeableness' 'Agreeableness'
 'Agreeableness' 'Agreeableness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Conscientiousness' 'Conscientiousness'
 'Conscientiousness' 'Extraversion' 'Extraversion' 'Extraversion'
 'Extraversion' 'Extraversion' 'Extraversion' 'Extraversion'
 'Extraversion' 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism'
 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism'
 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism'
 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism'
 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism'
 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism'
 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Neuroticism'
 'Neuroticism' 'Neuroticism' 'Neuroticism' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness'
 'Openness' 'Openness' 'Openness' 'Openness' 'Openness' 'Openness']
```

In [ ]:

```
from sklearn.metrics import accuracy_score, confusion_matrix
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

Out[ ]:

```
1.0
```

In [ ]:

```
rf.feature_importances_
```

Out[ ]:

```
array([0.09260907, 0.0737168 , 0.75604601, 0.07762811])
```

In [ ]:

```
x_test.head(2)
```

Out[ ]:

	letter_slant	line_slant	letter_size	word_spacing
0	-1	1	1307.2	-1
1	-1	1	932.0	-1

In [ ]:

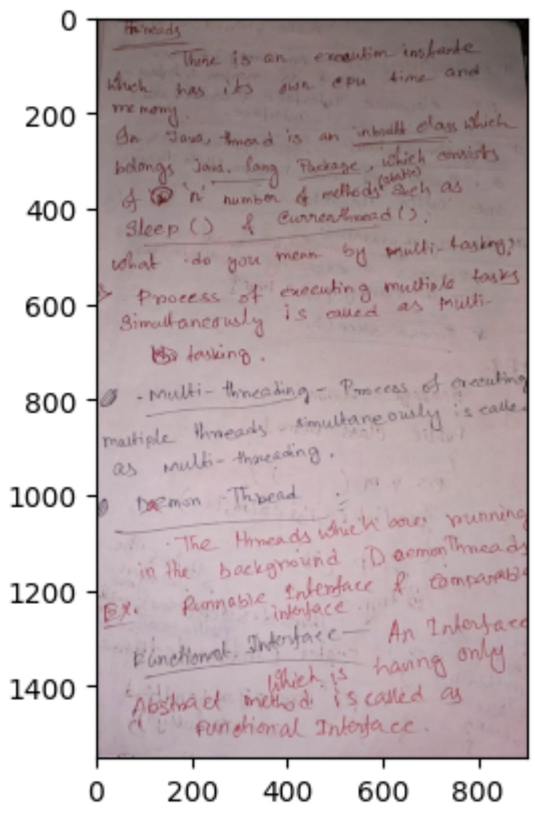
```
from package.features import *
import cv2
```

```
from matplotlib import pyplot as plt
```

```
In [ ]: image_path = input("Enter image path: ")
```

```
In [ ]: img = cv2.imread(image_path)
        plt.imshow(img)
```

```
Out[ ]: <matplotlib.image.AxesImage at 0x1e2cad04610>
```


0
200
400
600
800
1000
1200
1400
0
200
400
600
800

Threads

There is an execution instance which has its own cpu time and memory.

In Java, thread is an inbuilt class which belongs Java. lang Package, which consists of 'n' number of methods (static) such as Sleep () & CurrentThread ().

What do you mean by multi-tasking?

Process of executing multiple tasks Simultaneously is called as multi-tasking.

- Multi-threading - Process of executing multiple threads simultaneously is called as multi-threading.

Daemon Thread

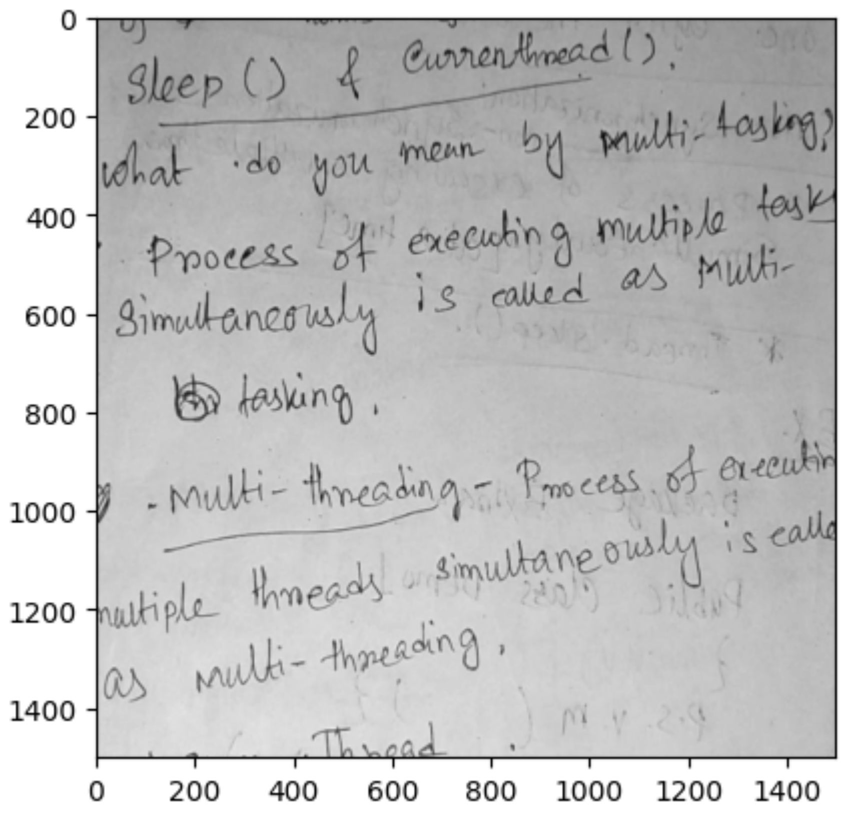
The threads which are running in the background Daemon Threads

Ex. Runnable Interface & Comparable interface.

Functional Interface - An Interface having only which is Abstract method is called as Functional Interface.

```
In [ ]: img = auto_crop_image(image_path)
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        gray = cv2.medianBlur(gray, 3)
        thresh = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY_INV, 2
        dilate = cv2.dilate(thresh, (5, 5), iterations=10)
        plt.imshow(gray, cmap="gray")
```

```
Out[ ]: <matplotlib.image.AxesImage at 0x1e2cae94550>
```


0
200
400
600
800
1000
1200
1400

Sleep () & Currentthread().

What do you mean by multi-tasking?

Process of executing multiple task simultaneously is called as Multi-tasking.

- Multi-threading - Process of executin
multiple threads simultaneously is calle
as multi-threading.

Thread

In [ ]:
```
mydataset = {
    'letter_slant': [get_letter_slant(image_path=image_path)[0]],
    'line_slant': [get_line_slant(image_path=image_path)[0]],
    'letter_size': [get_letter_size(image_path=image_path)[0]],
    'word_spacing': [gap_between_words(image_path=image_path)[0]],
}
my_df = pd.DataFrame(mydataset)
my_df["letter_slant"] = my_df["letter_slant"].map(letter_slant_mapping)
my_df["line_slant"] = my_df["line_slant"].map(line_slant_mapping)
my_df["word_spacing"] = my_df["word_spacing"].map(word_spacing_mapping)
my_df
```

Out[ ]:

	letter_slant	line_slant	letter_size	word_spacing
0	1	1	100.9	-1

In [ ]:
```
my_pred = rf.predict(my_df)
my_pred
```

Out[ ]:
```
array(['Openness'], dtype=object)
```