```
#import git
!wget --no-check-certificate \
  https://github.com/dicodingacademy/assets/raw/main/ml_pengembangan_academy/Chessman-image-dataset.zip \
  -O /tmp/Chessman-image-dataset.zip
```

```
--2024-01-03 09:06:15--  https://github.com/dicodingacademy/assets/raw/main/ml_pengembangan_academy/Chessman-image-dataset.zip
Resolving github.com (github.com)... 140.82.112.3
Connecting to github.com (github.com)|140.82.112.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/dicodingacademy/assets/main/ml_pengembangan_academy/Chessman-image-dataset.zip [followi
--2024-01-03 09:06:15--  https://raw.githubusercontent.com/dicodingacademy/assets/main/ml_pengembangan_academy/Chessman-image-datas
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.109.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 60684125 (58M) [application/zip]
Saving to: ‘/tmp/Chessman-image-dataset.zip’

/tmp/Chessman-image 100%[===================>]  57.87M   216MB/s    in 0.3s

2024-01-03 09:06:16 (216 MB/s) - ‘/tmp/Chessman-image-dataset.zip’ saved [60684125/60684125]
```

```
#ekstrak dataset git
import os
import zipfile
local_zip = '/tmp/Chessman-image-dataset.zip'
zip_ref = zipfile.ZipFile(local_zip, 'r')
zip_ref.extractall('/tmp')
zip_ref.close()
```

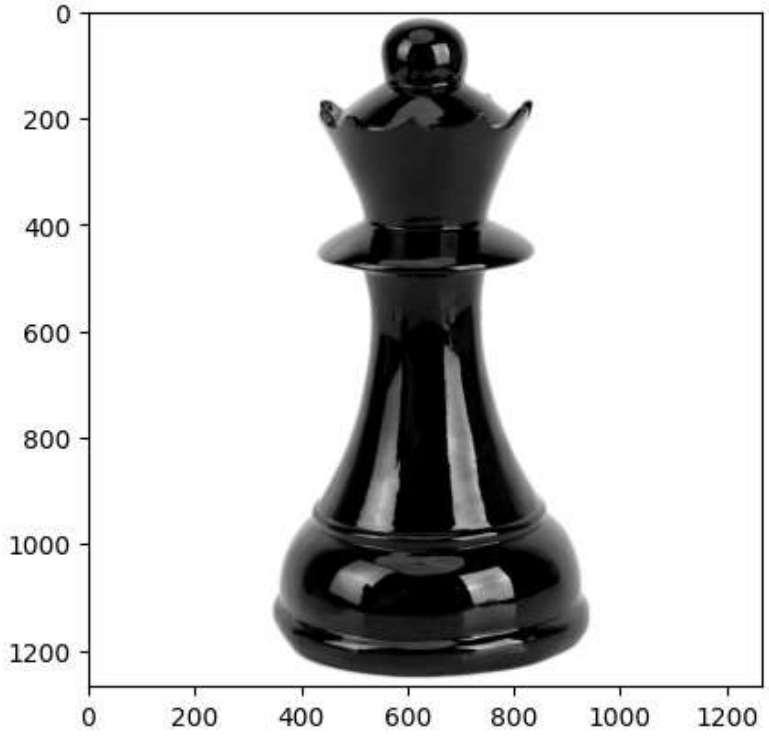
```
#read isi direktori dataset
os.listdir('/tmp/Chessman-image-dataset/Chess')
```

```
['Rook', 'Bishop', 'Queen', 'Pawn', 'Knight', 'King']
```

```
#menghitung jumlah data pada tiap direktori
print('total pawn images :', len(os.listdir('/tmp/Chessman-image-dataset/Chess/Pawn')))
print('total King images :', len(os.listdir('/tmp/Chessman-image-dataset/Chess/King')))
print('total Knight images :', len(os.listdir('/tmp/Chessman-image-dataset/Chess/Knight')))
print('total bishop images :', len(os.listdir('/tmp/Chessman-image-dataset/Chess/Bishop')))
print('total queen images :', len(os.listdir('/tmp/Chessman-image-dataset/Chess/Queen')))
print('total rook images :', len(os.listdir('/tmp/Chessman-image-dataset/Chess/Rook')))
```

```
total pawn images : 107
total King images : 76
total Knight images : 106
total bishop images : 87
total queen images : 78
total rook images : 102
```

```
#melihat salah satu sample dataset
from keras.preprocessing import image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
%matplotlib inline
img = image.load_img('/tmp/Chessman-image-dataset/Chess/Queen/00000001.jpg')
imgplot = plt.imshow(img)
```




```
#import image generator dan mengaplikasikan augmentasi gambar
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_dir = os.path.join('/tmp/Chessman-image-dataset/Chess')
train_datagen = ImageDataGenerator(rescale=1./255,
                                   rotation_range=20,
                                   zoom_range=0.2,
                                   shear_range=0.2,
                                   fill_mode='nearest',
                                   validation_split=0.1) #set validation split
```

```
#membagi data training dan data testing menggunakan parameter subset 'training'/'validation'
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(150, 150),
    batch_size=8,
    class_mode='categorical',
    subset='training'
) #set as training data
validation_generator = train_datagen.flow_from_directory(
    train_dir, #direktori yang sama dengan data train
    target_size=(150, 150),
    batch_size=16,
    class_mode='categorical',
    subset='validation'
)
```

```
 Found 499 images belonging to 6 classes.
 Found 52 images belonging to 6 classes.
```

```
#arsitektur model 3 layer convolution dan 2 hidden layer; 512 dan 256 unit perseptron
import tensorflow as tf
model = tf.keras.models.Sequential([
    #input shape 150x150 pixel dan 3 bytes warna
    tf.keras.layers.Conv2D(64, (3,3), activation='relu', input_shape=(150, 150, 3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Dropout(0.4),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Dropout(0.4),
    #flatten the result to feed into a DNN
    tf.keras.layers.Flatten(),
    #512 neuron hidden layer
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(256, activation='relu'),
    #output dengan 1 neuron
    tf.keras.layers.Dense(6, activation='softmax')
])
```

```
#optimizer dan loss
model.compile(optimizer=tf.optimizers.Adam(),
              loss='categorical_crossentropy',
              metrics=['accuracy'])


history = model.fit(train_generator,
                            validation_data=validation_generator,
                            epochs=50,
                            verbose=2)
```

```
63/63 - 6s - loss: 1.3003 - accuracy: 0.5130 - val_loss: 1.4584 - val_accuracy: 0.5054 - 6s/epoch - 102ms/step
Epoch 18/50
```

```
63/63 - 7s - loss: 0.6760 - accuracy: 0.7515 - val_loss: 1.5865 - val_accuracy: 0.5000 - 7s/epoch - 104ms/step
Epoch 36/50
63/63 - 9s - loss: 0.7047 - accuracy: 0.7515 - val_loss: 1.6281 - val_accuracy: 0.4808 - 9s/epoch - 140ms/step
Epoch 37/50
63/63 - 7s - loss: 0.6203 - accuracy: 0.7836 - val_loss: 1.5813 - val_accuracy: 0.4423 - 7s/epoch - 104ms/step
Epoch 38/50
63/63 - 8s - loss: 0.7450 - accuracy: 0.7214 - val_loss: 1.4331 - val_accuracy: 0.4231 - 8s/epoch - 120ms/step
Epoch 39/50
63/63 - 7s - loss: 0.7406 - accuracy: 0.7415 - val_loss: 1.6231 - val_accuracy: 0.4615 - 7s/epoch - 112ms/step
Epoch 40/50
63/63 - 7s - loss: 0.5770 - accuracy: 0.7856 - val_loss: 1.4467 - val_accuracy: 0.5385 - 7s/epoch - 105ms/step
Epoch 41/50
63/63 - 8s - loss: 0.5671 - accuracy: 0.7796 - val_loss: 1.5930 - val_accuracy: 0.4038 - 8s/epoch - 123ms/step
Epoch 42/50
63/63 - 7s - loss: 0.5466 - accuracy: 0.7916 - val_loss: 1.6129 - val_accuracy: 0.4423 - 7s/epoch - 106ms/step
Epoch 43/50
63/63 - 7s - loss: 0.5501 - accuracy: 0.7876 - val_loss: 1.2675 - val_accuracy: 0.4808 - 7s/epoch - 104ms/step
Epoch 44/50
63/63 - 6s - loss: 0.5687 - accuracy: 0.8116 - val_loss: 1.2717 - val_accuracy: 0.5962 - 6s/epoch - 102ms/step
Epoch 45/50
63/63 - 8s - loss: 0.5048 - accuracy: 0.8176 - val_loss: 1.3377 - val_accuracy: 0.4615 - 8s/epoch - 122ms/step
Epoch 46/50
63/63 - 7s - loss: 0.4994 - accuracy: 0.8317 - val_loss: 1.4568 - val_accuracy: 0.5962 - 7s/epoch - 117ms/step
```

```
#plot loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Loss Model')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```


```
#plot akurasi
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Akurasi Model')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```