```
!pip install yfinance --quiet
!pip install pmdarima --quiet


!pip install statsmodels==0.11.0rc1 --quiet
!pip install -Iv pulp==1.6.8 --quiet


import yfinance as yf

# getting data from Yahoo Finance
stock_name = 'AMD'  # here you can change the name of stock ticker, for ex
ample we will take AMD ticker
data = yf.download(stock_name, start="2020-03-26", end="2022-08-01")


# import plotly package for graphs
import plotly
import plotly.graph_objs as go
import plotly.express as px
from plotly.subplots import make_subplots


data_adf = data.drop(['Open', 'High', 'Low', 'Adj Close', 'Volume'], axis=
1)
data_adf = data_adf['Close']

from pmdarima.arima import ADFTest
adf_test = ADFTest(alpha = 0.05)
adf_test.should_diff(data_adf)

import os
import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import pmdarima as pm
plt.style.use('fivethirtyeight')
from pylab import rcParams
rcParams['figure.figsize'] = 10, 6
from statsmodels.tsa.arima_model import ARIMA
from pmdarima.arima import ADFTest
from pmdarima.datasets import load_wineind
import random
```

```
def arima(stock_name, data):
    df_close = data['Close']

    # Split data into train and test set (90% - train, 10% - test)
    df_log = df_close
    #train_data, test_data = df_log[3:int(len(df_log) * 0.9)], df_log[int(
len(df_log) * 0.9):]
    train_data, test_data = df_log[3:int(len(df_log) * 0.9)], df_log[int(l
en(df_log) * 0.9):]
    test_values = len(df_log) * 0.01 + 1.0
    x_train = list(range(0, 224))
    x_test = list(range(224, int(len(data))))

    fig = go.Figure()
    fig.add_trace(go.Scatter(x=x_train, y=train_data, mode='lines+markers'
, marker=dict(size=4),  name='train', marker_color='#39304A'))
    fig.add_trace(go.Scatter(x=x_test, y=test_data, mode='lines+markers',
marker=dict(size=4), name='test', marker_color='#A98D75'))
    fig.update_layout(legend_orientation="h",
                  legend=dict(x=.5, xanchor="center"),
                  plot_bgcolor='#FFFFFF',
                  xaxis=dict(gridcolor = 'lightgrey'),
                  yaxis=dict(gridcolor = 'lightgrey'),
                  title_text = f'{stock_name} ARIMA data', title_x = 0.5,
                  xaxis_title="Timestep",
                  yaxis_title="Stock price",
                  margin=dict(l=0, r=0, t=30, b=0))
    fig.show()

    model =  pm.auto_arima(df_log,start_p=0, d=None, start_q=0,
                           max_p=5, max_d=5, max_q=5, start_P=0,
                           D=1, start_Q=0, max_P=5, max_D=5,
                           max_Q=5, m=7, seasonal=True,
                           error_action='warn',trace = True,
                           supress_warnings=True,stepwise = True,
                           random_state=20,n_fits = 50 )

    model.summary()

    exo_data = data['Volume']
    exo_data = exo_data[int(len(exo_data) * 0.9):]

    preds = model.predict(n_periods = 22, X = exo_data)

    preds = np.vstack(preds)
```

```
    hist_data = yf.download(stock_name, start="2021-04-09", end="2021-07-
09")
    hist_data = hist_data.drop(['Open', 'High', 'Low', 'Adj Close', 'Volum
e'], axis=1)
    hist_data = hist_data['Close']
    hist_data = np.array(hist_data)

    rmse = np.sqrt(np.mean(((preds - hist_data) ** 2)))
    print(f'RMSE ARIMA: {rmse}')

    # build graphs
    preds_gr = np.reshape(preds, (22,))
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=list(range(0, 21)), y=hist_data, mode='line
s+markers',  name='historical', marker_color='#39304A'))
    fig.add_trace(go.Scatter(x=list(range(0, 21)), y=preds_gr, mode='lines
+markers', name='predictions', marker_color='#FFAA00'))
    fig.update_layout(legend_orientation="h",
                      legend=dict(x=.5, xanchor="center"),
                      plot_bgcolor='#FFFFFF',
                      xaxis=dict(gridcolor = 'lightgrey'),
                      yaxis=dict(gridcolor = 'lightgrey'),
                      title_text = f'{stock_name} ARIMA prediction', title_x =
 0.5,
                      xaxis_title="Timestep",
                      yaxis_title="Stock price",
                      margin=dict(l=0, r=0, t=30, b=0))
    fig.show()

    return preds, rmse

arima_pred, arima_rmse = arima(stock_name, data)
print(arima_pred.shape)
```