import datetime as dt  
import yfinance as yf  
import pandas as pd  
from typing import List, Tuple, Optional  
  
# 第一部分：导入模块和定义函数  
def get\_data(symbol: str) -> pd.DataFrame:  
 data = yf.download(symbol, start="2018-01-01", end="2022-01-01")  
 return data  
  
# 第二部分：下载股票数据  
def download\_stock\_data(symbols: List[str], start\_date: str, end\_date: str) -> dict:  
 *"""  
 Download stock data for given symbols and time period* ***:param*** *symbols: List of stock symbols to download data for* ***:param*** *start\_date: Start date for data (format: "YYYY-MM-DD")* ***:param*** *end\_date: End date for data (format: "YYYY-MM-DD")* ***:return****: Dictionary containing stock data for each symbol  
 """* stock\_data = {}  
 for symbol in symbols:  
 data = yf.download(symbol, start=start\_date, end=end\_date)  
 stock\_data[symbol] = data  
 return stock\_data  
  
# 第三部分: 计算金叉和死叉的日期以及相邻金叉之间的最低点  
def find\_crossing\_dates(ma\_list, symbol, data):  
 *"""  
 找出均线金叉和死叉的日期  
 """* # 创建一个字典来存储金叉和死叉的日期  
 crossings\_dict = {'symbol': symbol}  
  
 # 计算均线交叉点  
 for ma in ma\_list:  
 data['MA{}'.format(ma)] = data['Close'].rolling(ma).mean()  
 data.dropna(inplace=True)  
  
 # 找出金叉和死叉的日期  
 crossings = []  
 for i in range(1, len(data)):  
 if all(data['MA{}'.format(ma\_list[j])][i] > data['MA{}'.format(ma\_list[j+1])][i] for j in range(len(ma\_list)-1)):  
 crossings.append(('DEATH', data.index[i]))  
 elif all(data['MA{}'.format(ma\_list[j])][i] < data['MA{}'.format(ma\_list[j+1])][i] for j in range(len(ma\_list)-1)):  
 crossings.append(('GOLDEN', data.index[i]))  
  
 # 取最近的三个金叉和三个死叉  
 num\_crossings = min(6, len(crossings))  
 crossings\_dates = sorted([x[1] for x in crossings[-num\_crossings:]])  
  
 # 找出相邻两个金叉之间的最低点的日期和数值  
 lowest\_points = []  
 for i in range(len(crossings\_dates)-1):  
 start\_date = crossings\_dates[i]  
 end\_date = crossings\_dates[i+1]  
 temp\_data = data.loc[start\_date:end\_date]  
 low\_price = temp\_data['Low'].min()  
 low\_date = temp\_data[temp\_data['Low'] == low\_price].index[0]  
 lowest\_points.append((low\_date, low\_price))  
  
 crossings\_dict['Golden\_Crossing\_Date1'] = crossings\_dates[-2]  
 crossings\_dict['Golden\_Crossing\_Date2'] = crossings\_dates[-4]  
 crossings\_dict['Golden\_Crossing\_Date3'] = crossings\_dates[-6]  
 crossings\_dict['Death\_Crossing\_Date1'] = crossings\_dates[-1]  
 crossings\_dict['Death\_Crossing\_Date2'] = crossings\_dates[-3]  
 crossings\_dict['Death\_Crossing\_Date3'] = crossings\_dates[-5]  
 crossings\_dict['Lowest\_Point\_Date1'] = lowest\_points[-2][0]  
 crossings\_dict['Lowest\_Point\_Value1'] = lowest\_points[-2][1]  
 crossings\_dict['Lowest\_Point\_Date2'] = lowest\_points[-1][0]  
 crossings\_dict['Lowest\_Point\_Value2'] = lowest\_points[-1][1]  
  
 return crossings\_dict  
  
# 第四部分  
ma\_dict = {}  
  
for symbol in symbols:  
 ma\_dict[symbol] = {}  
 for ma in ma\_list:  
 ma\_dict[symbol][f"MA\_{ma}"] = add\_moving\_averages(data[symbol], ma)  
  
  
# 第五部分  
def get\_results(symbols, ma\_list, crossings\_dict, lowest\_points\_dict):  
 results\_dict = {'Symbol': [], 'Last Close Price': []}  
 for ma in ma\_list:  
 results\_dict[f'{ma}-day MA'] = []  
 for i in range(1, num\_crossings+1):  
 results\_dict[f'Golden Cross {i}'] = []  
 results\_dict[f'Death Cross {i}'] = []  
 results\_dict[f'Lowest Point {i}-1 Date'] = []  
 results\_dict[f'Lowest Point {i}-1 Value'] = []  
 results\_dict[f'Lowest Point {i}-2 Date'] = []  
 results\_dict[f'Lowest Point {i}-2 Value'] = []  
 results\_dict[f'Lowest Point Diff {i}'] = []  
 for symbol in symbols:  
 results\_dict['Symbol'].append(symbol)  
 last\_close = data[symbol]['Close'][-1]  
 results\_dict['Last Close Price'].append(last\_close)  
 for ma in ma\_list:  
 results\_dict[f'{ma}-day MA'].append(ma\_dict[symbol][ma][-1])  
 if symbol not in crossings\_dict:  
 for i in range(1, num\_crossings+1):  
 results\_dict[f'Golden Cross {i}'].append('')  
 results\_dict[f'Death Cross {i}'].append('')  
 results\_dict[f'Lowest Point {i}-1 Date'].append('')  
 results\_dict[f'Lowest Point {i}-1 Value'].append('')  
 results\_dict[f'Lowest Point {i}-2 Date'].append('')  
 results\_dict[f'Lowest Point {i}-2 Value'].append('')  
 results\_dict[f'Lowest Point Diff {i}'].append('')  
 else:  
 for i, crossing in enumerate(crossings\_dict[symbol][:num\_crossings]):  
 if i == 0:  
 if crossing[1] == 'Golden':  
 results\_dict[f'Golden Cross {i+1}'].append(crossing[0])  
 results\_dict[f'Death Cross {i+1}'].append('')  
 else:  
 results\_dict[f'Golden Cross {i+1}'].append('')  
 results\_dict[f'Death Cross {i+1}'].append(crossing[0])  
 else:  
 if crossing[1] == 'Golden':  
 results\_dict[f'Golden Cross {i+1}'].append(crossing[0])  
 results\_dict[f'Death Cross {i+1}'].append(crossings\_dict[symbol][i-1][0])  
 lp1 = lowest\_points\_dict[symbol][(crossings\_dict[symbol][i-1][0], crossing[0])]  
 lp2 = lowest\_points\_dict[symbol][(crossing[0], crossings\_dict[symbol][i+1][0])]  
  
 results\_dict[f'Lowest Point {i+1}-1 Date'].append(lp1[0])  
 results\_dict[f'Lowest Point {i+1}-1 Value'].append(lp1[1])  
 results\_dict[f'Lowest Point {i+1}-2 Date'].append(lp2[0])  
 results\_dict[f'Lowest Point {i+1}-2 Value'].append(lp2[1])  
 results\_dict[f'Lowest Point Diff {i+1}'].append(lp2[1]-lp1[1])  
 else:  
 results\_dict[f'Golden Cross {i+1}'].append(crossings\_dict[symbol][i-1][0])  
 results  
  
# 第六部分  
def save\_results(results: pd.DataFrame, filename: str):  
 results.to\_excel(filename, index=False)  
  
save\_results(get\_results(['AAPL', 'GOOG'], [10, 20]), 'results.xlsx')