Overview:
In our project proposal, we designed an application to detect trading signals real time. However, we cannot find a free real time stock price API, so instead we use an API that gives
us the daily closing price of stock(its database is updated after market closes), and our program will check any trading signals from 5 different indicators calculated from the closing prices. Other indicators are not available because either they require data other than closing price(such as volume, which is not given by the API we use), or they require all the historical data(data from the day they were publicly listed), such as KDJ. The five indicators we use are: Fibonacci ma, ma5 and ma10, MTM, ROC, W%R.

IDE
Pycharm
Replit
IDLE

Libraries

```
import tushare as ts
import datetime
```

```
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
```

Source code
Main module:

```
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;

class Main extends Thread {
  public static double max(int day, double[]prices){  //function to
calculate maximum price of certain days
    double max=0;
    for(int g=0;g<day;g++){
      if(prices[g]>max){
        max=prices[g];
```

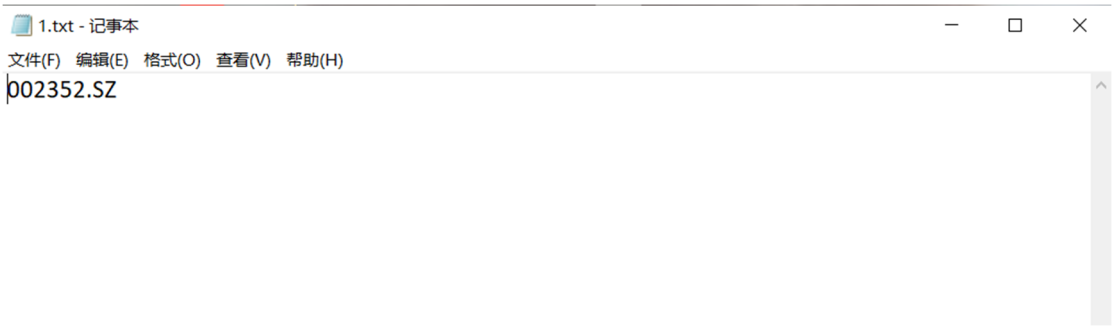
```
    }
   }return max;
  }

  public static double ma(int startma,int numberofdays,
double[]prices){
   double ma=0;
   for(int o=startma;o<numberofdays+startma;o++){ //function to
calculate moving average
    ma=ma+prices[o];
    }
   ma=ma/numberofdays;
   return ma;
  }

  public static double min(int day, double[]prices){ //function to
calculate minimum price of certain days
   double min=1000000;
   for(int h=0;h<day;h++){
    if(prices[h]<min){
     min=prices[h];
    }
   }return min;
  }

  public static void main(String[] args) {
   int repeat=1;
   do{
   Scanner myObj = new Scanner(System.in);
   System.out.println("Enter stock code(such as 600519.SH):");
//take input of stock code from user

   String code = myObj.nextLine();
   try {
    FileWriter myWriter = new FileWriter("1.txt"); //pass stock
code to txt for fetch program to read

    myWriter.write(code);
    myWriter.close();
    System.out.println("analysis starts");
   } catch (IOException e) {
    System.out.println("An error occurred.");
    e.printStackTrace();
   }
```

```
final Runtime runtime = Runtime.getRuntime(); //execute program to
fetch data of stock prices
Process process = null;
try {
process = runtime.exec("fetch.exe");
 } catch (final Exception e) {
System.out.println("Error exec!");
  }
      double[] prices = new double[24];//create an array to store
the prices
     try{
       Thread.sleep(6000);  //make  the  program  wait  for  stock
prices to be fetched
        }
       catch (InterruptedException e){
         System.out.println("An error occurred.");
     e.printStackTrace();
       }
   try {
     File myObje = new File("ass.txt");
     Scanner myReader = new Scanner(myObje);
     for(int i=0;i<24;i++) {
       String data = myReader.nextLine();
       double  price  =   Double.parseDouble(data);  //read  stock
prices fetched by python, store them in an array
       prices[i]=price;
     }
     myReader.close();
   } catch (FileNotFoundException e) {
     System.out.println("An error occurred.");
     e.printStackTrace();
   }

   double  ma3now=ma(0,3,prices);  //calculate  the  value  for
indicators
   double ma3prev=ma(1,3,prices);
   double ma13now=ma(0,13,prices);
   double ma13prev=ma(1,13,prices);
   double ma5now=ma(0,5,prices);
   double ma5prev=ma(1,5,prices);
   double ma10now=ma(0,10,prices);
   double ma10prev=ma(1,10,prices);

   double mtmnow=0;
```

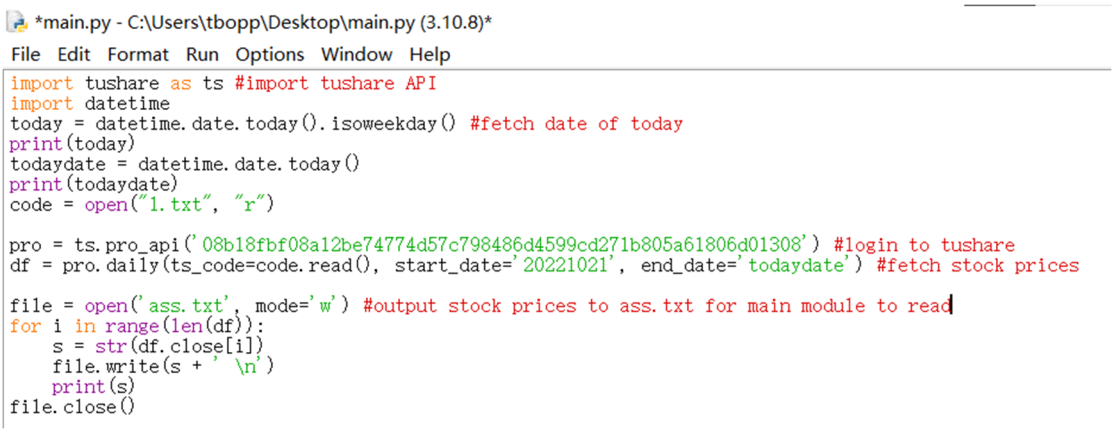
```
mtmnow=prices[0]-prices[10];
double mtmprev=0;
mtmprev=prices[1]-prices[11];

double rocnow=0;
rocnow=(prices[0]-prices[10])/prices[10];
double rocprev=0;
rocprev=(prices[1]-prices[11])/prices[11];

double wr=0;
double max11=max(11,prices);
double min11=min(11,prices);
wr=(max11-prices[0])/(max11-min11)*100;

String wrsignal=""; //identify the signal of indicators
if(wr<=10){
  wrsignal="sell";
}else if(wr>=90){
  wrsignal="purchase";
}else{
  wrsignal="neutral";
}

String mtmsignal="";
if(mtmnow<=0&&mtmprev>0){
  mtmsignal="sell";
}else if(mtmnow>=0&&mtmprev<0){
  mtmsignal="purchase";
}else if(mtmnow<=0){
  mtmsignal="bear market";
}else if(mtmnow>0){
  mtmsignal="bull market";
}

String rocsignal="";
if(rocnow<=0&&rocprev>0){
  rocsignal="sell";
}else if(rocnow>=0&&rocprev<0){
  rocsignal="purchase";
}else if(rocnow<=0){
  rocsignal="bear market";
}else if(rocnow>0){
  rocsignal="bull market";
}
```

```
    String masignal="";
    if(ma5prev<ma10prev&&ma5now>ma10now){
      masignal="purchase";
    }else if(ma5prev>ma10prev&&ma5now<ma10now){
      masignal="sell";
    }else if(ma5prev>ma10prev&&ma5now>ma10now){
      masignal="bull market";
    }else if(ma5prev<ma10prev&&ma5now<ma10now){
      masignal="bear market";
    }

    String fibsignal="";
    if(ma3prev<ma13prev&&ma3now>ma13now){
      fibsignal="purchase";
    }else if(ma3prev>ma13prev&&ma3now<ma13now){
      fibsignal="sell";
    }else if(ma3prev>ma13prev&&ma3now>ma13now){
      fibsignal="bull market";
    }else if(ma3prev<ma13prev&&ma3now<ma13now){
      fibsignal="bear market";
    }

    System.out.println("Signal of Willian indicator(W%R) is:
"+wrsignal); //output the signals
    System.out.println("Signal of moving average is: "+masignal);
    System.out.println("Signal of Fibonacci moving average is:
"+fibsignal);
    System.out.println("Signal of price rate of change
indicator(ROC) is: "+rocsignal);
    System.out.println("Signal of momentum indicator(MTM) is:
"+mtmsignal);

    Scanner myObject = new Scanner(System.in);
    System.out.println("press 1 to continue");

    int input = myObject.nextInt();
    repeat=input;
    }while(repeat==1);
  }
}
```

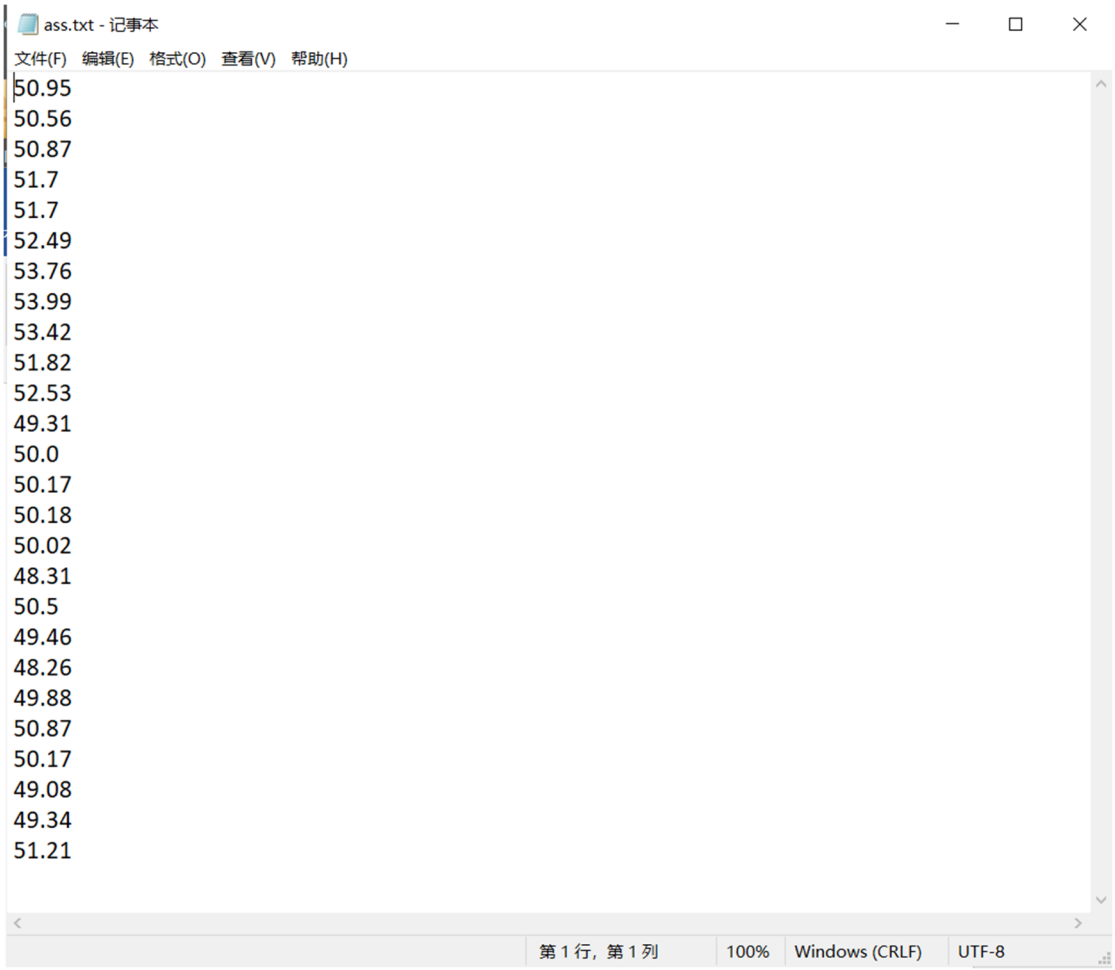
The main module will read the stock code input by user and store it in 1.txt for python module to read



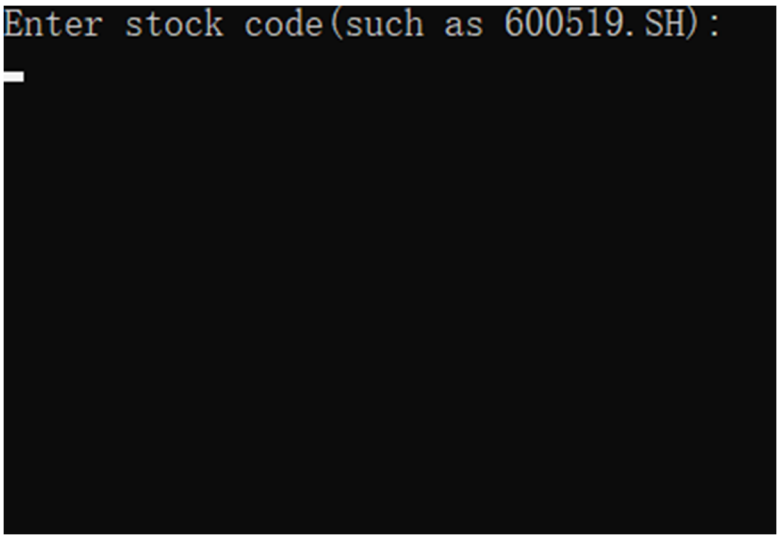
python module to fetch stock prices of the stock indicated by the user



The stock prices fetched by python module will be stored in ass.txt for main module to read



User interface prompting user to input stock code



User interface after stock code input, the signals are output

```
Enter stock code(such as 600519.SH):
600519.SH
analysis starts
Signal of Willian indicator(W%R) is: purchase
Signal of moving average is: bear market
Signal of Fibonacci moving average is: sell
Signal of price rate of change indicator(ROC) is: sell
Signal of momentum indicator(MTM) is: sell
press 1 to continue
```

Recent stock prices of 600519.SH.

Our system was run on this day and above are the signals given that day. Most indicators gave a "sell" signal.

User interface when the user repeats the program and input another stock code. Signals for the new stock is output

```
Enter stock code(such as 600519.SH):
600519.SH
analysis starts
Signal of Willian indicator(W%R) is: purchase
Signal of moving average is: bear market
Signal of Fibonacci moving average is: sell
Signal of price rate of change indicator(ROC) is: sell
Signal of momentum indicator(MTM) is: sell
press 1 to continue
1
Enter stock code(such as 600519.SH):
002460.SZ
analysis starts
Signal of Willian indicator(W%R) is: purchase
Signal of moving average is: bear market
Signal of Fibonacci moving average is: bear market
Signal of price rate of change indicator(ROC) is: bear market
Signal of momentum indicator(MTM) is: bear market
press 1 to continue
```

Recent stock trend of 002460,SZ

Our system was run on this day and above are the signals given that day. Most indicators gave a “bear market” signal.

Division of labor

Frank: Programmed the python module for fetching stock prices.

Daniel: Programmed the part that takes input from users. Did the testing of the program.

Tonny: Programmed the part that reading of stock prices, calculation and output of signals. Created this document

References

https://www.tushare.pro/ The website that provides the stock price API we use