Exercise 04

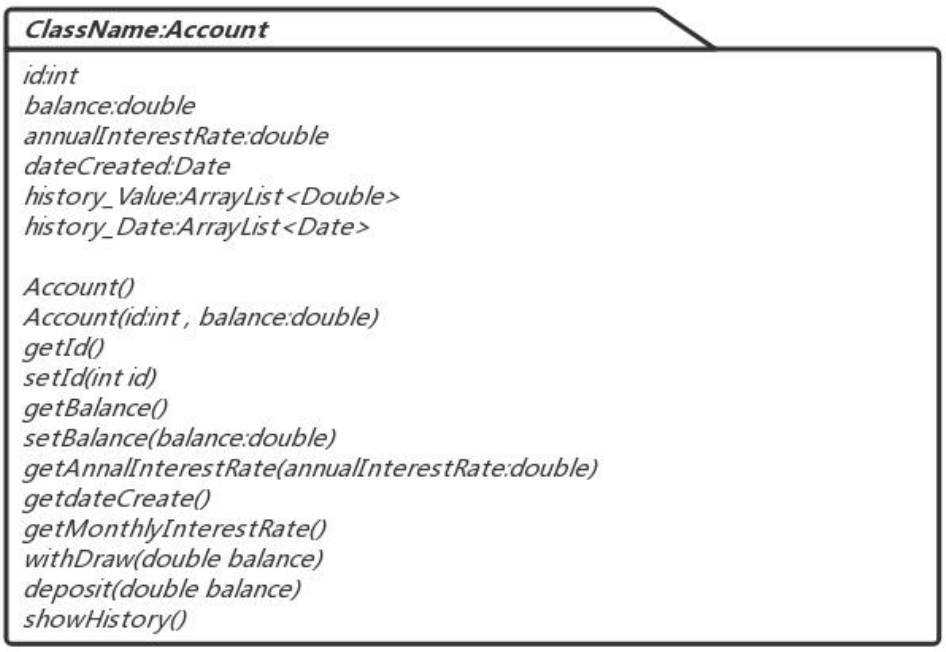
班级：2017211106 学号：2017212116 姓名：杨诺诚

问题一：创建 Account 类

解决思路：

需要新建一个 **Account** 类，在类中包含用户 **id**、存款、开户日期、利润率的信息。我还加入了一个模拟取款存款存折中的历史记录，可以记录每一次当前用户存取行为，这部分使用 **ArrayList** 类型。类中还要有一些方法，它们包括开户的函数（无参构造函数），指定 **id** 和存款的构造函数（有参构造函数）、获取或修改 **ID** 的函数，获取或修改存款金额的函数，获取或修改年利率的函数，获取开户时间的函数，获取每个月的利息的函数，存入或转出金额的函数，显示过往行为历史的函数。

IO 依据题设设计。**UML**图如下所示。



代码实现：

```
package com.company;

import java.util.ArrayList;
import java.util.Date;

public class Main {
   static class Account{
```

```
        private int id = 0;
        private double balance = 0.0;//本金
        private double annualInterestRate = 0.0;//年利率
        private Date dateCreated;
        private ArrayList<Double> history_Value = new ArrayList<Double>();
        private ArrayList<Date> history_Date = new ArrayList<Date>();
        Account(){
            dateCreated = new Date();
//            无参构造方法
        }
        Account(int id ,double balance){
            System.out.println("this.id= "+ this.id+'\n'+"id= "+id);
            dateCreated = new Date();
            this.id = id;
            this.balance = balance;
        }
        int getId(){
            return this.id;
        }
        void setId(int id){
            this.id = id;
        }
        String getBalance(){
            return String.valueOf(this.balance*100)+"%";
        }
        void setBalance(double balance){
            this.balance = balance;
        }
        double getAnnualInterestRate(){
            return this.annualInterestRate;
        }
        void setannualInterestRate(double annualInterestRate){
            this.annualInterestRate = annualInterestRate/100;
//            由小数转化为百分数（ 75 -> 75% ）
        }
        Date getdateCreate(){
            return this.dateCreated;
        }
        double getMonthlyInterestRate(){
            return balance*(annualInterestRate/12.0);
        }
        void withDraw(double balance){
            if(balance<this.balance){
               this.balance -= balance;
```

```
            System.out.println("用户取出"+balance+"元,"+"余额为"+this.balance);
            history_Value.add(-1*balance);
            history_Date.add(new Date());
        }
        else System.out.println("余额不足，请重新输入！");
    }
    void deposit(double balance){
        this.balance += balance;
        System.out.println("用户存入"+balance+"元,"+"余额为"+this.balance);
        history_Value.add(balance);
        history_Date.add(new Date());
    }
    void showHistory(){
        for(int i=0;i<history_Date.size();i++){
            if(i!=0) System.out.println("");
            System.out.print("用户在"+history_Date.get(i).toString());
            System.out.print(history_Value.get(i)<0 ? " 取出 ":" 存入 " +
Math.abs(history_Value.get(i)));
        }
    }
}
public static void main(String[] args) throws InterruptedException {
    Account MirSmith = new Account(1122,20000);
    MirSmith.setannualInterestRate(4.5);
    System.out.println("Smith 先生下个月将获得的利息为:
"+MirSmith.getMonthlyInterestRate());
    System.out.println("Smith 先生开户的时间为
"+MirSmith.getdateCreate().toString());
    Thread.sleep(5000);
    MirSmith.deposit(100000);
    Thread.sleep(5000);
    MirSmith.withDraw(9000);
    Thread.sleep(5000);
    MirSmith.withDraw(999999999);
    MirSmith.showHistory();
}
}
```

效果图展示:

```
this.id= 0
id= 1122
Smith先生下个月将获得的利息为: 75.0
Smith先生开户的时间为 Thu Oct 17 14:24:36 CST 2019
用户存入100000.0元,余额为120000.0
用户取出9000.0元,余额为111000.0
余额不足，请重新输入!
用户在Thu Oct 17 14:24:41 CST 2019 存入 100000.0
用户在Thu Oct 17 14:24:46 CST 2019 取出
Process finished with exit code 0
```

函数中所有的变量都设置为了 **private** 属性，可以在本类 **class** 中的函数中使用。

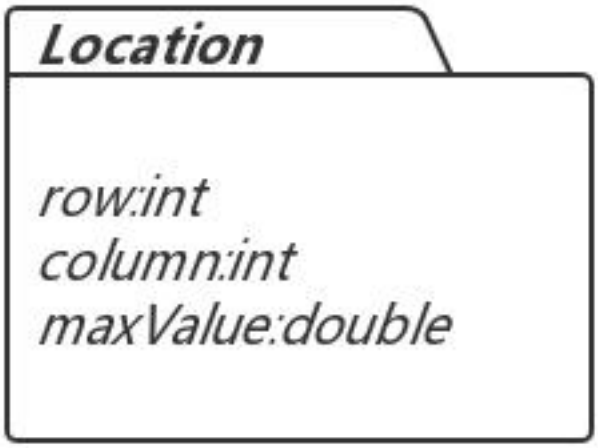
问题二：位置类（Location）

解决思路：

Location 中包含三个属性分别是行数，列数以及对应值，还需要建立一个方法，寻找二维矩阵中的最大值，并将位置和值保存在 **Location** 类中，在主函数中直接调用。

对于一些大型矩阵还可以直接通过随机数的方法生成，用户输入行数和列数后，可以选择是否自动生成矩阵中的数字，若选择否则继续手动输入全部数字。

UML 图如下所示。



代码实现：

```
package com.company;

import java.text.DecimalFormat;
import java.util.Random;
import java.util.Scanner;

public class q_2 {
    static class Location{
        int row = 0;
        int column = 0;
        double maxValue = 0.0;
    }
    public static Location locationLargest(double [][]a){
        Location myTable = new Location();
        myTable.maxValue = a[0][0];
```

```
        myTable.column = 0;
        myTable.row = 0;
        for(int i=0;i<a.length;i++){
            for(int j=0;j<a[0].length;j++){
                if(myTable.maxValue != Math.max(a[i][j],myTable.maxValue)){
                    myTable.column = i;
                    myTable.row = j;
                    myTable.maxValue = Math.max(a[i][j],myTable.maxValue);
                }
            }
        }
        return myTable;
    }
    public static void main(String[] args){
        Scanner input = new Scanner(System.in);
        System.out.println("Enter the number of rows and columns in the array:");
        int rows = input.nextInt();//行
        int colums = input.nextInt();//列
        double [][] table = new double [rows][colums];
        System.out.println("Do you want to input Automatic?(input y/n)");
        if(input.next().equals("n")){
            for (int i=0;i<rows;i++)
                for (int j=0;j<colums;j++)
                    table[i][j] = input.nextDouble();
        }
        else{
            Random rand =new Random((long) Math.random());
            DecimalFormat df = new DecimalFormat( "00");
            for (int i=0;i<rows;i++) {
                for (int j = 0; j < colums; j++)
                    table[i][j] =
Double.parseDouble(df.format(rand.nextInt(100)));
            }
            System.out.println("The table we created is :");
            for (int i=0;i<rows;i++){
                for(int j=0;j<colums;j++)
                    System.out.print(table[i][j]+" ");
                System.out.println("");
            }
        }
        Location ans = locationLargest(table);
        System.out.println("The location of the largest element is
"+ans.maxValue+" at ("+(ans.column+1)+","+(ans.row+1)+")");
```

```
    }
}
```

效果图展示：

```
Enter the number of rows and columns in the array:
5 5
Do you want to input Automatic?(input y/n)
y
The table we created is :
60.0 48.0 29.0 47.0 15.0
53.0 91.0 61.0 19.0 54.0
77.0 77.0 73.0 62.0 95.0
44.0 84.0 75.0 41.0 20.0
43.0 88.0 24.0 47.0 52.0
The location of the largest element is 95.0 at (3,5)
```

自动生成矩阵内容

```
Enter the number of rows and columns in the array:
5 5
Do you want to input Automatic?(input y/n)
n
1 2 3 4 5
6 7 8 9 8
7 6 4 2 1
1 3 6 3 2
5 2 8 9 1
The location of the largest element is 9.0 at (2,4)
```

手动输入矩阵内容