**Assignment 3**

Problem Description:

(*The Account class*) Design a class named Account that contains:

* A private int data field named id for the account (default 0).
* A private double data field named balance for the account (default 0).
* A private double data field named annualInterestRate that stores the current interest rate (default 0). Assume all accounts have the same interest rate.
* A private Date data field named dateCreated that stores the date when the account was created.
* A no-arg constructor that creates a default account.
* A constructor that creates an account with the specified id and initial balance.
* The accessor and mutator methods for id, balance, and annualInterestRate.
* The accessor method for dateCreated.
* A method named getMonthlyInterestRate() that returns the monthly interest rate.
* A method named withdraw that withdraws a specified amount from the account.
* A method named deposit that deposits a specified amount to the account.

Draw the UML diagram for the class. Implement the class. Write a test program that creates an Account object with an account ID of 1122, a balance of $20,000, and an annual interest rate of 4.5%. Use the withdraw method to withdraw $2,500, use the deposit method to deposit $3,000, and print the balance, the monthly interest, and the date when this account was created.

**Analysis:**

1. In this programme we have to calculate the balance and the monthly interest and date at which this account is created.
2. For this we have to design the class named account which contains the

* A private int data field named id.
* A private double data field named balance for the account.
* A private double data field named annualInterestRate that stores the current interest rate (default 0). Assume all accounts have the same interest rate.
* A private Date data field named dateCreated that stores the date when the account was created.
* A no-arg constructor that creates a default account.
* A constructor that creates an account with the specified id and initial balance.
* The accessor and mutator methods for id, balance, and annualInterestRate.
* The accessor method for dateCreated.
* A method named getMonthlyInterestRate() that returns the monthly interest rate.
* A method named withdraw that withdraws a specified amount from the account.
* A method named deposit that deposits a specified amount to the account.

1. For this we have to pass value account id as 1122 and we have to give balance as 20000$ and an annual interest rate is 4.5%.
2. After that we have to windrow the amount of 2500$ from the total balance.
3. After that we have to again make a deposit of 3000$ to the remaining amount.
4. For this above operation we have to calculate this by using the functions that calculates the remaining amount.
5. We have to print results of balance, monthly interest rate and account date.

**Design:**

**UML Diagram**

|  |
| --- |
| **Account** |
| -id:int  -balance:double  -annualIntrestRate:double  -dateCreated:date |
| +Account(id : int , balance :double)  +getid():int  +getBalance():double  +getAnnualIntrestRate():double  +getdateCreated()  +setid(id:int):void  +setBalance(Balance:double):void  +setAnnualIntrestRate(AnnualIntrestrate(double):void  +getMonthlyIntrestRate():double  +withdraw(amount:double)  +deposit(amount:double):void |

**Coding:**

/\*

\* @author Basavraj Jaliminche (8800149)

\* PROG8580 - Computer Programming - Section 1

\* Assignment 3

\*/

**package** Assignments;

**public** **class** Account {

**private** **int** id = 0;

**private** **double** balance = 0.0;

**private** **static** **double** *annualInterestRate* = 0.0;

**private** java.util.Date dateCreated;

**public** Account() {

dateCreated = **new** java.util.Date();

}

**public** Account(**int** id, **double** balance) {

**this**();

**this**.id = id;

**this**.balance = balance;

}

**public** **int** getId() {

**return** **this**.id;

}

**public** **double** getBalance() {

**return** **this**.balance;

}

**public** **double** getAnnualInterestRate() {

**return** *annualInterestRate*;

}

**public** String getDateCreated() {

**return** **this**.dateCreated.toString();

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** **void** setBalance(**double** balance) {

**this**.balance = balance;

}

**public** **void** setAnnualInterestRate(**double** annualInterestRate) {

**this**.*annualInterestRate* = annualInterestRate;

}

**public** **double** getMonthlyInterestRate() {

**return** (*annualInterestRate* / 100) / 12;

}

**public** **double** getMonthlyInterest() {

**return** balance \* getMonthlyInterestRate();

}

**public** **void** withdraw(**double** amount) {

**this**.balance -= amount;

}

**public** **void** deposit(**double** amount) {

**this**.balance += amount;

}

}

/\*

\* @author Basavraj Jaliminche (8800149)

\* PROG8580 - Computer Programming - Section 1

\* Assignment 3

\*/

**package** Assignments;

**public** **class** Assignment03 {

**public** **static** **void** main(String[] args) {

Account account = **new** Account(1122, 20000);

account.setAnnualInterestRate(4.5);

account.withdraw(2500.0);

account.deposit(3000.0);

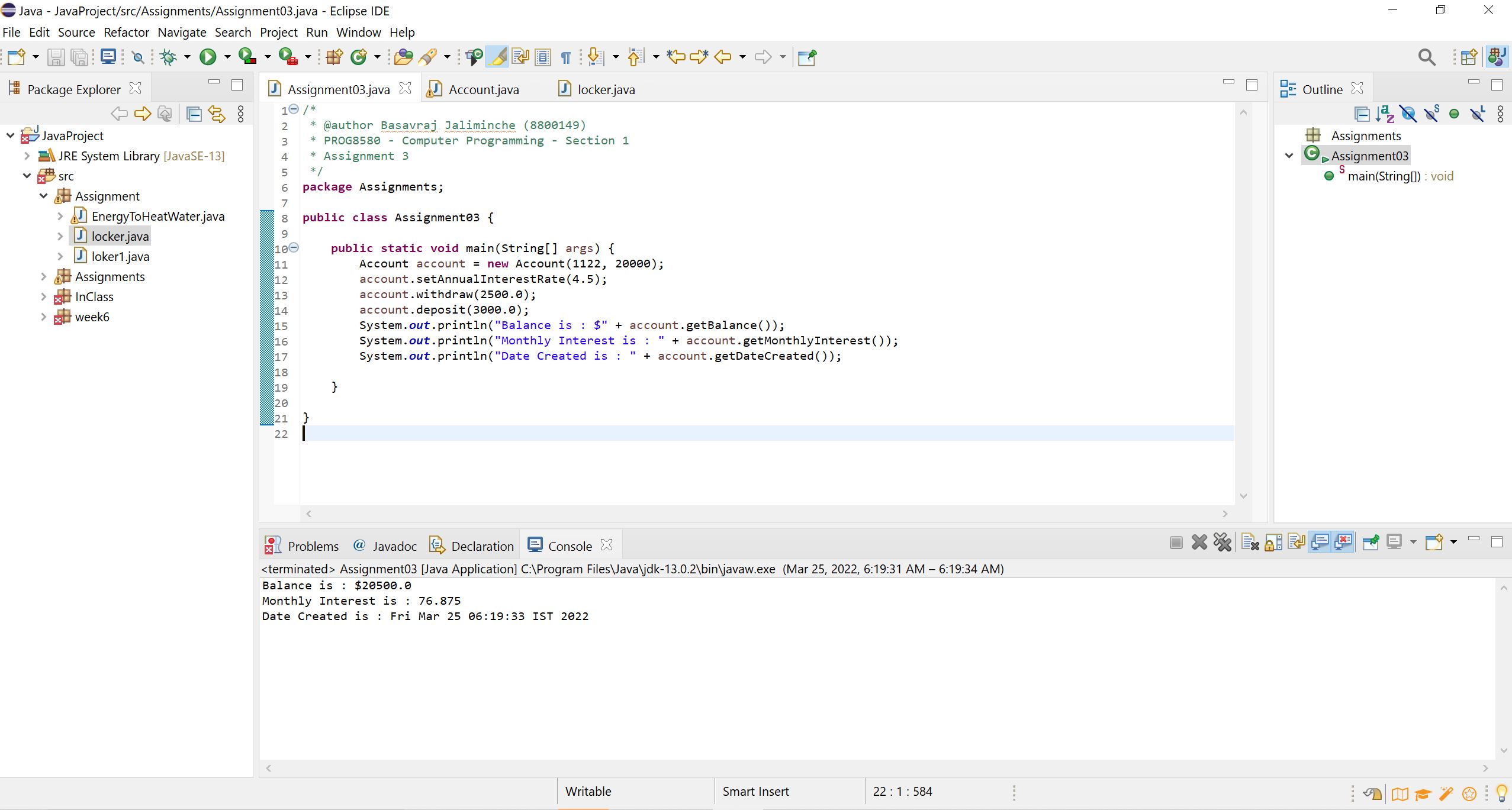
System.***out***.println("Balance is : $" + account.getBalance());

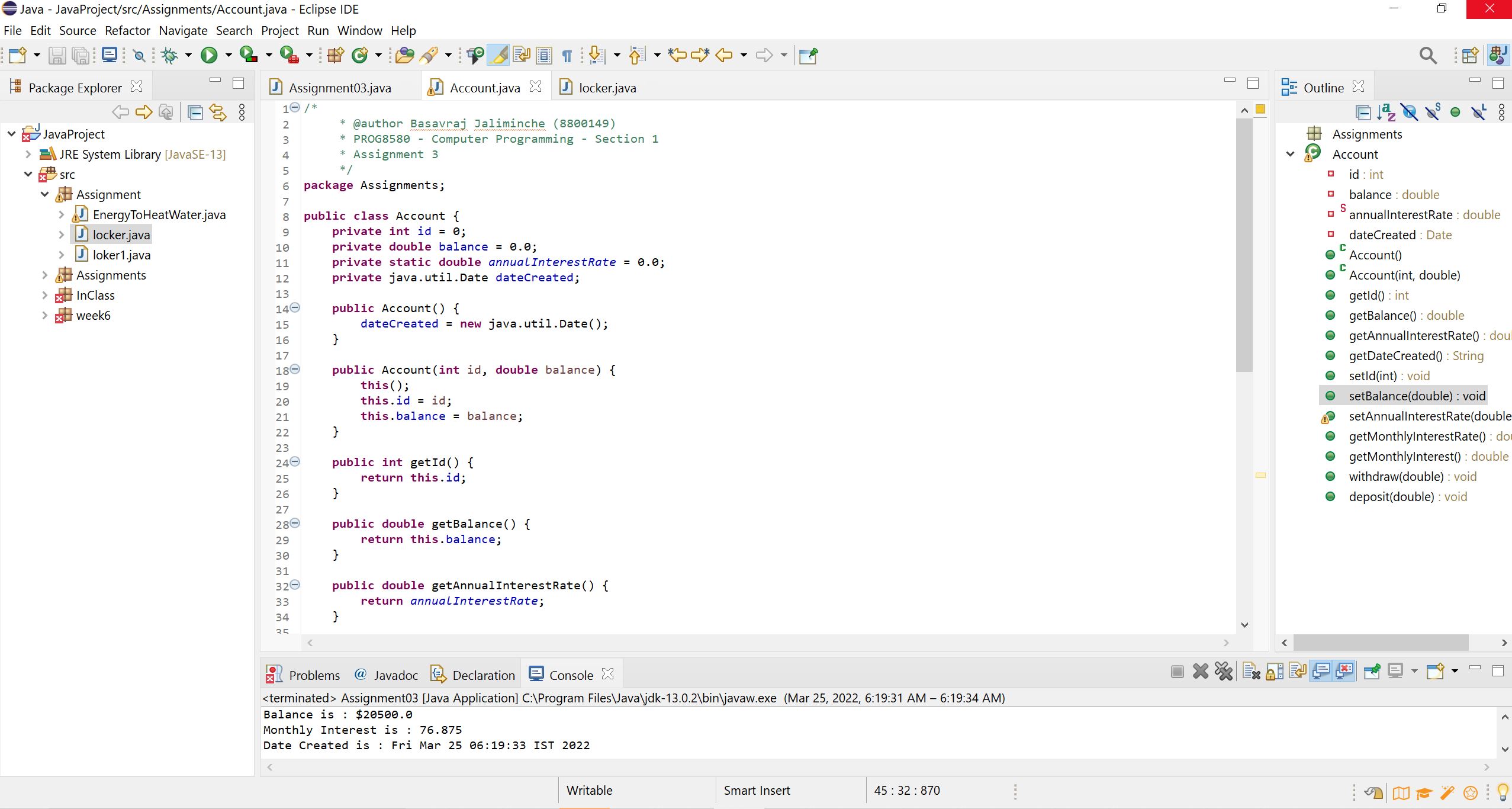
System.***out***.println("Monthly Interest is : " + account.getMonthlyInterest());

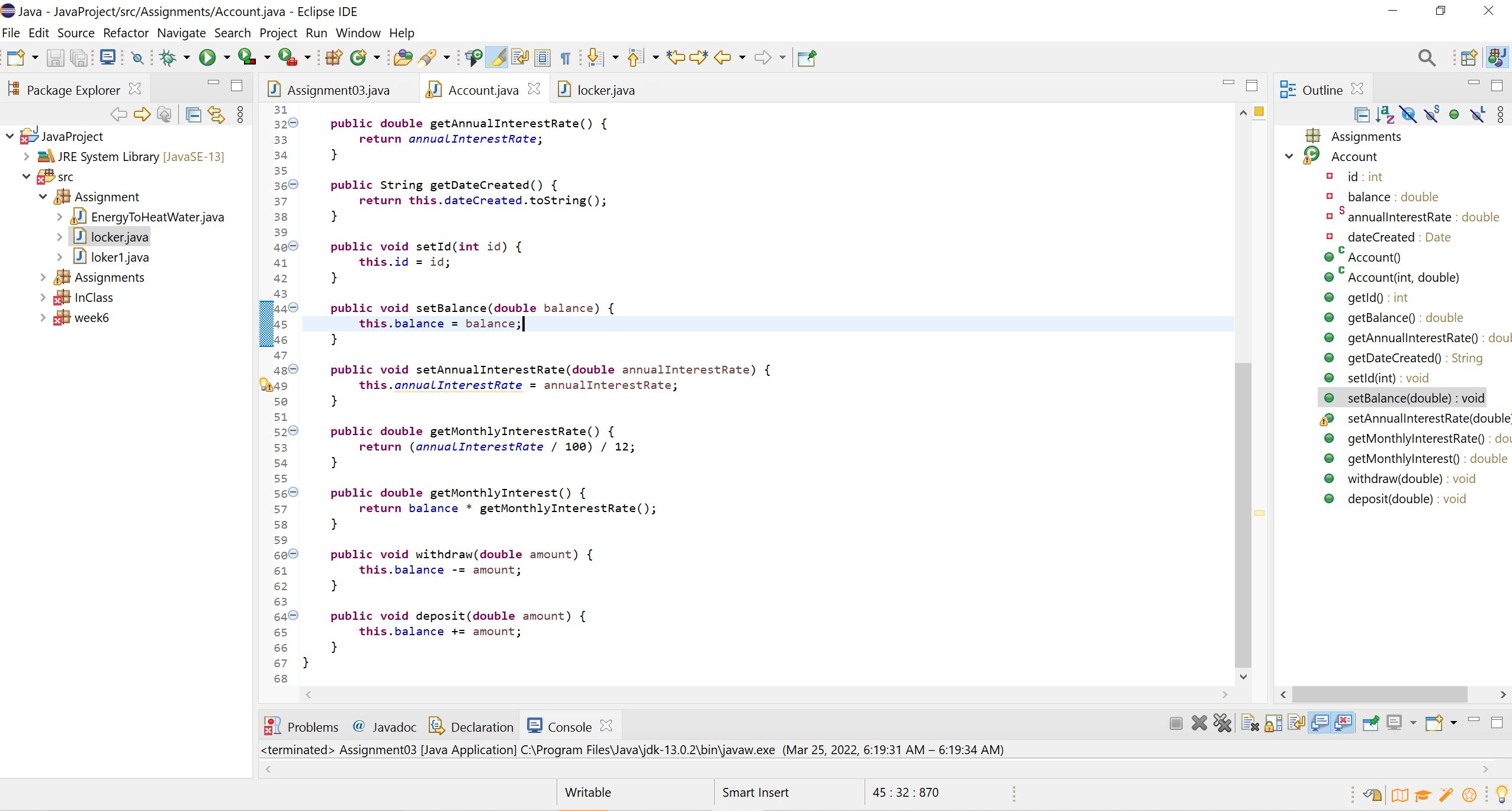
System.***out***.println("Date Created is : " + account.getDateCreated());

}

}







**Testing:**

For testing the code, we have to calculate the balance, monthly interest rate manually with the mathematical formula and match the results with the actual code.

