AASIGNMENT-1-OOAD-Guruteja_Kanderi-A20526883

1.Code

2.Exception handling

3.Output

4.Explanation

/* ACCOUNTHOLDERTEST.JAVA*/

```
package com.bankproject.Guru;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Scanner;

public class AccountHolderTest {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

//Set interest rate to 4%

AccountHolder.annualInterestRate = 0.04;

//declaringb the variable

AccountHolder accountHolder;

double balance = 0.0, deposit = 0.0, withdrawal = 0.0;

// Prompt user to enter initial balance, repeat until a positive value is
entered

while (true) {

System.out.println("Please enter the initial balance:");

try {

balance = Double.parseDouble(sc.nextLine());

if (balance < 0) {

System.out.println("Error: Balance cannot be negative. Please enter a
positive beginning balance.");

} else {
```

```
break;

}

//Executes this catch function if user gives negative values in Initial
Balance

} catch (NumberFormatException e) {

System.out.println("Invalid input. Please enter a valid number for
balance.");

}

}

accountHolder = new AccountHolder(balance);

// Prompt user to enter deposit amount, repeat until a positive value is
entered

while (true) {

System.out.println("Please enter the amount to be deposited:");

try {

deposit = Double.parseDouble(sc.nextLine());

if (deposit <= 0) {

System.out.println("Deposit amount should not be negative or zero. Please
enter a positive non-zero value for deposit.");

} else {

break;

}

//Executes this catch function if user gives negative values in deposit

} catch (NumberFormatException e) {

System.out.println("Invalid input. Please enter a valid number for
deposit.");

}

}
```

```java
accountHolder.deposit(deposit);

// Prompt user to enter withdrawal amount, repeat until a positive value is
entered

while (true) {

System.out.println("Please enter the amount to be withdrawn:");

try {

withdrawal = Double.parseDouble(sc.nextLine());

if (withdrawal <= 0) {

System.out.println("Withdrawal amount cannot be negative or zero. Enter a
positive non-zero value for withdrawal.");

} else if (withdrawal > accountHolder.getBalance()) {

System.out.format("Withdrawal amount should not be higher than account
balance. Enter the withdrawal amount within account balance. Your balance:
$%.2f%n", accountHolder.getBalance());

} else {

break;

}

//Executes this catch function if user gives negative values in withdrawal

} catch (NumberFormatException e) {

System.out.println("Invalid input. Please enter a valid number for
withdrawal.");

}

}

accountHolder.withdraw(withdrawal);

accountHolder.monthlyInterest();

//Prints New balance after adding interest for 12 months to the original
balance

System.out.println("New balance after one year with interest: $" +
accountHolder.getBalance());
```

AASIGNMENT-1-OOAD-Guruteja_Kanderi-A20526883

```
//Prints date and programmer's name at the end

String timeStamp = new SimpleDateFormat("yyyy/MM/dd
HH:mm:ss").format(Calendar.getInstance().getTime());

System.out.print("Current date & time =" + timeStamp + "\nProgrammed by
Guruteja\tkanderi\n");

sc.close(); // close the scanner object

}

}
```

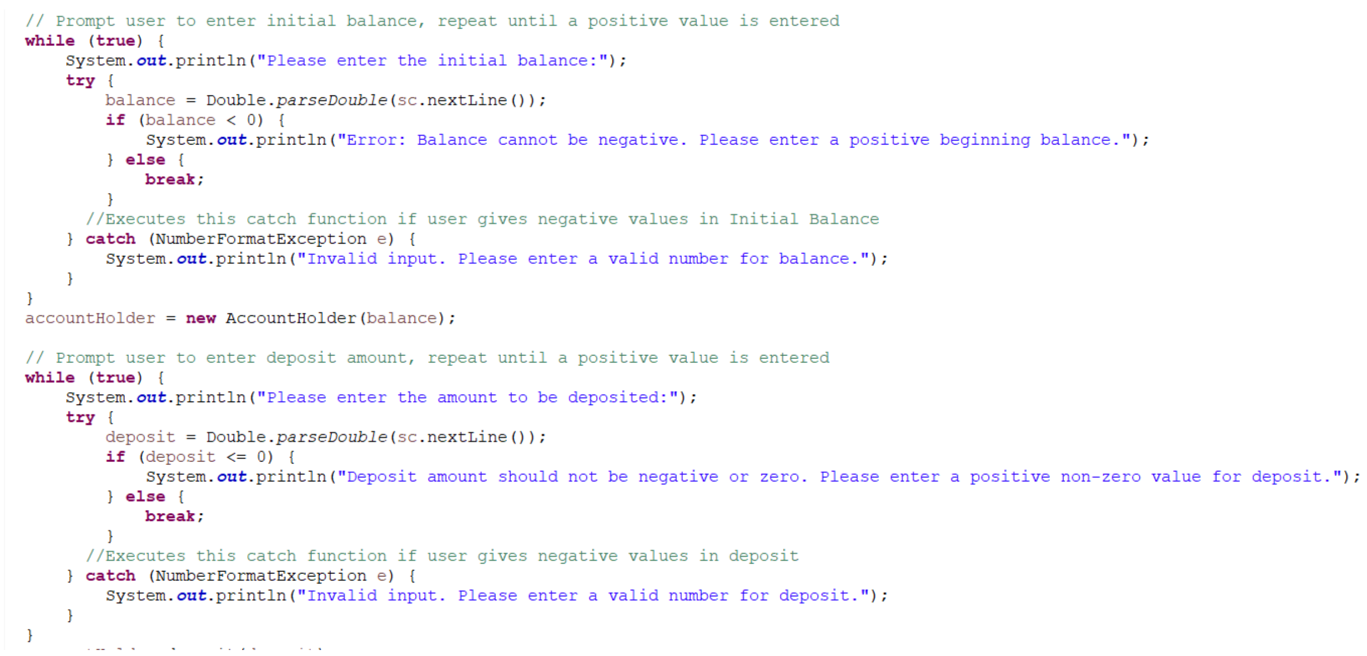
AccountHolder.Java

```
package com.bankproject.Guru;

import java.util.Scanner;

public class AccountHolder {

public static double annualInterestRate;

double balance;

Scanner sc = new Scanner(System.in);

public AccountHolder(double balance) {

while (balance < 0) {

System.out.println("Balance amount cannot be negative. Please enter a
positive value: ");

//Takes input from user and throws error if user gives negative value

balance = sc.nextDouble();

}

this.balance = balance;

}

public void deposit(double depositAmount) {

while (depositAmount <= 0) {

//Checks if value is greater than or equal to zero and throws error to re-
enter the value
```

```
System.out.println("Deposit amount should not be negative or zero. Please,
reenter a positive non-zero value for deposit: ");

Scanner sc = new Scanner(System.in);

depositAmount = sc.nextDouble();

}

//Here the balnce will be updated with deposit amount

balance += depositAmount;

}

public void withdraw(double withdrawalAmount) {

while (withdrawalAmount <= 0) {

System.out.println("Withdrawal amount cannot be negative or zero. Enter a
positive non-zero value for withdrawal: ");

withdrawalAmount = sc.nextDouble();

}

if (balance - withdrawalAmount < 50) {

System.out.println("Withdraw request cannot be processed - balance will drop
below $50" + "\n" + "balance: " + balance);

} else {

//Here the balance will deducted from withdrawal amount

balance -= withdrawalAmount;

}}

public void monthlyInterest() {

double monthlyInterest = annualInterestRate / 12.0;

System.out.println("Month\t\tNew Balance");

double balanceAtBeginning = balance;

//The iteration starts for 12 months interest and prints it in the console

for (int i = 1; i <= 12; i++) {
```

```
balance += balance * monthlyInterest;

if (i == 1) {

//Here the initial balance will print before executing the monthly updated
balance+interest rate

System.out.println("Base\t\t$" + balanceAtBeginning);

}//Here it prints the final balance after adding interest for 12 months

System.out.println(i + "\t\t$" + balance);

}

}public double getBalance() {

return balance;

}

}
```

2. Exception Handling:



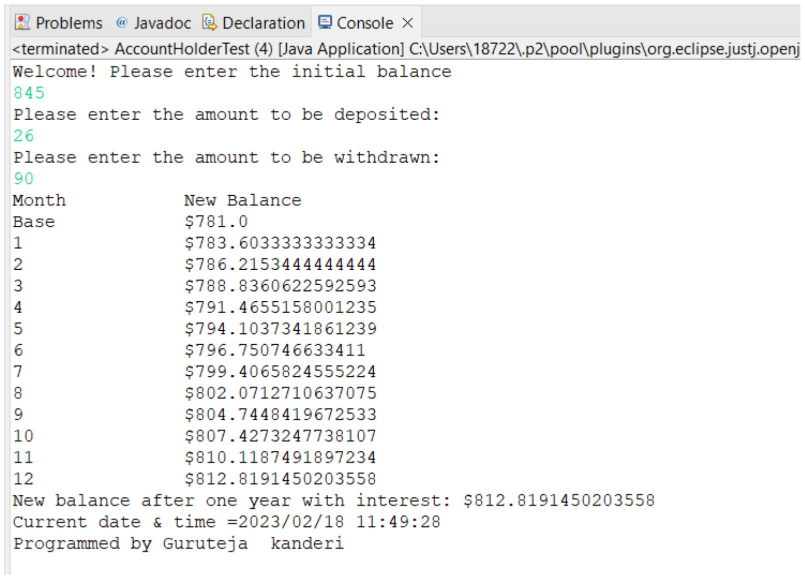
```
// Prompt user to enter initial balance, repeat until a positive value is entered
while (true) {
    System.out.println("Please enter the initial balance:");
    try {
        balance = Double.parseDouble(sc.nextLine());
        if (balance < 0) {
            System.out.println("Error: Balance cannot be negative. Please enter a positive beginning balance.");
        } else {
            break;
        }
      //Executes this catch function if user gives negative values in Initial Balance
    } catch (NumberFormatException e) {
        System.out.println("Invalid input. Please enter a valid number for balance.");
    }
}
accountHolder = new AccountHolder(balance);

// Prompt user to enter deposit amount, repeat until a positive value is entered
while (true) {
    System.out.println("Please enter the amount to be deposited:");
    try {
        deposit = Double.parseDouble(sc.nextLine());
        if (deposit <= 0) {
            System.out.println("Deposit amount should not be negative or zero. Please enter a positive non-zero value for deposit.");
        } else {
            break;
        }
      //Executes this catch function if user gives negative values in deposit
    } catch (NumberFormatException e) {
        System.out.println("Invalid input. Please enter a valid number for deposit.");
    }
}
```

This code repeatedly prompts the user to enter a valid positive number for balance. It checks if the input is valid and positive, and if not, asks the user to enter a valid number again using the try and catch methods.

AASIGNMENT-1-OOAD-Guruteja_Kanderi-A20526883

3. Outputs:

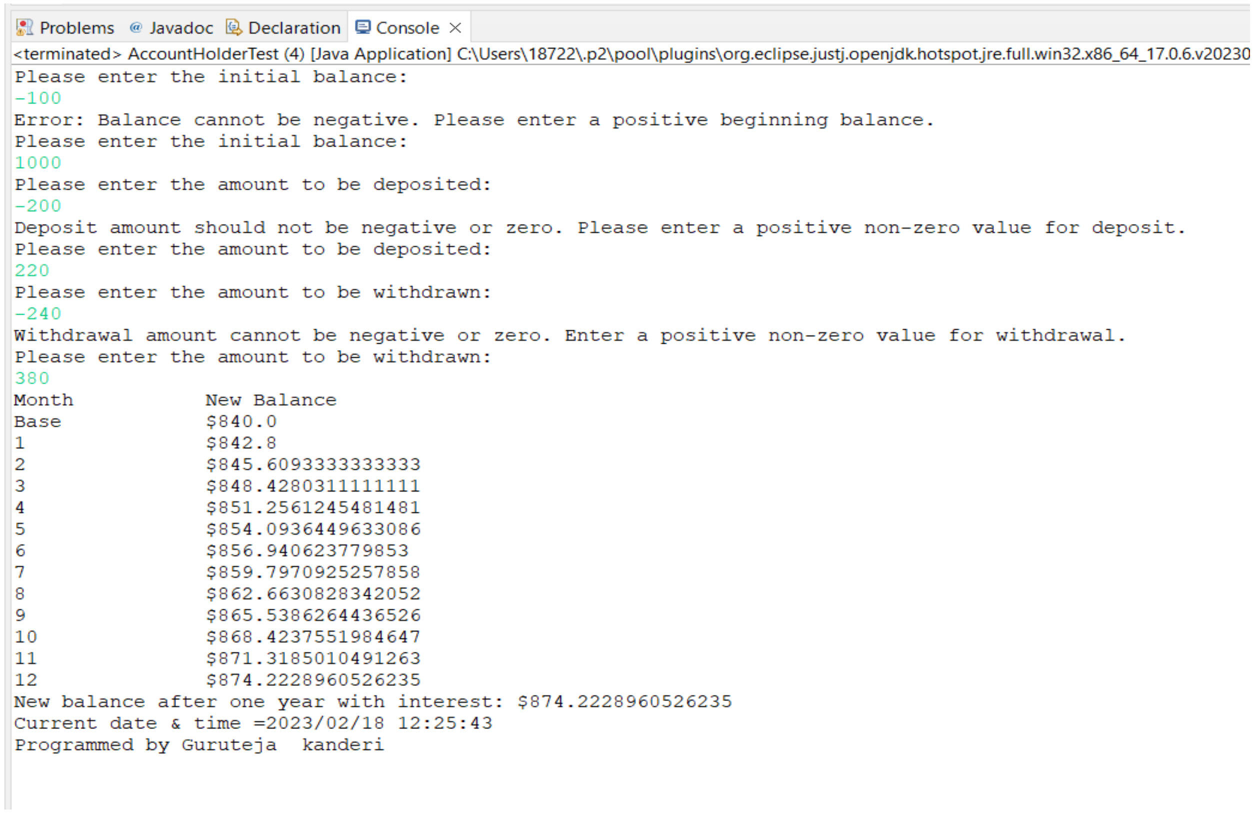
First output with all positive numbers: The Output for the program which has all the positive numbers showing the calculated interests for each month and finally the latest balance adding the original balance and interests.



Problems Javadoc Declaration Console

```
<terminated> AccountHolderTest (4) [Java Application] C:\Users\18722\.p2\pool\plugins\org.eclipse.justj.openj
Welcome! Please enter the initial balance
845
Please enter the amount to be deposited:
26
Please enter the amount to be withdrawn:
90
Month           New Balance
Base            $781.0
1               $783.6033333333334
2               $786.2153444444444
3               $788.8360622592593
4               $791.4655158001235
5               $794.1037341861239
6               $796.750746633411
7               $799.4065824555224
8               $802.0712710637075
9               $804.7448419672533
10              $807.4273247738107
11              $810.1187491897234
12              $812.8191450203558
New balance after one year with interest: $812.8191450203558
Current date & time =2023/02/18 11:49:28
Programmed by Guruteja  kanderi
```

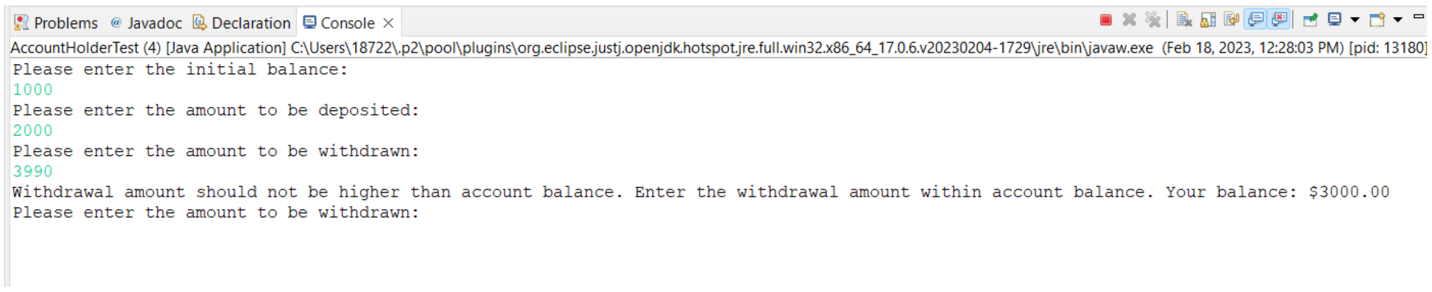
Output 2 with negative values: If we provide negative values it will pop up for enter the positive value again. So that user will realize and enters positive value.



Problems Javadoc Declaration Console

```
<terminated> AccountHolderTest (4) [Java Application] C:\Users\18722\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230
Please enter the initial balance:
-100
Error: Balance cannot be negative. Please enter a positive beginning balance.
Please enter the initial balance:
1000
Please enter the amount to be deposited:
-200
Deposit amount should not be negative or zero. Please enter a positive non-zero value for deposit.
Please enter the amount to be deposited:
220
Please enter the amount to be withdrawn:
-240
Withdrawal amount cannot be negative or zero. Enter a positive non-zero value for withdrawal.
Please enter the amount to be withdrawn:
380
Month           New Balance
Base            $840.0
1               $842.8
2               $845.6093333333333
3               $848.4280311111111
4               $851.2561245481481
5               $854.0936449633086
6               $856.940623779853
7               $859.7970925257858
8               $862.6630828342052
9               $865.5386264436526
10              $868.4237551984647
11              $871.3185010491263
12              $874.2228960526235
New balance after one year with interest: $874.2228960526235
Current date & time =2023/02/18 12:25:43
Programmed by Guruteja  kanderi
```

Output 3: when withdrawal amount is higher than account balance: If the user expects the withdrawal amount which makes the account balance less than $50. It throws the error saying the balance should not be less than $ 50 and enter a new withdrawal amount to withdrawal. Working as expected.



Problems Javadoc Declaration Console

```
AccountHolderTest (4) [Java Application] C:\Users\18722\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\bin\javaw.exe (Feb 18, 2023, 12:28:03 PM) [pid: 13180]
Please enter the initial balance:
1000
Please enter the amount to be deposited:
2000
Please enter the amount to be withdrawn:
3990
Withdrawal amount should not be higher than account balance. Enter the withdrawal amount within account balance. Your balance: $3000.00
Please enter the amount to be withdrawn:
```

4. Explanation of the classes AccountTestHolder and AccountHolder:

The code defines two classes - AccountHolderTest and AccountHolder. The AccountHolderTest class contains the main method, which is the starting point of the program, while the AccountHolder class defines the properties and methods of an account holder. The main method uses a Scanner object to read input from the console. It first asks the user for an initial balance, and then prompts for deposit and withdrawal amounts.

The AccountHolder class has two instance variables - balance and annualInterestRate. The balance variable stores the current balance of the account holder, while the annualInterestRate variable is a static variable that stores the annual interest rate for all accounts.

It has three methods - deposit, withdraw, and monthlyInterest. The deposit method adds a specified amount to the account balance, while the withdraw method subtracts a specified amount from the account balance. The monthlyInterest method calculates and prints the balance after each month's interest has been applied.

The AccountHolder constructor takes an initial balance as an argument and checks if it is positive. If the initial balance is negative, an error message is printed, and the balance is not set.

The deposit method takes a deposit amount as an argument and checks if it is positive. If the deposit amount is negative, an error message is printed, and the deposit is not added to the balance.

The withdraw method takes a withdrawal amount as an argument and checks if the balance will drop below $50 after the withdrawal. If the balance will drop below $50, an error message is printed, and the withdrawal is not processed.

The monthlyInterest method calculates the balance after each month's interest has been applied. It uses a for loop to iterate through each month of the year and adds the monthly interest to the balance.

The main method creates an instance of the AccountHolder class with the initial balance provided by the user. It then calls the deposit and withdraw methods with the deposit and withdrawal amounts provided by the user. Finally, it calls the monthlyInterest method and prints the total balance after 12 months of interest has been applied. And it prints the current date and time along with the name of the programmer at the end.