CTU training solutions CAREER SUCCESS STARTS AT CTU

<table>
<tr><td colspan="10">Faculty of Information Technology</td></tr>
<tr><td rowspan="5">I declare that I am familiar with, and will abide to the Examination rules of CTU

 
Signature</td><td colspan="9">SUBJECT NAME: Beginner Java
SUBJECT CODE: JD521</td></tr>
<tr><td colspan="4">FORMATIVE ASSESSMENT
Duration: Jul 18 – 31 Aug
Date 31 August 2023
Total Marks: 100
Total pages:</td><td colspan="5">Examiner: Mr. Junior Mangayi
Moderator: Mr. Newton</td></tr>
<tr><td colspan="9">Student number</td></tr>
<tr><td>2</td><td>0</td><td>2</td><td>3</td><td>2</td><td>7</td><td>5</td><td></td><td></td></tr>
<tr><td colspan="4">Surname:
Poponi</td><td colspan="3">Initials:
M.P</td><td>/</td><td>%</td></tr>
</table>

Table of Contents

1.1 Create a basic Java Application that will help CTU banking clients to perform basic transactions. The Application should meet the following menu requirements:

- Display all account details
- Deposit the amount
- Save the amount
- Withdraw the amount
- Exit

QUESTION 1

Coding of the problem domain

For this section I will be displaying code snippets of any methods and key implementations made that meet the requirements. The source code is included with the entirety of the code showing all the logic with comments saying what each method is for.

```
// Used to import the scanner class from the java.util
import java.util.Scanner;

public class sample {
  // Initial account balance
  private static double balance = 0;

  public static void main(String[] args) {
    // Create a SavingsAccount object
    SavingsAccount account = new SavingsAccount();
    // Allows the user to input choice
    Scanner scanner = new Scanner(System.in);
    boolean exit = false;

    while (!exit) {
      // Display the menu options
      displayMenu();
      // Read user choice
      int choice = scanner.nextInt();

      switch (choice) {
        case 1:
          // Print account details using toString method
          System.out.println(account.toString());
         // Display account details using displayAccountDetails method
```

```
            displayAccountDetails();
            break;
        case 2:
            System.out.print("Enter the amount to deposit: ");
            double depositAmount = scanner.nextDouble();
            // Call deposit method with user input
            deposit(depositAmount);
            break;
        case 3:
            System.out.print("Enter the amount to save: ");
            double saveAmount = scanner.nextDouble();
            System.out.print("Enter the savings duration in months: ");
            int duration = scanner.nextInt();
            // Call save method with user input
            save(saveAmount, duration);
            break;
        case 4:
            System.out.print("Enter the amount to withdraw: ");
            double withdrawAmount = scanner.nextDouble();
            // Call withdraw method with user input
            withdraw(withdrawAmount);
            break;
        case 5:
            exit = true;
            System.out.println("Thanks for banking with CTU, have a geat day.");
            break;
        default:
            System.out.println("Invalid choice. Please try again.");
            break;
      }
    }
}
// Used for displaying user choices
private static void displayMenu() {
    System.out.println("========== Savings Account Menu ==========");
    System.out.println("1. Display Account Details");
    System.out.println("2. Deposit");
    System.out.println("3. Save");
    System.out.println("4. Withdraw");
    System.out.println("5. Exit");
    System.out.print("Enter your choice:");
}

private static void displayAccountDetails() {
    System.out.println("Account Balance: R" + balance);
}
```

```
  private static void deposit(double amount) {
    // Add deposited amount to balance
    balance += amount;
    System.out.println("Deposit successful. Account balance: R" + balance);
  }

  private static void withdraw(double amount) {
    // Check if sufficient funds are available
    if (balance >= amount) {
      // Subtract withdrawal amount from balance
      balance -= amount;
      System.out.println("Withdrawal successful. Account balance: R" +
balance);
    } else {
      System.out.println("Insufficient funds to make the withdrawal.");
    }
  }
  // Calculate the interset earned depending on the amount saved
  private static void save(double amount, int duration) {
    double interestRate;
    if (amount >= 100 && amount <= 500)
      interestRate = 0.005;
    else if (amount > 500 && amount <= 1000)
      interestRate = 0.02;
    else if (amount > 1000)
      interestRate = 0.05;
    else {
      System.out.println("Invalid savings amount.");
      // Exit the method if savings amount is invalid
      return;
    }

    // Calculate interest earned
    double interestEarned = amount * interestRate * duration;
    // Add savings amount and interest to balance
    balance += (amount + interestEarned);

    System.out.println("Savings successful. Interest earned: R" +
interestEarned);
    System.out.println("Account balance: R" + balance);
  }
}
```

```
public class SavingsAccount {

  String Name = "Mzukis Poponi";
  String Account_type = "Savings Account";
  float Account_number = 123846;

  public String toString() {

    return "Name : " + Name + "\n" + "Account Type: " + Account_type + "\n" +
"Account Number: " + Account_number;
  }
}
```

toString () is a special method that all object inherit, that return a string that “textually represents” an object. In the scenario above it is used implicitly. This is used for printing the user’s details in choice 1.

1. CTU ATM display screen

```
// Used for displaying user choices
private static void displayMenu() {
  System.out.println(x:"========== Savings Account Menu ==========");
  System.out.println(x:"1. Display Account Details");
  System.out.println(x:"2. Deposit");
  System.out.println(x:"3. Save");
  System.out.println(x:"4. Withdraw");
  System.out.println(x:"5. Exit");
  System.out.print(s:"Enter your choice:");
}
```

This snippet gives the user the opportunity to select the desired option

```
========== Savings Account Menu ==========
1. Display Account Details
2. Deposit
3. Save
4. Withdraw
5. Exit
Enter your choice:
```

1. Display all account details

```
switch (choice) {
  case 1:
    // Print account details using toString method
    System.out.println(account.toString());
   // Display account details using displayAccountDetails method
    displayAccountDetails();
    break;
```

```
private static void displayAccountDetails() {
  System.out.println("Account Balance: R" + balance);
}
```

```
1
2  public class SavingsAccount {
3
4    String Name = "Mzukis Poponi";
5    String Account_type = "Savings Account";
6    float Account_number = 123846;
7
8    public String toString() {
9
10     return "Name : " + Name + "\n" + "Account Type: " + Account_type + "\n" + "Account Number: " + Account_number;
11   }
12 }
```

This code is from the class that deals with getting the users data/information from the account class.

```
2. Deposit
3. Save
4. Withdraw
5. Exit
Enter your choice:1
Name : Mzukis Poponi
Account Type: Savings Account
Account Number: 123846.0
Account Balance: R0.0
---------- Savings Account Menu ----------
```

2. Deposit the amount.

```
case 2:
    System.out.print(s:"Enter the amount to deposit: ");
    double depositAmount = scanner.nextDouble();
    // Call deposit method with user input
    deposit(depositAmount);
    break;
```

```
private static void deposit(double amount) {
    // Add deposited amount to balance
    balance += amount;
    System.out.println("Deposit successful. Account balance: R" + balance);
}
```

This snippet of code is responsible for depositing money into the users account, it find the users balance and updates it.

```
1. Display Account Details
2. Deposit
3. Save
4. Withdraw
5. Exit
Enter your choice:2
Enter the amount to deposit: 400
Deposit successful. Account balance: R400.0
```

3. Save the amount

```
case 3:
    System.out.print(s:"Enter the amount to save: ");
    double saveAmount = scanner.nextDouble();
    System.out.print(s:"Enter the savings duration in months: ");
    int duration = scanner.nextInt();
    // Call save method with user input
    save(saveAmount, duration);
    break;
```

```java
// Calculate the interset earned depending on the amount saved
private static void save(double amount, int duration) {
  double interestRate;
  if (amount >= 100 && amount <= 500)
    interestRate = 0.005;
  else if (amount > 500 && amount <= 1000)
    interestRate = 0.02;
  else if (amount > 1000)
    interestRate = 0.05;
  else {
    System.out.println(x:"Invalid savings amount.");
    // Exit the method if savings amount is invalid
    return;
  }

  // Calculate interest earned
  double interestEarned = amount * interestRate * duration;
  // Add savings amount and interest to balance
  balance += (amount + interestEarned);

  System.out.println("Savings successful. Interest earned: R" + interestEarned);
  System.out.println("Account balance: R" + balance);
}
```

This code snippet is used to perform calculations when the client saves money, depending how much money the clients saves and for how long.

```
1. Display Account Details
2. Deposit
3. Save
4. Withdraw
5. Exit
Enter your choice:3
Enter the amount to save: 700
Enter the savings duration in months: 1
Savings successful. Interest earned: R14.0
Account balance: R714.0
```

4. Withdraw the amount

```
case 4:
    System.out.print(s:"Enter the amount to withdraw: ");
    double withdrawAmount = scanner.nextDouble();
    // Call withdraw method with user input
    withdraw(withdrawAmount);
    break;
```

```
private static void withdraw(double amount) {
    // Check if sufficient funds are available
    if (balance >= amount) {
        // Subtract withdrawal amount from balance
        balance -= amount;
        System.out.println("Withdrawal successful. Account balance: R" + balance);
    } else {
        System.out.println(x:"Insufficient funds to make the withdrawal.");
    }
}
```

This code snippet is responsible for cash withdrawals from a user's account and will connect to the balance for the updates.

```
1. Display Account Details
2. Deposit
3. Save
4. Withdraw
5. Exit
Enter your choice:4
Enter the amount to withdraw: 350
Withdrawal successful. Account balance: R364.0
```

5. Exit

```
case 5:
  exit = true;
  System.out.println(x:"Thanks for banking with CTU, have a geat day.");
  break;
```

This code snippet exits the code.

```
========== Savings Account Menu ==========
1. Display Account Details
2. Deposit
3. Save
4. Withdraw
5. Exit
Enter your choice:5
Thanks for banking with CTU, have a geat day.
```

```
default:
  System.out.println(x:"Invalid choice. Please try again.");
  break;
```

This code snippet alert the user if they enter a number that is not displayed on the screen.

```
1. Display Account Details
2. Deposit
3. Save
4. Withdraw
5. Exit
Enter your choice:6
Invalid choice. Please try again.
```