

COSC 2006 Assignment 1

Test Data

There are two parts of the test data. First, the bank.txt file that you must use to read the data that you will be processing. This file is organized as stated in the assignment document. If on your chosen platform you see any strange characters in this file, you may edit them out. However, in NO way may you change the overall format of the file or the other order of the data itself.

The queries you must perform on the file are as follows. There are no typos here, any errors must be handled in your program by printing an appropriate message.

1. perform an S command, with 275 for the account number, and X for the account type

The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer view displays several Java source files and a text file named 'afidata.txt'. In the center, the code editor shows a Java program named 'BankProgram.java' with the following content:

```
82
83     //This method prints the total balances for each customer.
84     private static void printTotalBalances() {
85         selectionSort();
86         int currentCustomer = customerNumbers[0];
87         double totalBalance = balances[0];
88         for (int i = 1; i < numAccounts; i++) {
89             if (customerNumbers[i] == currentCustomer) {
90                 totalBalance += balances[i];
91             } else {
92                 System.out.printf("%d\t%.2f\n", currentCustomer, totalBalance);
93                 currentCustomer = customerNumbers[i];
94                 totalBalance = balances[i];
95             }
96         }
97         System.out.printf("%d\t%.2f\n", currentCustomer, totalBalance);
98     }
99
100    //This method performs a simple selection sort on the accounts based on customer numbers.
101    private static void selectionSort() {
102        for (int i = 0; i < numAccounts - 1; i++) {
103            int minIndex = i;
104            for (int j = i + 1; j < numAccounts; j++) {
105                if (customerNumbers[j] < customerNumbers[minIndex]) {
106                    minIndex = j;
107                }
108            }
109            if (minIndex != i) {
110                swap(i, minIndex);
111            }
112        }
113    }
114}
```

On the right, the terminal window shows the execution of the program and its output:

```
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit;
S
Enter customer number (0 for don't care): 275
Enter account type (X for don't care): X
275  C   256.67
275  S   484.48
275  M   480.00
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit;
```

2. perform an S command, with 384 for the account number, and X for the account type

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows multiple Java projects including COSC1046_Lab1 through COSC1047_Lab9, COSC2006_Ass1, and COSC2006_Assignment1.
- Code Editor:** Displays the `BankProgram.java` file with Java code for searching accounts by customer number and account type.
- Console:** Shows the output of the Java application. The user enters "S" to search for account 384, which is found. The user then enters "X" for account type, but the application prints "No accounts found." because the account number 384 was already found.

```

    package com.example;
    import java.util.Scanner;
    public class BankProgram {
        private static void searchForAccount() {
            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter customer number (0 for don't care): ");
            int customerNumber = scanner.nextInt();
            System.out.print("Enter account type (X for don't care): ");
            char accountType = scanner.next().charAt(0);
            boolean found = false;
            for (int i = 0; i < numAccounts; i++) {
                if ((customerNumber == 0 || customerNumbers[i] == customerNumber) && (accountType == 'X' || accountTypes[i] == accountType)) {
                    System.out.printf("%d\t%s\t%.2f\n", customerNumbers[i], accountTypes[i], balances[i]);
                    found = true;
                }
            }
            if (!found) {
                System.out.println("No accounts found.");
            }
        }
        //This method prints the total balances for each customer.
        private static void printTotalBalances() {
            selectionSort();
            int currentCustomer = customerNumbers[0];
            double totalBalance = balances[0];
            for (int i = 1; i < numAccounts; i++) {
                if (customerNumbers[i] > currentCustomer) {
                    totalBalance += balances[i];
                } else {
                    System.out.printf("%d\t%.2f\n", currentCustomer, totalBalance);
                    currentCustomer = customerNumbers[i];
                    totalBalance = balances[i];
                }
            }
            System.out.printf("%d\t%.2f\n", currentCustomer, totalBalance);
        }
        //This method performs a simple selection sort on the accounts based on customer numbers.
        private static void selectionSort() {
            for (int i = 0; i < numAccounts - 1; i++) {
                for (int j = i + 1; j < numAccounts; j++) {
                    if (customerNumbers[i] > customerNumbers[j]) {
                        int tempCustomerNumber = customerNumbers[i];
                        customerNumbers[i] = customerNumbers[j];
                        customerNumbers[j] = tempCustomerNumber;
                        String tempAccountType = accountTypes[i];
                        accountTypes[i] = accountTypes[j];
                        accountTypes[j] = tempAccountType;
                        double tempBalance = balances[i];
                        balances[i] = balances[j];
                        balances[j] = tempBalance;
                    }
                }
            }
        }
    }
  
```

3. perform an S command, with 6 for the account number, and X for the account type

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows multiple Java projects including COSC1046_Lab1 through COSC1047_Lab9, COSC2006_Ass1, and COSC2006_Assignment1.
- Code Editor:** Displays the `BankProgram.java` file with Java code for reading data from a file and searching accounts by customer number and account type.
- Console:** Shows the output of the Java application. The user enters "S" to search for account 6, which is found. The user then enters "X" for account type, and the application correctly prints "No accounts found." because the account number 6 was already found.

```

    package com.example;
    import java.io.File;
    import java.util.Scanner;
    public class BankProgram {
        private static void readDataFromFile() {
            try {
                Scanner scanner = new Scanner(new File("/Users/apple/Documents/AlgomaU_CS/SEM2/COSC2006/ASS1/adata.txt"));
                while (scanner.hasNextLine()) {
                    customerNumbers[numAccounts] = Integer.parseInt(scanner.nextLine());
                    accountTypes[numAccounts] = scanner.nextLine().charAt(0);
                    balances[numAccounts] = Double.parseDouble(scanner.nextLine());
                    numAccounts++;
                }
                scanner.close();
            } catch (FileNotFoundException e) {
                System.out.println("File not found.");
            }
        }
        //This method allows the user to search for accounts based on customer number and account type.
        private static void search() {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter customer number (0 for don't care): ");
            int customerNumber = sc.nextInt();
            System.out.print("Enter account type (X for don't care): ");
            char accountType = sc.next().charAt(0);
            boolean found = false;
            for (int i = 0; i < numAccounts; i++) {
                if ((customerNumber == 0 || customerNumbers[i] == customerNumber) && (accountType == 'X' || accountTypes[i] == accountType)) {
                    System.out.printf("%d\t%s\t%.2f\n", customerNumbers[i], accountTypes[i], balances[i]);
                    found = true;
                }
            }
            if (!found) {
                System.out.println("No accounts found.");
            }
        }
    }
  
```

4. perform an S command, with 1234 for the account number, and S for the account type

```

private static void readuserdatafromfile() {
    try {
        Scanner scanner = new Scanner(new File("/Users/apple/Documents/AlgomaU_CS/SEM2/COSC2006/ASS1/adata.txt"));
        while (scanner.hasNextLine()) {
            customerNumbers[numAccounts] = Integer.parseInt(scanner.nextLine());
            accountTypes[numAccounts] = scanner.nextLine().charAt(0);
            balances[numAccounts] = Double.parseDouble(scanner.nextLine());
            numAccounts++;
        }
        scanner.close();
    } catch (FileNotFoundException e) {
        System.out.println("File not found.");
    }
}

//This method allows the user to search for accounts based on customer number and account type.
private static void search() {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter customer number (0 for don't care): ");
    int customerNumber = sc.nextInt();
    System.out.print("Enter account type (X for don't care): ");
    char accountType = sc.next().charAt(0);
    boolean found = false;
    for (int i = 0; i < numAccounts; i++) {
        if ((customerNumber == 0 || customerNumbers[i] == customerNumber)
                && (accountType == 'X' || accountTypes[i] == accountType)) {
            System.out.printf("%d\t%.2f\n", customerNumbers[i], balances[i]);
            found = true;
        }
    }
    if (!found) {
        System.out.println("No accounts found.");
    }
}

```

BankProgram [3] (Java Application) [/Library/Java/VirtualMachines/temurin-17.jdk/Contents/Home/bin/java (22-Sep-2023, 1:27:07 pm) [pid: 2036]

Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:
S

Enter customer number (0 for don't care): **1234**
Enter account type (X for don't care): **S**
1234 12.34
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:

5. perform an E command (i.e. enter E when prompted for a command)

```

private static void readuserdatafromfile() {
    try {
        Scanner scanner = new Scanner(new File("/Users/apple/Documents/AlgomaU_CS/SEM2/COSC2006/ASS1/adata.txt"));
        while (scanner.hasNextLine()) {
            customerNumbers[numAccounts] = Integer.parseInt(scanner.nextLine());
            accountTypes[numAccounts] = scanner.nextLine().charAt(0);
            balances[numAccounts] = Double.parseDouble(scanner.nextLine());
            numAccounts++;
        }
        scanner.close();
    } catch (FileNotFoundException e) {
        System.out.println("File not found.");
    }
}

//This method allows the user to search for accounts based on customer number and account type.
private static void search() {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter customer number (0 for don't care): ");
    int customerNumber = sc.nextInt();
    System.out.print("Enter account type (X for don't care): ");
    char accountType = sc.next().charAt(0);
    boolean found = false;
    for (int i = 0; i < numAccounts; i++) {
        if ((customerNumber == 0 || customerNumbers[i] == customerNumber)
                && (accountType == 'X' || accountTypes[i] == accountType)) {
            System.out.printf("%d\t%.2f\n", customerNumbers[i], balances[i]);
            found = true;
        }
    }
    if (!found) {
        System.out.println("No accounts found.");
    }
}

```

BankProgram [3] (Java Application) [/Library/Java/VirtualMachines/temurin-17.jdk/Contents/Home/bin/java (22-Sep-2023, 1:27:31 pm) [pid: 2050]

Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:
E

Invalid command. Please try again.
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:

6. perform an P command,

The screenshot shows the Eclipse IDE interface with the Java EE perspective selected. The Package Explorer view on the left lists various Java files and resources. The BankProgram.java file is open in the editor, showing code for reading account data from a file and searching for accounts based on customer number and account type.

```

private static void readdatafromfile() {
    try {
        Scanner scanner = new Scanner(new File("/Users/apple/Documents/AlgomaU_CS/SEM2/COSC2006/ASS1/adata.txt"));
        while (scanner.hasNextLine()) {
            customerNumbers[numAccounts] = Integer.parseInt(scanner.nextLine());
            accountTypes[numAccounts] = scanner.nextLine().charAt(0);
            balances[numAccounts] = Double.parseDouble(scanner.nextLine());
            numAccounts++;
        }
        scanner.close();
    } catch (FileNotFoundException e) {
        System.out.println("File not found.");
    }
}

//This method allows the user to search for accounts based on customer number and account type.
private static void search() {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter customer number (0 for don't care): ");
    int customerNumber = sc.nextInt();
    System.out.print("Enter account type (X for don't care): ");
    char accountType = sc.next().charAt(0);
    boolean found = false;
    for (int i = 0; i < numAccounts; i++) {
        if ((customerNumber == 0 || customerNumbers[i] == customerNumber)
                && (accountType == 'X' || accountTypes[i] == accountType)) {
            System.out.printf("%d\t%.2f\n", customerNumbers[i], balances[i]);
            found = true;
        }
    }
    if (!found) {
        System.out.println("No accounts found.");
    }
}

```

The Console tab at the bottom displays the output of the program. It prints the total balance for each account in the file, followed by a prompt for the user to enter a customer number and account type to search for.

```

BankProgram (3) [Java Application] /Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java (22-Sep-2023, 1:28:05 pm) [pid: 2059]
Print Q to Quit:
P
1 124.46
12 2.46
47 29.29
275 1225.65
384 39839.39
1234 4292.83
2922 23224.43
2923 29.29
2929 3929.20
2958 398.47
3342 4241.47
3391 3849.40
4759 39.29
11257 2011.91
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:

```

7. perform an S command, with 0 for the account number, and C for the account type

This screenshot shows the same Eclipse IDE setup as the previous one, but the user has entered the 'S' command to search for an account. The console output shows the user entering '0' for the customer number and 'C' for the account type, which triggers the search logic in the program.

```

BankProgram (3) [Java Application] /Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java (22-Sep-2023, 1:28:56 pm) [pid: 2066]
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:
S
Enter customer number (0 for don't care): 0
Enter account type (X for don't care): C
11257 C 11.59
275 C 256.67
3342 C 3848.14
1234 C 3958.20
3391 C 3849.38
2958 C 398.47
47 C 39.00
1 C 1.81
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:

```

8. perform an S command, with 0 for the account number, and M for the account type

The screenshot shows the Eclipse IDE interface with the following details:

- Left Side (Package Explorer):** Shows the project structure with various Java files and folders like COSC1046_Lab1 through COSC1047_Lab10, COSC2006_Ass1, and TestSF.
- Middle (Code Editor):** Displays the `BankProgram.java` code. The code reads data from a file named `alldata.txt` and allows users to search for accounts by customer number and account type, or print total balances.
- Bottom (Console View):** Shows the output of the program. The user enters "S" to search for an account. The program prints the following results:

```

Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:
S
Enter customer number (0 for don't care): 0
Enter account type (X for don't care): M
1125 M 2359.29
4769 M 2359.29
275 M 484.50
2922 M 23324.43
2923 M 1.00
1234 M 334.34
    
```

9. perform an Q command to quit the program

The screenshot shows the Eclipse IDE interface with the following details:

- Left Side (Package Explorer):** Shows the project structure with various Java files and folders like COSC1046_Lab1 through COSC1047_Lab10, COSC2006_Ass1, and TestSF.
- Middle (Code Editor):** Displays the `BankProgram.java` code. The code includes a switch statement for commands S, P, and Q, and a method to read data from a file.
- Bottom (Console View):** Shows the output of the program. The user enters "Q" to quit the program. The program prints:

```

<terminated> BankProgram [3] [Java Application] [/Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java (22-Sep-2023, 1:34:17 pm - 1:34:19 pm) [pid: 2124]
Enter command to operate:
Print S to Search for an account;
Print P to Print Total Balances;
Print Q to Quit:
Q
Assignment 1 ends, Goodbye, have a great day!
    
```