Programming and Data Structures
Programming Project 2: Object Oriented Programming

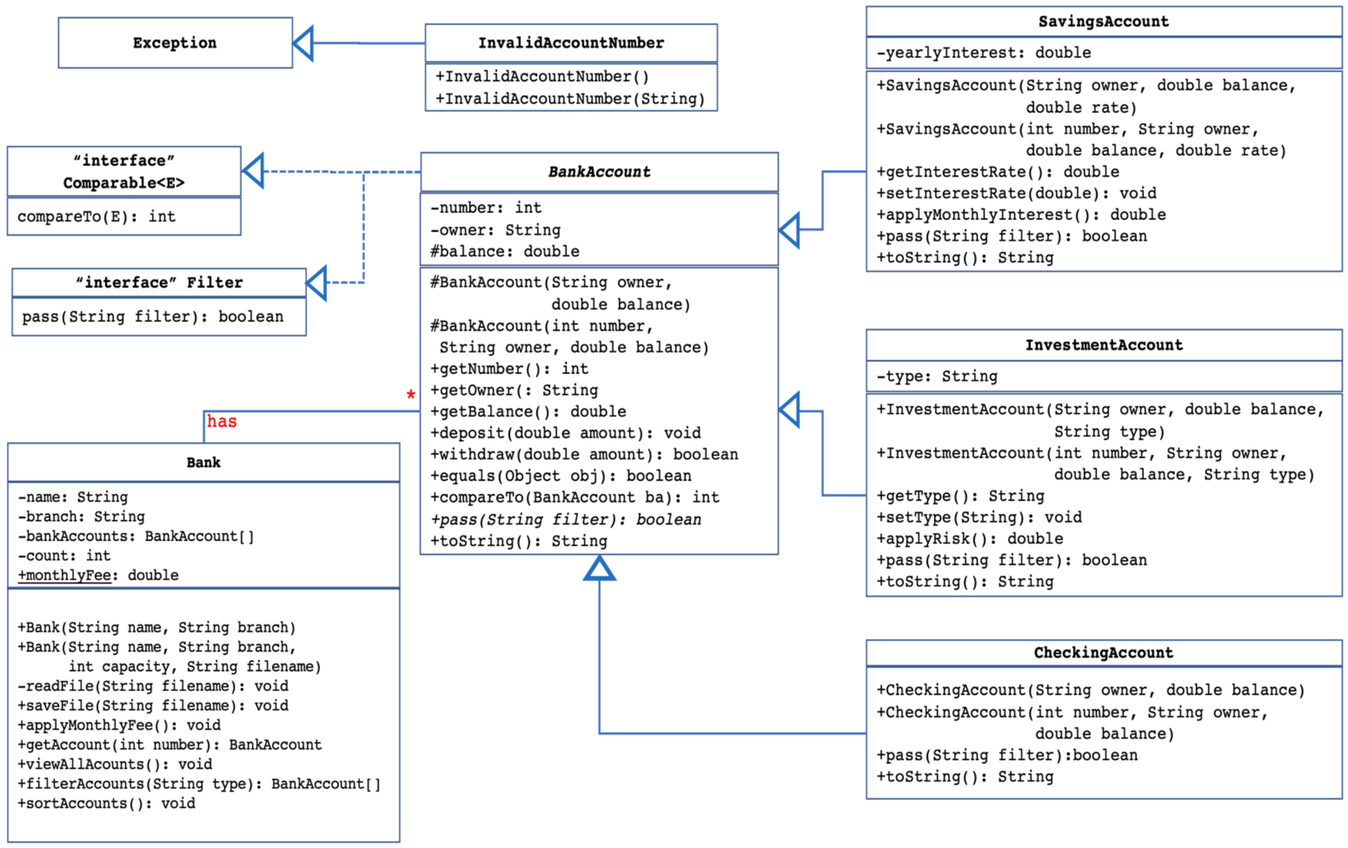
Objectives of the project

Students should demonstrate the following abilities:

1. Create classes in Java with the required data members and methods
2. Extend a class using inheritance to model common behavior between related classes
3. Create an interface to model common behavior between unrelated classes
4. Implement multiple inheritance using interfaces
5. Use polymorphism to manipulate subtype instances using the supertype

Project

Create the classes and the relationships shown in the UML diagram below.


Exception
InvalidAccountNumber
+InvalidAccountNumber()
+InvalidAccountNumber(String)
"interface" Comparable<E>
compareTo(E): int
"interface" Filter
pass(String filter): boolean
BankAccount
-number: int
-owner: String
#balance: double
#BankAccount(String owner, double balance)
#BankAccount(int number, String owner, double balance)
+getNumber(): int
+getOwner(: String
+getBalance(): double
+deposit(double amount): void
+withdraw(double amount): boolean
+equals(Object obj): boolean
+compareTo(BankAccount ba): int
+pass(String filter): boolean
+toString(): String
has
*
Bank
-name: String
-branch: String
-bankAccounts: BankAccount[]
-count: int
+monthlyFee: double
+Bank(String name, String branch)
+Bank(String name, String branch, int capacity, String filename)
-readFile(String filename): void
+saveFile(String filename): void
+applyMonthlyFee(): void
+getAccount(int number): BankAccount
+viewAllAcounts(): void
+filterAccounts(String type): BankAccount[]
+sortAccounts(): void
SavingsAccount
-yearlyInterest: double
+SavingsAccount(String owner, double balance, double rate)
+SavingsAccount(int number, String owner, double balance, double rate)
+getInterestRate(): double
+setInterestRate(double): void
+applyMonthlyInterest(): double
+pass(String filter): boolean
+toString(): String
InvestmentAccount
-type: String
+InvestmentAccount(String owner, double balance, String type)
+InvestmentAccount(int number, String owner, double balance, String type)
+getType(): String
+setType(String): void
+applyRisk(): double
+pass(String filter): boolean
+toString(): String
CheckingAccount
+CheckingAccount(String owner, double balance)
+CheckingAccount(int number, String owner, double balance)
+pass(String filter):boolean
+toString(): String

Detailed descriptions of the classes are shown in the tables below.

`BankAccount`	**Abstract class**
`number: int`	Account number
`owner: String`	Name of the account owner
`balance: double`	Amount of money in the account
`BankAccount(int n, String o, double b)`	Assigns **`n`** to **`number`**, **`o`** to owner, and **`b`** to **`balance`**
`BankAccount(String o, double b)`	Assigns the parameters **`o`** and **`b`** to the data members **`owner`** and **`balance`**. It assigns a random integer composed of 6 digits to **`number`**.
`getOwner(): String`	Returns the name of the account owner
`getNumber(): int`	Returns the account number
`getBalance(): double`	Returns the balance of the account
`deposit(double amount): void`	Adds **`amount`** to the account balance
`withdraw(double amount): boolean`	Subtracts **`amount`** from the account balance if there is enough money and returns `true`, otherwise returns `false`.
`toString(): String`	Returns the information of the account (type, number, owner, balance)
`compareTo(BankAccount ba): int`	Compares the balances of two account numbers and returns 0 if the two balances are equal, 1 if the first balance is greater than the second balance, and -1 if the first balance is less than the second one.
`pass(String filter): boolean`	Returns **`true`** if the instance of the **`BankAccount`** object matches the value of **`filter`**. For example, if **`filter`** is equal to "**`Savings`**", the method returns **`true`** if the instance of the bank account is **`SavingsAccount`** or **`false`** otherwise. The method is abstract in class **`BankAccount`**.
`equals(Object o): boolean`	Returns true if two bank accounts have the same account number or false otherwise.

`CheckingAccount`	**Description**
`CheckingAccount(int n, String o, double b)`	Assigns **`n`** to **`number`**, **`o`** to **`owner`**, **`b`** to **`balance`** using the first constructor from class **`BankAccount`**.
`CheckingAccount(String o, double b)`	Invokes the second constructor in class **`BankAccount`** with the arguments **`o`** and **`b`**.
`pass(String filter): boolean`	Returns **`true`** if **`filter`** is equal to **`"Checking"`**
`toString(): String`	Returns the information of the account (type, number, owner, balance)

`SavingsAccount`	**Description**
`yearlyInterest: double`	The yearly interest rate as a percentage
`SavingsAccount(int n, String o, double b, double yi)`	Assigns **`n`** to **`number`**, **`o`** to **`owner`**, **`b`** to **`balance`** using the constructor from class **`BankAccount`**, and **`yi`** to **`yearlyInterest`**
`SavingsAccount(String o, double b, double yi)`	Invokes the second constructor in class BankAccount with the arguments **`o`** and **`b`** and assigns **`yi`** to **`yearlyInterest`** .
`getInterestRate(): double`	Returns the yearly interest
`setInterestRate(double yi): void`	Sets the yearly interest to **`yi`**
`applyMonthlyInterest(): double`	Returns the value of the monthly interest as `(balance * yearlyInterest/12)/100`. The monthly interest is added to balance before the method returns.
`pass(String filter): boolean`	Returns **`true`** if **`filter`** is equal to **`"Savings"`**
`toString(): String`	Returns the information of the account (type, number, owner, balance, yearly interest)

`InvestmentAccount`	**Description**
`type: String`	The investment type: Property, growth, or Shares
`InvestmentAccount(int n, String o, double b, String t)`	Assigns **`n`** to **`number`**, **`o`** to **`owner`**, **`b`** to **`balance`** using the constructor from class **`BankAccount`**, and t to **`type`**.
`InvestmentAccount(String o, double b, String t)`	Invokes the second constructor in class **`BankAccount`** with the arguments **`o`** and **`b`** and assigns **`t`** to **`type`**.
`getType(): String`	Returns the investment type
`setType(String t): void`	Sets the investment type to **`t`**

`applyRisk(): double`	Simulates the profit or loss on an investment account. The Method generates two random numbers **`risk`** and **`rate`** between 0 and 1, and calculates **`profitLoss`** as follows:
	`if (risk > 0.5) // profit case`
	`  profitLoss = balance * rate / 100;`
	`else // loss case`
	`  profitLoss = - balance * rate / 100;`
	The method adds the value of **`profitLoss`** to **`balance`** and returns the value **`profitLoss`**.

`pass(String filter): boolean`	Returns **`true`** if **`filter`** is equal to **`"Investment"`**
`toString(): String`	Returns the information of the account (type, number, owner, balance, investment type)

Bank	**Description**
`name: String`	Name of the bank
`branch: String`	Location of the bank branch
`bankAccounts: BankAccount[]`	List of the bank accounts
`monthlyFee: double`	Static member of the class that contains the value of a flat fee applied monthly to all bank accounts
`count: int`	Number of bank accounts stored in array **`bankAccounts`**
`Bank(String name, String branch)`	Initializes **`name`** and **`branch`** with the parameters `name` and `branch`, creates the array **`bankAccounts`** with size 10, and initializes **`accounts`** to 0.
`Bank(String n, String b, int capacity, String filename)`	Initializes **`name`** and **`branch`** to the values **`n`** and **`b`**, creates the array **`bankAccounts`** with size **`capacity`**, and fills it with the bank account information read from the text file **`filename`** by calling **`readFile(filename)`**.
`readFile(String filename): void`	Reads bank account information from **`filename`** and fills the array **`bankAccounts`** with instances of the classes **`CheckingAccount`**, **`SavingsAccount`**, or **`InvestmentAccount`** according to the type of the account. See footnote[1] for the format of data in each line of **`filename.`**
`saveFile(String filename): void`	Writes the information of the bank accounts from the array **`bankAccounts`** to **`filename`** using the same format of the input file.
`getAccount(int number): BankAccount`	Returns a reference to an object of class **`BankAccount`** whose account number is equal to **`number`**, otherwise returns **`null.`**
`+applyMonthlyFee(): void`	Deducts the value of the static member **`monthlyFee`** from the balance of all the bank accounts
`+viewAllAccounts(): void`	Prints, in tabular format, the information of the bank (name and branch) and all the accounts in the bank (type, number, owner, balance, interest rate/investment type)
`+filterAccounts(String filter): BankAccount[]`	Returns a list of accounts that have the type **`filter`**. **`filter`** may be equal to "Checking", "Savings", or "Investment". The method **`pass()`** is called on all the objects in the array **`bankAccounts`** and a new array is created to copy the accounts for which **`pass()`** returns **`true`**.
`+sortAccounts`	Sorts all the bank accounts in the array **`bankAccounts`** based on their balances using the method **`java.util.Arrays.sort()`**

[1] Each line has 4 or 5 data items:
`AcccountType AccountOwner AccountNumber Balance` (for checking accounts)
`AcccountType AccountOwner AccountNumber Balance InterestRate` (for savings accounts)
`AcccountType AccountOwner AccountNumber Balance InvestmentType` (for investment accounts)

Create a class **BankManager** to test the classes **BankAccount**, **CheckingAccount**, **SavingsAccount**, **InvestmentAccount**, and **Bank**. In the main method, do the following:

1. Initialize the static data member **monthlyFee**, from class **Bank**, to **20.0**.
2. Create an instance of the class **Bank** named **myBank** using the constructor with four parameters and pass the following arguments: "**LEHIGH UNIVERSITY BANK**", "**Bethlehem**", **500**, and **"accounts.txt"**.
3. Display a main menu with 6 operations for the user to select from:
    a. **Manage Existing Account**: Prompt the user to enter an account number that is formed of 6 digits and display an error message if the account number is invalid (not 6 digits) using exception handling (**InvalidAccountNumber**). Find the account in **myBank** list of accounts. If the account number is not found, display a message and prompt the user to enter another operation. If the account number is found, display the following sub menu of operations for the user to select from repeatedly until operation 6 is selected.

        1: **View Balance** – Display the balance of the account.

        2: **Deposit** - Prompt the user for the amount to deposit, call **deposit()** to add the amount to the account balance and display the new balance of the account.

        3: **Withdraw** - Prompt the user for the amount to withdraw and call method **withdraw()**. If the withdrawal is completed successfully, display the new balance, otherwise print an error message.

        4: ***View the monthly interest*** – call the method **applyMonthlyInterest**() and display the return value if the account is a savings account. If the user entered an account that is not of type "Savings", the operation is canceled, and an error message is displayed.

        5: ***View the current profit/loss*** for an investment account - call the method **applyRisk()** and display the return value as a profit if it is positive or as a loss if it is negative. If the user entered an account that is not of type "Investment", the operation is canceled, and an error message is displayed.

6: **Exit Manage Account**: the program returns to the bank main menu.

b. **View All Accounts**: displays the information of all the bank accounts (type, number, owner, balance, and Interest rate or investment type depending on the type of the account).
c. **Apply Monthly Fee:** call method applyMonthlyFee() on myBank.
d. **Sort Accounts:** Sort the list of bank accounts based on their balance using the method **`sort()`** from class **`Bank`**.
e. **Filter Accounts:** Prompt the user to enter the type of account they want to run the filter on (Checking, Savings, or Investment) and display the number of accounts found. Ask the user if he/she wants to see the list of accounts filtered and display the list if the answer is yes.
f. **Exit:** The current information of all the accounts in the array **`bankAccounts`** should be saved back to the file "**`accounts.txt`**" before exiting the program by calling the method **`saveFile()`**.

Test your program for the different operations using the sample runs given below.

Document your code and submit the files:

`InvalidAccountNumber.java,`

`Filter.java`,

`BankAccount.java`,

`CheckingAccount.java`,

`SavingsAccount.java`,

`InvestmentAccount.java`,

`Bank.java`, and

`BankManager.java`

Sample RUN 1

```
Welcome to LEHIGH UNIVERSITY BANK
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
2
Bank name: LEHIGH UNIVERSITY BANK   Bank branch: Bethlehem
Type        Account number    Owner                   Balance     Interest/Investment Type
Investment  149236            James,Butt              53209.73    Growth
Savings     113197            Josephine,Darakjy       8953.61     6.87%
Savings     152889            Art,Venere              88527.96    0.07%
Investment  998961            Lenna,Paprocki          70292.61    Growth
.  .  .
Checking    379661            Alesia,Hixenbaugh       15691.31
Checking    178176            Lai,Harabedian          93381.99
Checking    781741            Brittni,Gillaspie       29384.16
Savings     883228            Raylene,Kampa           3902.02     7.85%
Investment  129855            Flo,Bookamer            8812.16     Shares
Checking    833783            Jani,Biddy              11001.66
Checking    144152            Chauncey,Motley         32364.91
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
1
Enter the account number:
17817
Invalid Account Number. Must be 6 digits.
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
```

```
4: sort accounts
5: filter accounts
6: Exit
1
Enter the account number:
178172
Account not found.
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
1
Enter the account number:
178176
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
1
Balance = $93381.99
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
2
Enter the amount to withdraw:
381.99
Withdrawal completed successfully.
New balance: $93000.00
Select an operation:
1: View Balance
```

```
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
3
Enter the amount to deposit:
7000
Deposit completed successfully.
New balance: $100000.00
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
4
Not A Savings Account. Operation is not applicable.
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
5
Not an Investment Account. Operation is not applicable.
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
6
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
```

```
4: sort accounts
5: filter accounts
6: Exit
2
Bank name: LEHIGH UNIVERSITY BANK   Bank branch: Bethlehem
Type        Account number    Owner                   Balance     Interest/Investment Type
Investment  149236            James,Butt              53209.73    Growth
Savings     113197            Josephine,Darakjy       8953.61     6.87%
Savings     152889            Art,Venere              88527.96    0.07%
Investment  998961            Lenna,Paprocki          70292.61    Growth
.  .  .
Checking    379661            Alesia,Hixenbaugh       15691.31
Checking    178176            Lai,Harabedian          100000.00
Checking    781741            Brittni,Gillaspie       29384.16
Savings     883228            Raylene,Kampa           3902.02     7.85%
Investment  129855            Flo,Bookamer            8812.16     Shares
Checking    833783            Jani,Biddy              11001.66
Checking    144152            Chauncey,Motley         32364.91
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
1
Enter the account number:
883228
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
4
Monthly Interest = $25.53
Select an operation:
1: View Balance
2: Withdraw
```

```
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
1
Balance = $3927.55
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
5
Not an Investment Account. Operation is not applicable.
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
6
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
2
Bank name: LEHIGH UNIVERSITY BANK   Bank branch: Bethlehem
Type        Account number    Owner                   Balance     Interest/Investment Type
Investment  149236            James,Butt              53209.73    Growth
Savings     113197            Josephine,Darakjy       8953.61     6.87%
Savings     152889            Art,Venere              88527.96    0.07%
Investment  998961            Lenna,Paprocki          70292.61    Growth
.  .  .
Checking    379661            Alesia,Hixenbaugh       15691.31
Checking    178176            Lai,Harabedian          100000.00
Checking    781741          Brittni,Gillaspie       29384.16
Savings     883228          Raylene,Kampa           3927.55     7.85%
Investment  129855          Flo,Bookamer            8812.16     Shares
Checking    833783          Jani,Biddy              11001.66
Checking    144152          Chauncey,Motley         32364.91
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
1
Enter the account number:
129855
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
1
Balance = $8812.16
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
5
Profit = $78.50
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
```

```
1
Balance = $8890.66
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
4
Not A Savings Account. Operation is not applicable.
Select an operation:
1: View Balance
2: Withdraw
3: Deposit
4: Monthly Interest (Savings account only)
5: Profit/Loss (Investment account only)
6: Return to Main Menu
6
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
2
Bank name: LEHIGH UNIVERSITY BANK   Bank branch: Bethlehem
Type        Account number    Owner                   Balance     Interest/Investment Type
Investment  149236            James,Butt              53209.73    Growth
Savings     113197            Josephine,Darakjy       8953.61     6.87%
Savings     152889            Art,Venere              88527.96    0.07%
Investment  998961            Lenna,Paprocki          70292.61    Growth
.  .  .
Checking    379661            Alesia,Hixenbaugh       15691.31
Checking    178176            Lai,Harabedian          100000.00
Checking    781741            Brittni,Gillaspie       29384.16
Savings     883228            Raylene,Kampa           3927.55     7.85%
Investment  129855            Flo,Bookamer            8890.66     Shares
Checking    833783            Jani,Biddy              11001.66
Checking    144152          Chauncey,Motley         32364.91
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
3
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
2
Bank name: LEHIGH UNIVERSITY BANK   Bank branch: Bethlehem
Type        Account number    Owner                   Balance     Interest/Investment Type
Investment  149236            James,Butt              53189.73    Growth
Savings     113197            Josephine,Darakjy       8933.61     6.87%
Savings     152889            Art,Venere              88507.96    0.07%
Investment  998961            Lenna,Paprocki          70272.61    Growth
.  .   .
Checking    379661            Alesia,Hixenbaugh       15671.31
Checking    178176            Lai,Harabedian          99980.00
Checking    781741            Brittni,Gillaspie       29364.16
Savings     883228            Raylene,Kampa           3907.55     7.85%
Investment  129855            Flo,Bookamer            8870.66     Shares
Checking    833783            Jani,Biddy              10981.66
Checking    144152            Chauncey,Motley         32344.91
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
6
Thank you for using LEHIGH UNIVERSITY BANK
```

Sample RUN 2 (after saving the file `accounts.txt` and reading it again)

```
Welcome to LEHIGH UNIVERSITY BANK
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
2
Bank name: LEHIGH UNIVERSITY BANK   Bank branch: Bethlehem
Type        Account number    Owner                  Balance     Interest/Investment Type
Investment  149236            James,Butt             53189.73    Growth
Savings     113197            Josephine,Darakjy      8933.61     6.87%
Savings     152889            Art,Venere             88507.96    0.07%
Investment  998961            Lenna,Paprocki         70272.61    Growth
.  .  .
Checking    379661            Alesia,Hixenbaugh      15671.31
Checking    178176            Lai,Harabedian         99980.00
Checking    781741            Brittni,Gillaspie      29364.16
Savings     883228            Raylene,Kampa          3907.55     7.85%
Investment  129855            Flo,Bookamer           8870.66     Shares
Checking    833783            Jani,Biddy             10981.66
Checking    144152            Chauncey,Motley        32344.91
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
4
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
```

```
6: Exit
2
Bank name: LEHIGH UNIVERSITY BANK   Bank branch: Bethlehem
Type        Account number    Owner                   Balance Interest/Investment Type
Savings     723395            Lauran,Burnard          114.96      4.70%
Savings     738286            Willodean,Konopacki     130.02      1.39%
Savings     633427            Detra,Coyier            150.19      8.28%
. . .
Investment  985489            Yvonne,Tjepkema         26451.54    Growth
Investment  712872            Carissa,Batman          26468.10    Property
Investment  973689            Karan,Karpin            27141.08    Growth
Checking    419225            Abel,Maclead            27447.25
.  .  .
Checking    747687            Alaine,Bergesen         98469.62
Savings     434461            Annelle,Tagala          98608.15    8.38%
Checking    671976            Junita,Stoltzman        98702.07
Savings     653544            Candida,Corbley         98970.88    4.48%
. . .
Checking    559319            Thaddeus,Ankeny         99887.29
Checking    886339            Shawnda,Yori            99920.88
Checking    178176            Lai,Harabedian          99980.00
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
5
Enter the type of account to filter:
Checking
168 accounts found.
Do you want to display the list of filtered accounts? (yes/no):
no
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
```

```
6: Exit
5
Enter the type of account to filter:
Savings
176 accounts found.
Do you want to display the list of filtered accounts? (yes/no):
no
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
5
Enter the type of account to filter:
Investment
156 accounts found.
Do you want to display the list of filtered accounts? (yes/no):
yes
Type        Account number    Owner                   Balance     Investment Type
Investment  296422            Sue,Kownacki            236.37      Shares
Investment  242542            Cecily,Hollack          4154.31     Property
Investment  715985            Serina,Zagen            4218.17     Property
. . .
Investment  996287            Minna,Amigon            94410.61    Property
Investment  617236            Lili,Paskin             95315.68    Growth
Investment  635116            Leonida,Gobern          95613.17    Shares
Investment  667244            Ressie,Auffrey          95801.54    Growth
Select an operation:
1: Manage Existing Account
2: List All Accounts
3: Apply Monthly Fee to all accounts
4: sort accounts
5: filter accounts
6: Exit
6
Thank you for using LEHIGH UNIVERSITY BANK
```