# SIT232 – OBJECT ORIENTED DEVELOPMENT

## PROJECT 2 DUE 5:00PM, MAY 20

### SUBMISSION INSTRUCTIONS

You need to zip your entire project and submit the zip file in the provided assignment folder by 5:00pm May 20. Note that this is individual work, and thus you cannot share your solution with others.

### UNIT LEARNING OUTCOMES

UL1. You will be required to demonstrate the correct application of object-oriented concepts including abstraction, encapsulation, inheritance, and polymorphism

UL2. You will be required to apply object-oriented techniques in analysing and modifying an existing C# program

UL3. You will be required to demonstrate the ability to modify an existing application to satisfy new functional requirements

UL4. You will be required to illustrate aspects of the design of the existing application using UML notation.

### PROJECT DESCRIPTION

You are required to refine your assignment 1 using C#. Your tasks include

1. Modifying Customer class to include an ID attribute. This attribute is read-only attribute and can be either loaded from files or generated automatically by your application. IDs are supposed to start with 1 and generated incrementally.
   **Hint:** You would need to have a constructor which receives input as a **StreamReader.**
2. Similarly, modifying Account class so that ID can be either loaded from files or generated automatically by your application. IDs are supposed start with 1 and generated incrementally.
3. Making a class diagram of your application using UML.
4. Applying exception handling for data validation.

5. Loading data for customers from a text file. Customers are stored in a file named “customers.txt” formatted as follows,

   The first row contains the current number of customers.

   Attributes of each customer are then described in the next 7 lines (each attribute per line) and in order: customer ID, first name, last name, address, date of birth (DD/MM/YYYY), contact, and email. Optional attributes (e.g. contact and email) can be blank if not provided. For example, customers used in Assignment 1 can be stored in a file as follows (note that empty lines correspond to blank, e.g. contact = “”)

   ```
   3
   1
   Arley
   Praise
   12 Hay Rd
   2/10/1990
   0412232116
   arleyp@gmail.com
   2
   Joseph
   Abot
   4/1 Mandy Pl
   11/5/1970
   0413221624

   3
   Rose
   Magaret
   30 Buxton St
   6/7/1980

   rmt@yahoo.com
   ```

6. Loading accounts from a text file named “accounts.txt” and formatted as follows,

   The first row contains the current number of accounts.

   Information of each account is then described in the next 4 lines (each attribute per line) and in order: account type (1 or 2 corresponding to Type1 or Type2 account), account ID, owner ID (i.e. customer ID), opened date (DD/MM/YYYY), closed date (DD/MM/YYYY or blank if the account is still active), and balance. For example, suppose that the current date is 01/05/2018, accounts used in Assignment 1 (prior to applying any transactions, e.g. transfer, withdraw, etc.) can be stored in a file as follows. Note that empty lines correspond to blank, e.g. contact = “”.

```
5
1
1
1
1/2/2018

100
2
2
1
15/2/2018

5000
1
3
2
20/3/2018

0
2
4
3
4/3/2018

2000
2
5
3
1/5/2018

3000
```

7. You are to develop a command line-based user interface (CUI) in Console window.

a. **Main menu**: This is the starting interface of your application and should include

```
Enter 1 if you want to create a new customer
Enter 2 if you want to search a customer
Enter 3 if you want to open an account
Enter 4 if you want to search an account
Enter 5 if you want to transfer money
Enter 6 if you want to deposit
Enter 7 if you want to withdraw
Enter 8 if you want to set monthly deposit
Enter any other keys if you want to exit
Your option:
```

b. **Menu 1**: This is the interface for creating a new customer (i.e. when users enter "1" from **Main menu**). This interface requires users to enter the information of a new customer (required information is prompted one by one).

```
Enter First Name:
Enter Last Name:
Enter Address:
Enter Date of birth (DOB):
Enter Contact:
Enter Email:
Press "s" or "S" if you want to submit
Press any other keys if you want to cancel
```

If users enter "s" or "S", the supplied date of birth and contact will be validated (see Assignment 1). You also need to make sure that there can be NO two or more customers having the same first name and last name.

If all the entered information is valid, then a new customer is created and **Main menu** will be displayed. Otherwise, appropriate error messages are presented and **Menu 1** is recalled to allow users re-enter information.

In **Menu 1**, if users enter any letters other than "s" or "S", then no customer will be created and **Main menu** will be displayed.

c. **Menu 2**: This is the interface for searching a customer (i.e. when users enter "2" from **Main menu**). This interface requires users to enter the information of a customer (required information is prompted one by one)

```
Enter First Name:
Enter Last Name:
Enter Address:
Enter Date of birth (DOB):
Enter Contact:
Enter Email:
Press "s" or "S" if you want to submit
Press any other keys if you want to cancel
```

If users enter "s" or "S", your application needs to search ALL customers who match ONE of the searching criteria provided by users and then display all matching customers in **Menu 2.1** as illustrated below.

```
Search results:
ID: 1 Name: Arley Praise Address: 12 Hay Rd DOB: 2/10/1990 Contact:
0412232116 Email: arley@gmail.com Total balance: 6,614.7

Press any keys to go back to Main menu
```

In **Menu 2.1**, if users enter any keys, then **Main menu** is presented.

**Hint:** You should modify ToString() method in Customer class to include customer ID and use ToString() to display search results.

In **Menu 2**, if users enter any letters other than “s” or “S”, then no customer will be displayed and **Main menu** will be turned back.

d. **Menu 3**: This is the interface for opening a new account (i.e. when users enter “3” from **Main menu**). This interface requires users to enter the information of a new account (required information is prompted one by one)

```
Enter Owner ID:
Enter Account Type (1 or 2):
Enter Initial Balance:
Press “s” or “S” if you want to submit
Press any other keys if you want to cancel
```

If users enter “s” or “S”, the supplied data will be validated, e.g. supplied owner ID must exist, input balance must be greater than 0, entered account type must be either 1 or 2, etc.

If all the entered information is valid, then a new account is created and **Main menu** will be displayed. Otherwise, appropriate error messages are presented and **Menu 3** is recalled to allow users re-enter information.

e. **Menu 4**: This is the interface for searching an account (i.e. when users enter “4” from **Main menu**). This interface requires users to enter the information of an account (required information is prompted one by one)

```
Enter Account ID:
Press “s” or “S” if you want to submit
Press any other keys if you want to cancel
```

If users enter “s” or “S”, your application will search for an account whose ID is given by users, and then display the information of the matching account in **Menu 4.1** as illustrated below.

```
Search results:
ID: 1 Opened Date: 1/2/2018 100 Arley Praise

Press any keys to go back to Main menu
```

In **Menu 4.1**, if users enter any keys, then **Main menu** is presented.

f. **Menu 5**: This is the interface for transferring money (i.e. when users enter “5” from **Main menu**). This interface requires users to enter the information of a source and destination account and an amount of money (required information is prompted one by one).

```
Enter Source Account ID:
Enter Destination Account ID:
Enter Amount:
Press “s” or “S” if you want to submit
Press any other keys if you want to cancel
```

If users enter “s” or “S”, the supplied data will be validated, e.g. supplied account IDs must exist, account types must be valid for this transaction, transferred amount cannot exceed the balance of the source account, etc.

If all the entered information is valid, then the transferring transaction will be performed and **Main menu** will be displayed. Otherwise, appropriate error messages are presented and **Menu 5** is recalled to allow users re-enter information.

In **Menu 5,** if users enter any letters other than “s” or “S”, then no transaction will be performed and **Main menu** will be displayed.

g. **Menu 6**: This is the interface used for deposit (i.e. when users enter “6” from **Main menu**). This interface requires users to enter the information of an account on which the deposit is made and an amount of money (required information is prompted one by one).

```
Enter Account ID:
Enter Amount:
Press “s” or “S” if you want to submit
Press any other keys if you want to cancel
```

If users enter “s” or “S”, the supplied data will be validated, e.g. supplied account IDs must exist, account type must be valid for this transaction, amount must be greater than 0, etc.

If all the entered information is valid, then the deposit will be performed and **Main menu** will be displayed. Otherwise, appropriate error messages are presented and **Menu 6** is recalled to allow users re-enter information.

In **Menu 6,** if users enter any letters other than “s” or “S”, then no transaction will be performed and **Main menu** will be displayed.

h. **Menu 7**: This is the interface used for withdrawal (i.e. when users enter “7” from **Main menu**). This interface requires users to enter the information of an account on which the withdrawal is made and an amount of money (required information is prompted one by one).

```
Enter Account ID:
Enter Amount:
Press “s” or “S” if you want to submit
Press any other keys if you want to cancel
```

If users enter “s” or “S”, the supplied data will be validated, e.g. supplied account IDs must exist, account type must be valid for this transaction, withdrawn amount must be valid, etc.

If all the entered information is valid, then the withdrawal will be performed and **Main menu** will be displayed. Otherwise, appropriate error messages are presented and **Menu 7** is recalled to allow users re-enter information.

In **Menu 7,** if users enter any letters other than “s” or “S”, then no transaction will be performed and **Main menu** will be displayed.

i. **Menu 8**: This is the interface used for setting monthly deposit (i.e. when users enter “8” from **Main menu**). This interface requires users to enter the information of an account and a monthly deposit (required information is prompted one by one).

```
Enter Account ID:
Enter Monthly Deposit:
Press “s” or “S” if you want to submit
Press any other keys if you want to cancel
```

If users enter “s” or “S”, the supplied data will be validated, e.g. supplied account IDs must exist, account type must be valid for this transaction, monthly deposit must be greater than 0, etc.

If all the entered information is valid, then the monthly deposit will be set and **Main menu** will be displayed. Otherwise, appropriate error messages are presented and **Menu 8** is recalled to allow users re-enter information.

In **Menu 8,** if users enter any letters other than “s” or “S”, then no transaction will be performed and **Main menu** will be displayed.

j. In **Main menu,** if users enter any letters other than “1”, “2”, “3”, “4”, “5”, “6”, “7”, and “8”, then customers and accounts will be saved back to “customers.txt” and “accounts.txt” (with the same format) and the application ends.

## OTHERS

1) Your implementation should reflect object-oriented principles such as abstraction, encapsulation, polymorphism.
2) Make sure that you have sufficient comments to enhance the readability of your code, organise your program reasonably and make it maintainable conveniently (e.g. avoiding hard-code and repeating code, naming variables and methods meaningfully, using static, constants reasonably).
3) Make your outputs in a reasonable and tidy format.
4) If your program cannot be compiled, the maximum score you could get is 50% of the total mark (i.e. 7.5%).

**END**