



**ENGR 101**  
**Introduction to Programming**  
*Spring 2019*

**Mini Project II**

**SEHIR BANK**



**+ Bonus Part**

**Due Date:** Monday, April 29<sup>th</sup>, 2019, 23:59

In mini project 2, you are going to develop the second version of the text-based simple banking application you did in mini project 1. The primary goal of this project is to make you practice dictionaries, lists, tuples, recursion, functions, loops, conditional execution, and getting user input through keyboard. The detailed explanations about the application that you are going to develop are provided below.

### **Project Logic:**

The application is based on the idea that customers should be able to interact with their bank accounts in the easiest way possible. The main features you will develop in this mini project are as follows:

- withdraw and deposit money
- transfer money to other accounts
- display the users' account information history or report the actions of the user in the bank.

In addition, Admin and Admin's options to maintain users accounts is the bonus part of the project.

### **Data Structures that MUST be used:**

What is meant by a data structure?

Data structure is any allowable combination of lists, tuples, and dictionaries containing integers, strings or floats.

E.g. `[ ['abc', '0000'], [1, 2, 3] ]`;

`{ '2': { (2, 2): [1, 1, 1], (2, 3): ['s', 'd'] }, { (1, 1): [2, 2, 2] } }`;  
`{ [1, 1, 1]: "abc" }`.

You should have three data structures separate from each other for this project:

- **Data Structure 1:** You need to have a data structure that will store the admin credentials namely admin name and admin password (for bonus part only).
- **Data Structure 2:** You need to have a data structure that will store the user credentials namely username and user password.

- **Data Structure 3:** You need to have a data structure that that will hold all the user activities => (user, password - balance, deposit, withdrawals, transfers etc.).

*PLEASE NOTE THAT YOU SHOULD MAINTAIN DATA STRUCTURE 3 SUCH THAT ALL THE CHANGES MADE ARE SORTED BY TIME, YOU CAN NOT USE ANY BUILT-IN SORTING FUNCTION OR ANYTHING OF THAT SORT!!!*

Each user needs to have an instance in the data structure that will have a:

- balance which should contain – current balance amount
- user deposits which should contain - amount, timestamp
- user withdrawals which should contain – amount, timestamp
- user transfers which should contain – amount, to who (username), timestamp

Time Stamp should contain date - time (Istanbul) as shown:

```
21/03/2019 19:57:53
```

!!! AFTER EACH CORRESPONDING ACTION, THE BALANCE(S) SHOULD BE UPDATED ACCORDINGLY AND THE INFORMATION OF EACH ACTION WILL BE APPENDED TO A STRUCTURE THAT WILL KEEP A RECORD OF WHAT WAS DONE !!!

!!! ALL THE CORRESPONDING DEPOSITS, WITHDRAWALS AND TRANSFERS SHOULD BE ORDERED BY TIME AS SHOWN IN **Figure 10** !!!

Note: For Data Structure 3, the exact number and type of data structure are not specified. What matters is that the functionality and data storage capacity should be met within **one** encompassing **structure**.

### **Text-Based Bank App version 2:**

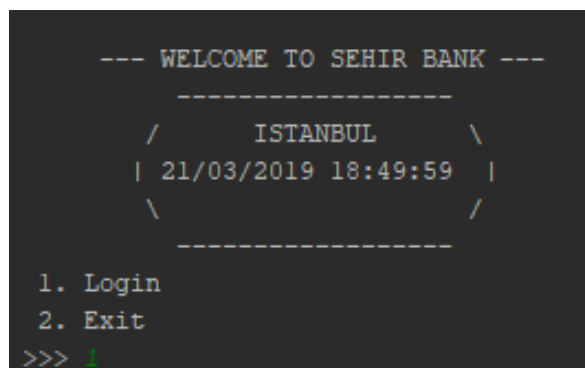
Your bank **MUST** have 3 existing customers by default (when you run your program they should already have been added into your data structure):

- First user's username should be Ahmed and his password should be **1234**
- Second user's username should be Zeynep and her password should be **4321**
- Third user's username should be Alberto and his password should be **4422**

These three user names and their corresponding passwords should be placed in a proper data structure.

When your program first starts, welcome message and time in Istanbul will be displayed as shown below in **Figure 1**. Then, users will be asked the service they want. They will have two options: either log in or exit the application.

Valid inputs are 1 or 2. **Figure 1** below is a sample of how your welcome page should look like:



```

--- WELCOME TO SEHIR BANK ---
-----
/      ISTANBUL      \
| 21/03/2019 18:49:59 |
\                      /
-----
1. Login
2. Exit
>>> |

```

**Figure 1.** Welcome message.

If the user chooses login, your application should ask the user to enter his/her user name and password. If the user enters a wrong input, your application should warn the user and ask for a valid username/password again, until the user provides the correct one. Next, the user will be asked three options as shown in **Figure 2**. The details of the **Admin** will be explained in the bonus part. If **Go Back** is entered, the login/exit page should show up.

Your program should keep asking until the user gives the correct credentials. Once your application gets the correct credentials, your program should again **welcome** the user but this time with his/her **name clearly displayed** (**Figure 2**). Your application should also display the list of the available services (1,2,3,4,5 in **Figure 2**).

```
What do you want to login as:
  1. Admin
  2. User
  3. Go Back
>>> 2
User Name: Ahmed
Password: 1234

Ahmed Welcome to Sehir Bank

Please enter the number of the service
  1. Withdraw Money
  2. Deposit Money
  3. Transfer Money
  4. My Account Information
  5. Logout
>>> |
```

**Figure 2.** Login screen and available services after a proper login.

Wrong user inputs and **Go Back** should be handled properly (**Figure 3**).

```
What do you want to login as:
  1. Admin
  2. User
  3. Go Back
>>> 4
Please enter a number that is a valid input
```

**Figure 3.** Wrong input selected by the user.

The user should have 4 services to choose from and a logout option (**Figure 2**). Your program should print the service numbered and the users should be able to enter the number corresponding to the service they want to use.

If **option 1** (**Figure 2**) is picked, your program should ask the user to provide an amount of money to **withdraw** from their account as shown in **Figure 4** below. Your program should check whether or not the user's account has **enough money** to do that transaction. If not, your program should warn the user that he/she doesn't have enough money and go back to the main menu. If the user has enough money to do the transaction required, then your program should inform the user that the amount he/she requested has been withdrawn from his/her account.

```
>>> 1
Please enter the amount you want to withdraw: 100
100 TL withdrawn from your account
Going back to main menu...
```

**Figure 4.** The user been asked the amount to withdraw.

The change in amount of money after withdrawal should be reflected on the balance and recorded in data structure 3. A time stamp and the amount should also be appended to the respected **withdrawals** substructure which is a part of the data structure 3.

If **option 2** (**Figure 2**) is picked, your program should ask the user to provide an amount of money to **deposit** to their account (**Figure 5**). After the deposit is successful, your program should inform the user that the amount he/she entered has been deposited into his/her account. This change should again be reflected on the balance and be recorded in data structure 3. A time stamp and the amount should also be appended to the respected deposits structure which is a part of the data structure 3.

Your program should go back to the main menu after the deposit is complete.

```
>>> 2
Please enter the amount you want to deposit: 1000
1000 TL added to your account
Going back to main menu...
```

**Figure 5.** The user been asked the amount to deposit.

If **option3** (**Figure 2**) is picked, your program should ask the user to enter a name of the user to whom the money will be **transferred** to (**Figure 6**).

If the user enters his/her own name, the program should print that the user does not exist (**Figure 6**). Or, if a name of a non-existing user is entered, the program should print that the user does not exist and give another chance to the user to enter another username.

```
>>> 3
Warning: If you want to abort the transfer please enter abort
Please enter the name of the user you want transfer money to: Ahmed
Transferring to user with the name Ahmed is not possible!
User does not exist!

Warning: If you want to abort the transfer please enter abort
Please enter the name of the user you want transfer money to: Zeynep
```

**Figure 6.** Wrong username given for transfer.

If the user wants to cancel the transfer process, you should provide the functionality to stop the service: if the input **abort** is entered, the transfer service should stop and go back to the main menu (**Figure 7**).

```
Warning: If you want to abort the transfer please enter abort
Please enter the name of the user you want transfer money to: abort
Going back to main menu...
```

**Figure 7.** Aborting transfer process.

If the user exists, then you should give the chance to enter the money for transfer. If the user has enough money in their account for the amount of transfer, the transfer should be allowed (**Figure 8**). Otherwise, your program should not allow the user to transfer money and print that the user does not have enough money and give the choice to transfer again or go back to services menu (**Figure 9**).

```

>>> 3
Warning: If you want to abort the transfer please enter abort
Please enter the name of the user you want transfer money to: Seynep
Please enter the amount you want to transfer: 100
Money transferred successfully!
Going back to main menu...

```

**Figure 8.** Transferring money successful when the user has enough money in his/her balance.

```

>>> 3
Warning: If you want to abort the transfer please enter abort
Please enter the name of the user you want transfer money to: Seynep
Please enter the amount you want to transfer: 100000
Sorry!, you don't have the entered amount

1.Go back to main menu
2.Transfer again
|>>>

```

**Figure 9.** Transferring money unsuccessful when the user does not have enough money in his/her balance.

If the transfer is possible and successful you should reflect this to the balance of BOTH users and update the necessary information (amount, to/from who (username), timestamp) in BOTH users' transfer structures as part of data structure 3.

If **option 4 (Figure 2)** is picked, **My Account Information**, your application should display the name of the bank, the current date and time. The USER NAME, PASSWORD, BALANCE AND ALL THE TRANSACTIONS, DEPOSITS, WITHDRAWALS BEING ORDERED BY TIME should be displayed as shown in **Figure 10**.

PLEASE NOTE THAT YOU SHOULD MAINTAIN THE DATA STRUCTURES SO THAT ALL THE CHANGES MADE ARE SORTED BY TIME, YOU CAN NOT USE ANY SORTING FUNCTION OR ANYTHING OF THAT SORT!!!

When you log out from the system and then log in with **another user account** and go to **My Account Information**, the right information for the right user should be displayed.



```

>>> 
----- SEHIR Bank -----
----25/03/2019 12:20:58----
-----
Your Name: Ahmed
Your Password: 1234
Your Balance Amount (TL): 1250
-----
User Activities Report:

Your Withdrawals:
  25/03/2019 12:19:29 100 TL
  25/03/2019 12:20:35 500 TL

Your Deposits:
  25/03/2019 12:18:59 1000 TL
  25/03/2019 12:20:25 1500 TL

Your Transfers:
  Time Person Amount:
  25/03/2019 12:19:22 Transferred to Zeynep 200 TL
  25/03/2019 12:19:55 Transferred to Alberto 300 TL
  25/03/2019 12:20:55 Transferred to Zeynep 150 TL

-----
Going back to main menu...

>>> 
----- SEHIR Bank -----
----25/03/2019 12:21:57----
-----
Your Name: Zeynep
Your Password: 4321
Your Balance Amount (TL): 350
-----
User Activities Report:

Your Withdrawals:

Your Deposits:

Your Transfers:
  Time Person Amount:
  25/03/2019 12:19:22 Transferred to me from Ahmed 200 TL
  25/03/2019 12:20:55 Transferred to me from Ahmed 150 TL

-----
Going back to main menu...

>>> 
----- SEHIR Bank -----
----25/03/2019 12:23:25----
-----
Your Name: Zeynep
Your Password: 4321
Your Balance Amount (TL): 501
-----
User Activities Report:

Your Withdrawals:
  25/03/2019 12:23:22 199 TL

Your Deposits:
  25/03/2019 12:22:36 1000 TL

Your Transfers:
  Time Person Amount:
  25/03/2019 12:19:22 Transferred to me from Ahmed 200 TL
  25/03/2019 12:20:55 Transferred to me from Ahmed 150 TL
  25/03/2019 12:22:48 Transferred to Ahmed 400 TL
  25/03/2019 12:23:00 Transferred to Alberto 250 TL

-----
Going back to main menu...

>>> 
----- SEHIR Bank -----
----25/03/2019 12:24:10----
-----
Your Name: Ahmed
Your Password: 1234
Your Balance Amount (TL): 1650
-----
User Activities Report:

Your Withdrawals:
  25/03/2019 12:19:29 100 TL
  25/03/2019 12:20:35 500 TL

Your Deposits:
  25/03/2019 12:18:59 1000 TL
  25/03/2019 12:20:25 1500 TL

Your Transfers:
  Time Person Amount:
  25/03/2019 12:19:22 Transferred to Zeynep 200 TL
  25/03/2019 12:19:55 Transferred to Alberto 300 TL
  25/03/2019 12:20:55 Transferred to Zeynep 150 TL
  25/03/2019 12:22:48 Transferred to me from Zeynep 400 TL

-----
Going back to main menu...

>>> 
----- SEHIR Bank -----
----25/03/2019 12:26:28----
-----
Your Name: Alberto
Your Password: 4422
Your Balance Amount (TL): 550
-----
User Activities Report:

Your Withdrawals:

Your Deposits:

Your Transfers:
  Time Person Amount:
  25/03/2019 12:19:55 Transferred to me from Ahmed 300 TL
  25/03/2019 12:23:00 Transferred to me from Zeynep 250 TL

-----
Going back to main menu...

```

**Figure 10.** My Account Information for different users: Ahmed, Zeynep and Alberto at different date-times.

Each user should be able to transfer to all the existing users in the bank!!!

If **option 5 (Figure 2)** is picked, your program should **logout** but keep the users' account balance and all the information in the data structure such that if the user logs in again, his/her account balance is correct and reflects the most recent update. For example, if the user with 0 balance logs in, deposits 60 TL and then logs out; when he/she logs in again and checks his/her account information, the balance should display 60 TL (providing that he did not withdraw or deposited any other money.)

### **BONUS PART**

You will have an Admin data structure containing one admin with name Ibrahim and password 1122, which have to be in the data structure prior to running your program. The functionality you will provide for Admin is that the Admin can add, remove or display users of the bank (**Figure 11**).

```
What do you want to login as:
 1. Admin
 2. User
 3. Go Back
>>> 1
Admin Name: Ibrahim
Password: 1122
Welcome Ibrahim
  - - - Admin Menu - - -
Please enter a number of the settings operations supported:
 1. Add User
 2. Remove User
 3. Display all Users
 4. Exit Admin Menu
|>>>
```

**Figure 11.** Options for **Admin** log-in.

If the Admin chooses option 1 to **Add User** (*Figure 12*); a new username and password can be entered. After adding user(s), both data structures 2 and 3 should be updated accordingly.

```
>>> 1
Access Granted
Enter the new user name:
>>> Andre
Enter the new user password
>>> 1313
Andre was added as an user
- - - Admin Menu - - -
Please enter a number of the settings options:
1. Add User
2. Remove User
```

*Figure 12. Add User* option for Admin.

If Admin chooses option 3 to **Display All Users**, all users of Sehir Bank with their username and password should be displayed as shown in *Figure 13*.

```
>>> 3
Access Granted
There are 4 users using Sehir Bank:
Name & Password:
1. Andre 1313
2. Ahmed 1234
3. Zeynep 4321
4. Alberto 4422
-----
- - - Admin Menu - - -
Please enter a number of the settings options:
1. Add User
```

*Figure 13. Display All Users* option for Admin.

If the Admin tries to **Remove User** (option 2 in *Figure 11*) that does not exist, the program should handle this error as shown in *Figure 14*.

```

4. Exit Admin Menu
>>> 2
Access Granted
Enter the user name:
>>> asd
asd does not exist as an user to Sehir Bank
- - - Admin Menu - - -
Please enter a number of the settings operations supported:
1. Add User
2. Remove User
3. Display all Users

```

**Figure 14.** Admin trying to **Remove User** that does not exist.

If the **Remove User** is successful, your program should print that the user is removed (**Figure 15**).

```

>>> 2
Access Granted
Enter the user name:
>>> Ahmed
Ahmed was removed as an user to Sehir Bank
- - - Admin Menu - - -
Please enter a number of the settings operations supported:
1. Add User
2. Remove User

```

**Figure 15.** Admin **Removing User** successfully.

When the users are removed successfully and Admin **displays all users** again, the removed users should not be displayed (**Figure 16**). The user credentials in data structure 2 and the user information in data structure 3 should be deleted as well!!!

```

>>> 3
Access Granted
There are 2 users using Sehir Bank:
Name & Password:
1. Andre 1313
2. Zeynep 4321

```

**Figure 16.** Admin **displaying all users** after removed some users successfully.

After removing users, if the admin exits (**Figure 17**) and a user then tries to log in with the username that was removed by the Admin, they will not be able to log in because the user credentials and user information was deleted (**Figure 18**)!!!

```

4. Exit Admin Menu
>>> 4

--- WELCOME TO SEHIR BANK ---
-----
/      ISTANBUL      \
| 24/03/2019 11:55:13 |
\                    /
-----

1. Login
2. Exit
|>>>

```

**Figure 17.** Admin exiting Admin Menu.

```

What do you want to login as:
1. Admin
2. User
3. Go Back
>>> 2
User Name: Ahmed
Password: 1234

Wrong User Name or Password. Please Try Again!

```

**Figure 18.** A user trying to enter with a username that was removed by the Admin.

A user cannot login with a removed username. However, if the user enters with an existing username like Andre that was newly added by the admin as shown in **Figure 12**, he/she could use all the services just like any user, and see account Information as shown in **Figure 19** reflecting all the actions that was carried out in Andre's account.

```

>>> 4

----- SEHIR Bank -----
-----2019-03-24 08:59:33-----
-----
Your Name: Andre
Your Password: 1313
Your Balance Amount (TL): 1050
-----
User Activities Report:

Your Withdrawals:

Your Deposits:
  24/03/2019 11:59:00 1000 TL
  24/03/2019 11:59:08 100 TL

Your Transfers:
  Time Person Amount:
  24/03/2019 11:59:31 Zeynep 50 TL

-----
Going back to main menu...

```

**Figure 19.** Account information of an existing user Andre.

### Implementation Notes:

- You may check this link to get further information about the time stamps with the time module. (<https://docs.python.org/3/library/time.html>)
- You **MUST** use functions. You are strongly advised to use *a function for each service*. However, you have to use **at least 3 functions**. (10% reduction if you do not create 3 functions)
- YOU **MUST** USE 3 DATA STRUCTURES (including the one for bonus part), NO MORE AND NO LESS.
- Your code should be efficient, easy to follow, and track.

### Warnings:

- If you don't follow the implementation notes, you are very likely to lose points.
- **Do not** talk to your classmates on project topics when you are implementing your projects. **Do not** show or email your code to others. If you need help, talk to your TA's or the instructor, not to your classmates. If somebody asks you for help, explain them the lecture slides, but do not explain any project related topic or solution. Any similarity in your source codes will have **serious consequences** for both parties.
- Carefully read the project document, and pay special attention to sentences that involve **"should", "must", "should not", "do not"**, and other underlined/bold font statements.
- If you use code from a resource (web site, book, etc.), make sure that you reference those resource at the top of your source code file in the form of comments. You should give details of which part of your code is from what resource. Failing to do so **may result in** plagiarism investigation.
- Even if you work as a group of two students, each member of the team should know every line of the code well. Hence, it is **important** to understand all the details in your submitted code.

### How and when do I submit my project? :

- Projects may be done individually or as a small group of two students (doing it individually is recommended for best learning experience). If you are doing it as a group,

only **one** of the members should submit the project. File name will tell us group members (Please see the next item for details).

- Submit your own code in a **single** Python file. Name your code file with your and your partner's first and last names (see below for naming):
  - ❖ If your team members are Deniz Barış and Ahmet Çalışkan, then name your code file as `deniz_baris_ahmet_caliskan.py` (**Do not** use any Turkish characters in file name). If you are doing the project alone, then name it with your name and last name similar to the above naming scheme. Those who **do not** follow the above naming conventions will get **5 points off** of their grade.
- Submit it online on LMS by **Monday, April 29<sup>th</sup>, 2019, 23:59**.

**Late Submission Policy:**

- 10%: Submissions which are 1 hour late
- 20%: Submissions which are 2-23 hour late
- 30%: Submissions which are 24 hours late.
- 50%: Submissions which are 48 hours late.
- Submission more than 48 hours late will not be accepted

**Grading Criteria:**

Meaningful variable names and following the text interface as given in this document <b>(4 pts)</b> and sufficient commenting <b>(2 pts)</b>	Proper use of functions, compact code with no unnecessary repetitions	Proper use of data structures as explained in this document	Login Function	Depositing <b>(10 pts)</b> and Withdrawing <b>(10 pts)</b> and Transferring <b>(15 pts)</b>	Display My Information	Logout	Bonus Part
<b>(6 pts)</b>	<b>(4 pts)</b>	<b>(30 pts)</b>	<b>(10 pts)</b>	<b>(35 pts)</b>	<b>(10 pts)</b>	<b>(5 pts)</b>	<b>(20 pts)</b>

**Have further questions?**

Please contact your TAs if you have further questions. If you need help with anything, please use the office hours of your TAs and the instructor to get help. Do not walk in randomly (especially on the last day) into your TAs' or the instructor's offices. Make an appointment first. This is important. Your TAs have other responsibilities. Please respect their personal schedules!

***Good Luck!!***