1. Program Source Code with Explanation

The first part of the design is the admin operation page. The function of create_newaccount() is created for the administrator to create a new profile for the new customer. The heading of the operation page will be formed with print(). Both text files used to store the customer's login username and password as well as the customer's personal details will be produced to append new data of the customer with open(,'a'). Later on, the number of customer's record that the administrator would want to keep will be accepted as integer using int(input()). Subsequently, for every number of record that the administrator has entered will be looped within the range of the number of records keyed in using 'for x in range(data)'. Within the loop, login username, password, name, identification number of the new customer will be accepted using input() and be written into the text files created with write(). Both text files will be closed with close(), the loop will be ended and the function will end with a notification using print(). Every functions in this part except create_newaccount() is built in functions.

```
##Admin Page##

#Create new account for customer
def create_newaccount():
    print('\n\t*****Create New Account for Customer*****\n\t')
    customer_login=open('customerlogin.txt','a')
    customer_details=open('customerdetails.txt','a')
    data = int(input("\n\tPlease key in the number of customer's record that you want to keep: "))
    for x in range(data):
        loginid = input("Please create customer's login ID: ")
        customer_password = input("Please create customer's password: ")
        customer_name = str(input("Please enter customer's name: "))
        customer_ID = input("Please enter customer's identification no.: ")
        customer_login.write(loginid+"\t"+customer_password+"\n")
        customer_details.write(loginid+"\t\t"+customer_name+"\t\t"
                               +customer_ID+"\n")
    customer_login.close()
    customer_details.close()
    print('\n\t***Account successfully created***\n\t')
```

This part of the design is to allow the administrator to view all the customer accounts or profiles. The function of view_allprofiles() is created. A heading specifying the function of the operation will be printed using print(). The text file that stores the customer details will be opened for reading with open(,'r') and read along using read() with its headings for each columns with print(). Then, the file will be closed once the function is ended with close(). Every functions in this part except view_allprofiles() is built in functions.

```
#View all customer accounts
def view_allprofiles():
    print("\n\t*****All Customers' Profiles*****\n\t")
    myfile = open("customerdetails.txt","r")
    print("\n\nCustomer ID:\tCustomer Name:\tCustomer Identification No:")
    print(myfile.read())
    myfile.close()
```

In this part, the function of search_customer() is defined to search the profile of a specific customer. A print() is used to print the heading that specifies the function of this operation. Later on, any customer detail will be accepted with input(). When the text file that stores customer's details is opened for reading with open(,'r'), the name of each columns will be

printed. For every line in the text file (i.e. for line in myfile), any unnecessary space or character will be removed with rstrip(). Later on, if the customer detail accepted is within the line in the text file (i.e. if detail in line), the details of the customer will be printed (i.e. print(line)). Then, the text file will be closed with the close() once the defined function is ended. Every functions in this part except search_customer() is built in functions.

```
#Search for specific customer account
def search_customer():
    print("\n\t*****Search for Specific Customer's Profile*****\n\t")
    detail=input("Enter any customer detail:")
    myfile=open("customerdetails.txt","r")
    print("\n\nCustomer ID:\tCustomer Name:\tCustomer Identification No:")
    for line in myfile:
        line=line.rstrip()
        if detail in line:
            print(line)
    myfile.close()
```

This part of the design permits the administrator to search transaction of a specific customer. The function of search_transaction() is defined. Later on, the heading showing the purpose of the function will be printed with print(). The customer login username will be then accepted with input(). Subsequently, the text file that stores customers' transaction will be opened for reading using open(,'r') along with the name of each columns using print(). For every line in the text file (i.e. for line in myfile), any unnecessary space and character will be removed with rstrip(). If the customer login username is within the line (i.e. if detail in line), the details of the customer's transaction will be printed with print(). The text file will be closed once the defined function is ended. Every functions in this part except search_transaction() is built in functions.

```
def search_transaction():
    print("\n\t*****View Transaction*****\n\t")
    detail=input("Enter the customer ID:")
    myfile=open("customeraccount.txt","r")
    print("\n\nCustomer ID:\tCustomer Name:\tDeposit Amount:\tWithdrawal Amount:\tBalance:
")
    for line in myfile:
        line=line.rstrip()
        if detail in line:
            print(line)
    myfile.close()
```

This part covers the operation for customers. The first function created is check_auth(x,y) in order to screen through whether the username and password entered by the customer is aligned with the details created in the text file by the administrator. The x represents the username whereas the y represents the password. The text that stores the customer's username and password will be opened for reading. Subsequently, an operation that shows the customer's username (x) and password (y) with its space in between (i.e. x+'\t'+y) is formed. Another pre-defined operation is created to assume the authentication is false. Later on, for every line in the text file, any unnecessary character or space will be removed using rstrip(). If the username and password entered is alighed with the details in the text file, the authentication will be considered as true. The text file will then be closed with close() and return the pre-

defined authentification which assumed to be false. Every functions in this part except check_auth(x,y) is built in functions.

```
###Customer Page###

#Check customer authentication

def check_auth(x,y):
    customer_login=open("customerlogin.txt","r")
    auth = x+"\t"+y
    condition = False
    for line in customer_login:
        line = line.rstrip()
        if auth == line:
            condition = True
    customer_login.close()
    return condition
```

In this part, the function of deposit_withdrawal() is created for the customer to deposit and withdraw money. A symbolic constant is created with the output of True to assume the transaction system can be validated. Later on, the heading stating the purpose of the function will be printed. The text file that stores the customers' transaction will be opened for appending with open(,'a'). Three symbolic constants (i.e., customer_balance, customer_deposit and customer_withdraw) are assigned to represent the numbers of transaction. The customer's login id and name will then be accepted with input(). Subsequently, a list of transactions done by the customer will be printed.

While the transaction_system is validated as True (i.e. while (transaction_system == True)), the option operation will be printed with print(). The customer's option code will then be accepted with input(). If the customer selects option 'A', the customer will be directed to the deposit page. The deposit amount inserted by the customer will then be accepted with input(). The deposit amount accepted will be converted to float data type (i.e. float()) and symbolized as customer_deposit. The customer_withdraw will symbolize with 0 (i.e. customer_withdraw = 0) as no withdraw amount has been inserted. The total amount of the customer's transaction will be symbolized as customer_balance by adding the deposit amount accepted. The transaction made by the customer will be printed with the names of each column using print() and be written into the text file for record with write().

If the customer selects option 'B', the customer will be directed to the withdrawal page. The withdrawal amount inserted by the customer will then be accepted with input(). If the withdrawal amount accepted is less than or equal to the balance_amount, the withdrawal amount will be converted to float data type (i.e. float()) and symbolized as customer_withdraw. The customer_deposit will symbolize with 0 as no deposit amount has been inserted. The total amount of the customer's transaction will then be symbolized as customer_balance by substracting the withdrawal amount accepted. The transaction made by the customer will be printed using print() with the names of each column and be written into the text file for record with write(). However, if the withdrawal amount accepted is more than the balance _amount, the transaction will be rejected with a notification using print(). Lastly, if the customer chooses option 'C', the text file will be closed with close() and the system will be ended. If the customer enters the wrong option code, a notification to try again will be printed with print().

```
#function for customer's deposit and withdrawal
transaction_system = True
def deposit_withdrawal():
    print("\n\t*****Deposit or Withdraw Money*****\n\t")
    customer_account=open("customeraccount.txt","a")
    customer_balance = 0
    customer_deposit = 0
    customer_withdraw = 0
    loginid = input("Please enter your login ID: ")
    customer_name=input("Please enter your name: ")
    print("Customer ID:\tCustomer Name:\tBalance:")
    print(loginid+"\t\t"+customer_name+"\t\t"+str(customer_balance))
    while (transaction_system == True):
        print('\n\tPlease select the below option:\n\t')
        print('\n\t[A] Deposit money\n\t')
        print('\n\t[B] Withdraw money\n\t')
        print('\n\t[C] Exit\n\t')
        transaction = input("Please enter the option above: ")

        if transaction == 'A':
          deposit = input("Please enter the amount that you would like to deposit: ")
            customer_deposit = float(deposit)
            customer_withdraw = 0.00
            customer_balance += float(deposit)
            print("\n\nCustomer ID:\tCustomer Name:\tDeposit Amount:
            \tWithdrawal Amount:\tBalance:")
            print(loginid+"\t\t"+customer_name+"\t\t"+str(customer_deposi
t)+
            "\t\t"+str(customer_withdraw)+"\t\t"+str(customer_balance)+"\n
")
            customer_account.write(loginid+"\t\t"+customer_name+"\t\t"+str(customer_deposit)+
             "\t\t"+str(customer_withdraw)+"\t\t"+str(customer_balance)+"\n")
        elif transaction == 'B':
            withdraw = input("Please enter the amount that you would like to withdraw:
")
            if float(customer_withdraw) <= float(customer_balance):
                customer_withdraw = float(withdraw)
                customer_balance -= float(withdraw)
                print("\n\nCustomer ID:\tCustomer Name:\tDeposit Amount:
                \tWithdrawal Amount:\tBalance:")
                print(loginid+"\t\t"+customer_name+"\t\t"+str(customer_deposit)+
                 "\t\t"+str(customer_withdraw)+"\t\t"+str(customer_balance)+"\n")
                customer_account.write(loginid+"\t"+customer_name+"\t"+str(customer_deposit)+
                 "\t"+str(customer_withdraw)+"\t"+str(customer_balance)+"\n")
            else:
                print("You have insufficient amount. The transaction cannot be done.")
        elif transaction == 'C':
            customer_account.close()
            break
        else:
            print("Incorrect option code. Please try again.")
```

This part allows the customers to view their own transactions. The defined view_transaction is created. The purpose of this function will be printed using print(). The customer username

will then be accepted using input() while the text file will be opened for reading using open(,'r') with its name of columns using print(). For every line in the text file (i.e. for line in myfile), unnecessary character or space will be removed using rstrip(). If the customer username accepted is within the line in the text line (i.e. if detail in line), the customer's transaction will be printed with print(). Once the function carries out its operation, the text file will be closed with close(). Every functions in this part except view_transaction() is built in functions.

```
#function for customer to view transaction
def view_transaction():
    print("\n\t*****View Transaction*****\n\t")
    detail=input("Enter your customer ID:")
    myfile=open("customeraccount.txt","r")
    print("\n\nCustomer ID:\tCustomer Name:\t\tDeposit Amount:
    \t\tWithdrawal Amount:\t\tBalance:")
    for line in myfile:
        line=line.rstrip()
        if detail in line:
            print(line)
    myfile.close()
```

The last part is the activation of the system. A symbolic constant is created with the output of True to assume the online banking system can be validated. While the created symbolic constant is validated, the heading of the page with the operation codes will be printed using print(). The code will then be accepted using input() as integer data type. If the menu code 1 is selected by the administrator, the heading of the operation will be printed using print(). Later on, the administrator has to enter his or her username and password and these will be accepted using input(). If the username and password accepted are aligned with the designated username and password, the heading and options of the operation will be printed. The administrator will then require to enter the option code and the code will be accepted.

If the code selected is 1, the administrator will be directed to the defined function create_newaccount(). If the code selected is 2, defined function view_allprofiles will be shown to the administrator. The defined function search_customer will appear if the administrator selects code 3. Contrary, if the administrator chooses code 4, the defined function search_transaction will be shown. Lastly, if code 5 is entered, the system will be ended. If wrong code is entered, a notification to try again will be printed using print(). The administer has to re-enter the correct code and the code will be accepted again using input().

On the other hand, if the menu code 2 is entered by the customer, the heading of the page will be printed with print(). The customer will then have to enter the given login id and password to access the customer page and both of them will be accepted by the system. If the loginid and password accepted align with the details in the text file (i.e. if check_auth(loginid, password)), the option operations will be printed using print(). Subsequently, the code entered by the customer will be accepted.

If the customer code accepted is 1, the customer will be directed to the deposit_withdrawal() defined function. Contrary, if customer code 2 is selected, the defined function view_transaction will be shown. Lastly, if the customer chooses customer code 3, a notification

of exiting the page will be printed using print() and the online banking system will be ended (i.e., break). However, if the customer entered the wrong login ID or password, a notification will be printed using print() and the customer will be re-directed to the menu page. Moreover, if the wrong code is inserted by the customer, a notification of invalid code will be printed with print() and the customer will be re-directed to the menu page.

```
#loop function for the main page
system_activated = True
while(system_activated == True):dde
    print('\n\t*****Welcome to APU Online Banking Main Page*****\n\t')
    print('\n\tPlease key in the page code:\n\t')
    print('\n\t[1] Admin Page\n\t')
    print('\n\t[2] Customer Page\n\t')
    print('\n\t[3] Quit\n\t')
    menu_code=int(input('Enter the main menu code here:'))

    if menu_code == 1:
        print("\n\t*****Welcome to Admin Management Page*****\n\t")
        admin_username=input('Enter your administrator username: ')
        admin_password=input('Enter your administrator password: ')
        if admin_username == 'APUAdmin' and admin_password == '12345':
            print('\n\t*****Admin Management Page*****\n\t')
            print('\n\tPlease select the below option:\n\t')
            print('\n\t[1] Create New Customer Account\n\t')
            print('\n\t[2] View All Profiles\n\t')
            print('\n\t[3] Search for Specific Customer Account\n\t')
            print('\n\t[4] Search for Specific Customer Transaction\n\t')
            print('\n\t[5] Log Out\n\t')
            admin_page=int(input('Please key in the option code: '))
            if admin_page == 1:
                create_newaccount()
            elif admin_page == 2:
                view_allprofiles()
            elif admin_page == 3:
                search_customer()
            elif admin_page == 4:
                search_transaction()
            elif admin_page == 5:
                break
            else:
                print("You entered wrong option code. Please try again.")
                admin_page=int(input('Please key in the option code: '))

        else:
            print("You entered the wrong username or password. Please try again
")


    elif menu_code == 2:
        print("\n\t*****Welcome to Customer Page*****\n\t")
        loginid=input('Enter your login ID: ')
        password=input('Enter your password: ')

        if check_auth(loginid,password):
```

```
            print('\n\tPlease select the below option:\n\t')
            print('\n\t[1] Deposit OR Withdraw Money\n\t')
            print('\n\t[2] View Transactions\n\t')
            print('\n\t[3] Log Out\n\t')
            customer_page=int(input('Please key in the option code: '))
            if customer_page == 1:
                deposit_withdrawal()
            elif customer_page == 2:
                view_transaction()
            elif customer_page == 3:
                break
        else:
            print('\n\t***Incorrect login ID or password. Please try again***\n\
t')

    elif menu_code == 3:
        print('\n\t***Exit Successful. Thank you for using our service***\n\t
')
        break

else:
    print('\n\t***Invalid Code. Please try again.***\n\t')
```