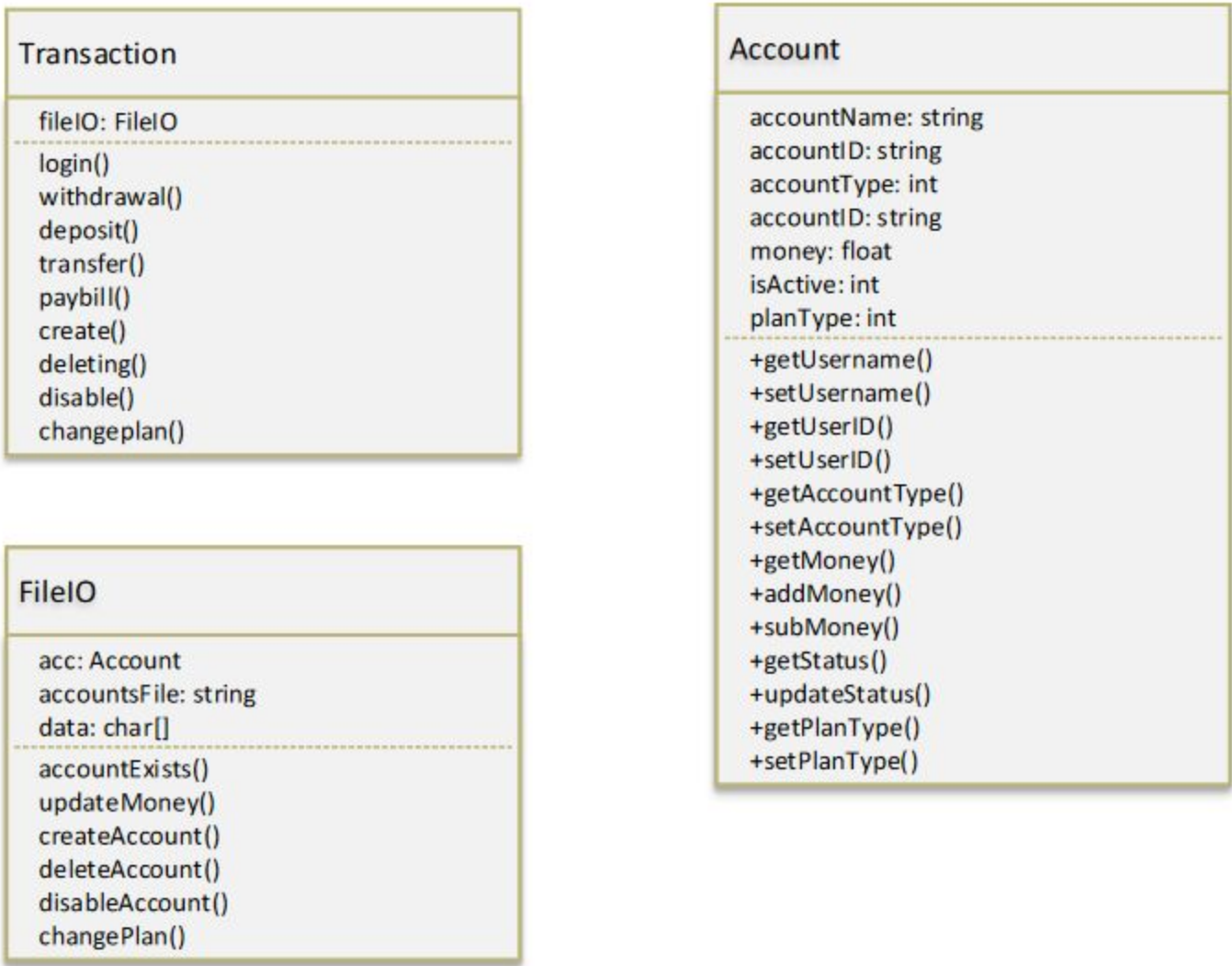You should hand in:

1. A design document for your Front End, giving the overall structure of your solution, showing the classes and methods in a UML class diagram and a brief (one sentence) description of the intention of each method and class in a table.

2. The first version of the source code to implement your design. This version should run on at least some inputs, but should not yet be completely tested (since that will be the next assignment, and it's better to leave some failures until then).

1.

Transaction
fileIO: FileIO
login() withdrawal() deposit() transfer() paybill() create() deleting() disable() changeplan()

Account
accountName: string accountID: string accountType: int accountID: string money: float isActive: int planType: int
+getUsername() +setUsername() +getUserID() +setUserID() +getAccountType() +setAccountType() +getMoney() +addMoney() +subMoney() +getStatus() +updateStatus() +getPlanType() +setPlanType()

FileIO
acc: Account accountsFile: string data: char[]
accountExists() updateMoney() createAccount() deleteAccount() disableAccount() changePlan()

Type	name	function
class	bank	Store bank details
class	account	Store account details

method	login()	Login into class account
method	transfer()	Transfer money from one account to another
method	Pay bill()	Transfer money from account to payee
method	deposit()	Transfer money into account
method	withdraw()	Transfer money out of the account
method	create()	Create a new bank account with an initial balance
method	delete()	Deletes the bank account
method	disable()	Terminates and disables the bank account assuming correct permissions
method	changeplan()	Change the payment plan for the account holder

2.