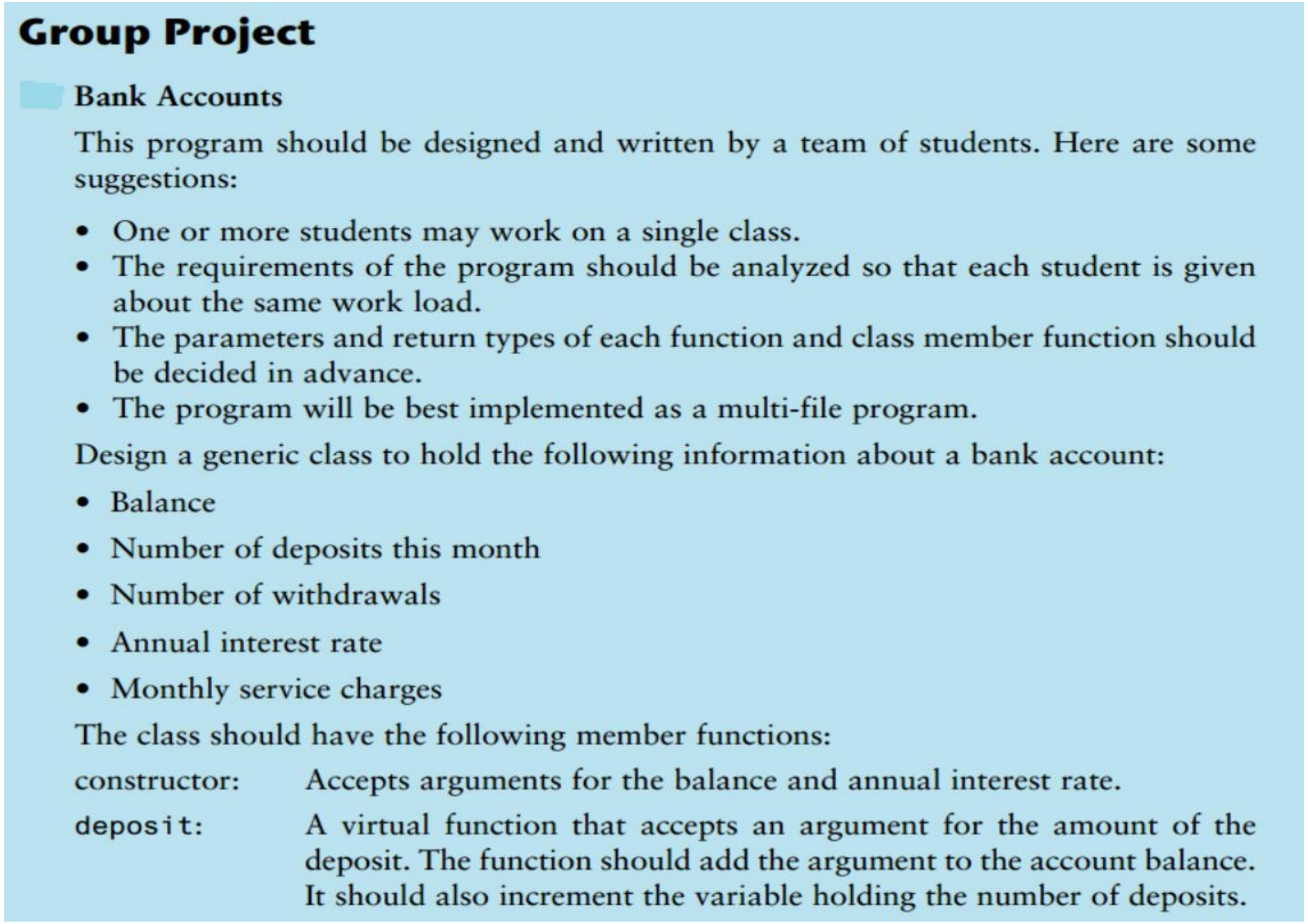
Group Project I

- To be worked in a groups of three elements.
- To be submitted on moodle
- Due date: **02/25/2021**

- Resolve the problem bellow and apply all the concepts from chapter 3, 4, and 5 of the textbook.

Group Project I



Group Project

Bank Accounts

This program should be designed and written by a team of students. Here are some suggestions:

- One or more students may work on a single class.
- The requirements of the program should be analyzed so that each student is given about the same work load.
- The parameters and return types of each function and class member function should be decided in advance.
- The program will be best implemented as a multi-file program.

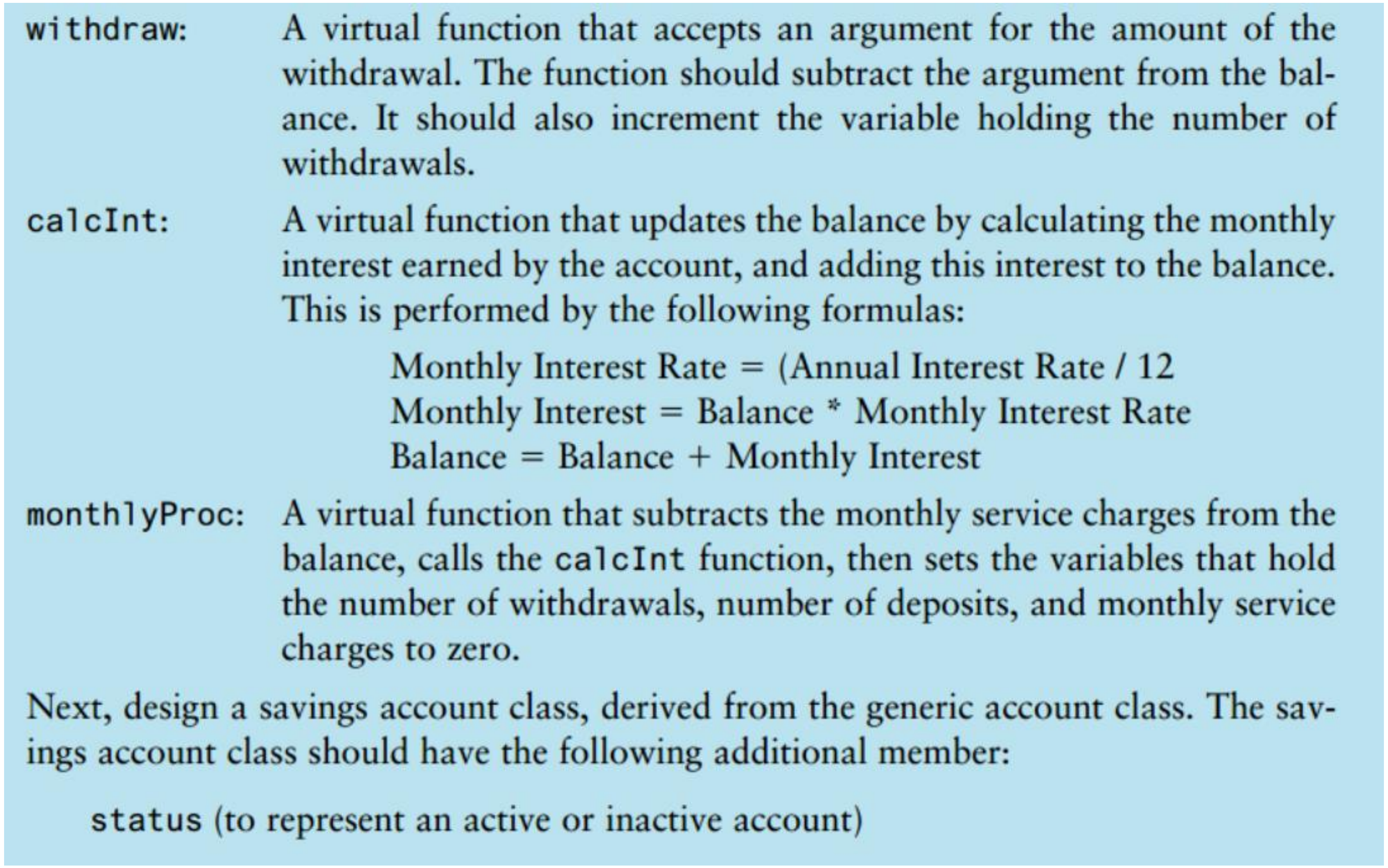
Design a generic class to hold the following information about a bank account:

- Balance
- Number of deposits this month
- Number of withdrawals
- Annual interest rate
- Monthly service charges

The class should have the following member functions:

constructor:	Accepts arguments for the balance and annual interest rate.
`deposit:`	A virtual function that accepts an argument for the amount of the deposit. The function should add the argument to the account balance. It should also increment the variable holding the number of deposits.

Group Project I, cont.

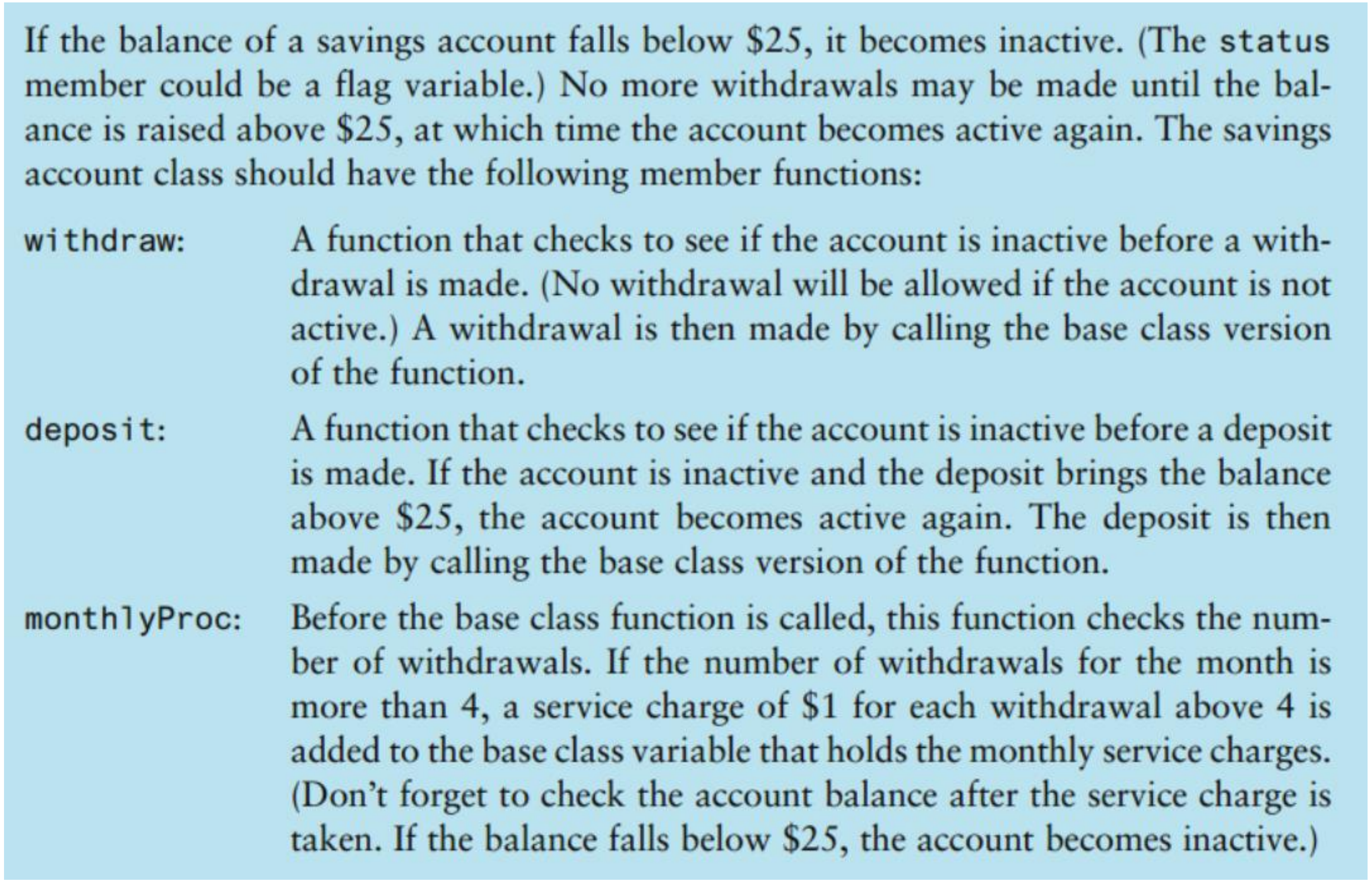


`withdraw:`	A virtual function that accepts an argument for the amount of the withdrawal. The function should subtract the argument from the balance. It should also increment the variable holding the number of withdrawals.
`calcInt:`	A virtual function that updates the balance by calculating the monthly interest earned by the account, and adding this interest to the balance. This is performed by the following formulas: Monthly Interest Rate = (Annual Interest Rate / 12 Monthly Interest = Balance * Monthly Interest Rate Balance = Balance + Monthly Interest
`monthlyProc:`	A virtual function that subtracts the monthly service charges from the balance, calls the `calcInt` function, then sets the variables that hold the number of withdrawals, number of deposits, and monthly service charges to zero.

Next, design a savings account class, derived from the generic account class. The savings account class should have the following additional member:

`status` (to represent an active or inactive account)

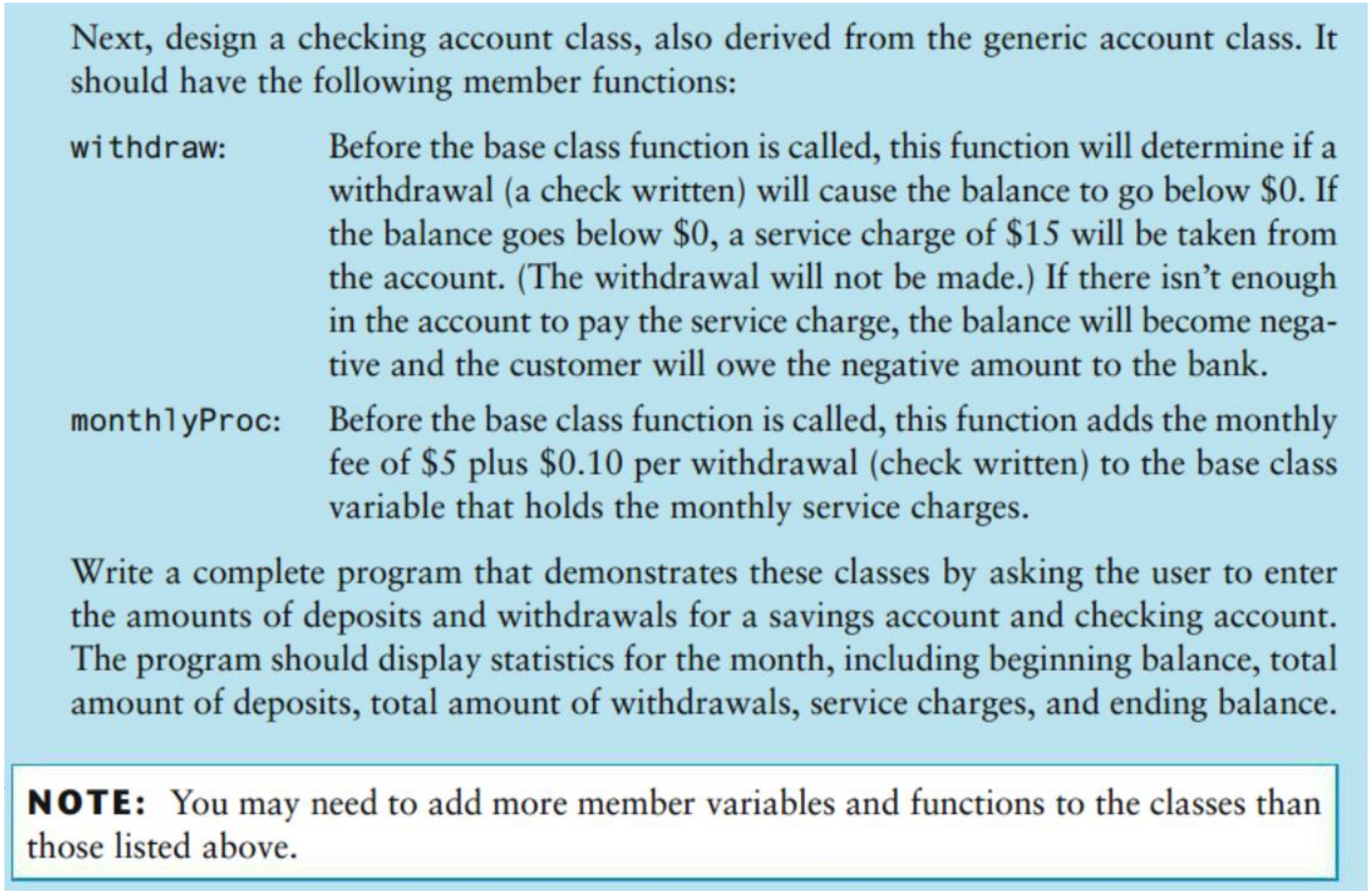
Group Project I, cont.



If the balance of a savings account falls below $25, it becomes inactive. (The `status` member could be a flag variable.) No more withdrawals may be made until the balance is raised above $25, at which time the account becomes active again. The savings account class should have the following member functions:

`withdraw`:	A function that checks to see if the account is inactive before a withdrawal is made. (No withdrawal will be allowed if the account is not active.) A withdrawal is then made by calling the base class version of the function.
`deposit`:	A function that checks to see if the account is inactive before a deposit is made. If the account is inactive and the deposit brings the balance above $25, the account becomes active again. The deposit is then made by calling the base class version of the function.
`monthlyProc`:	Before the base class function is called, this function checks the number of withdrawals. If the number of withdrawals for the month is more than 4, a service charge of $1 for each withdrawal above 4 is added to the base class variable that holds the monthly service charges. (Don't forget to check the account balance after the service charge is taken. If the balance falls below $25, the account becomes inactive.)

Group Project I, cont.



Next, design a checking account class, also derived from the generic account class. It should have the following member functions:

`withdraw`:	Before the base class function is called, this function will determine if a withdrawal (a check written) will cause the balance to go below $0. If the balance goes below $0, a service charge of $15 will be taken from the account. (The withdrawal will not be made.) If there isn't enough in the account to pay the service charge, the balance will become negative and the customer will owe the negative amount to the bank.
`monthlyProc`:	Before the base class function is called, this function adds the monthly fee of $5 plus $0.10 per withdrawal (check written) to the base class variable that holds the monthly service charges.

Write a complete program that demonstrates these classes by asking the user to enter the amounts of deposits and withdrawals for a savings account and checking account. The program should display statistics for the month, including beginning balance, total amount of deposits, total amount of withdrawals, service charges, and ending balance.

NOTE: You may need to add more member variables and functions to the classes than those listed above.