//Dave Makin

//SDEV 120

//Due Date

//Makin Final Project

//Create a program to operate an ATM

//Program Name: ATM OS

//Program version: 1.0

//

Main Module

Call Declare

Call LoadData

Call GeneralWelcome

Call UserWelcome

Call LogIn

Call LogInFail

Call NumberCheck

Call NotNumberCheck

Call ActionMenu

Call MainMenu

Call Deposit

Call Withdrawal

Call Transfer

Call ShowBalance

Call HasEnoughMoney

Call NoCoins

Call UpdateServer

End Main Program

//

// declares and initializes all variables

Declare

Set Count = 1

Set phase[1] = “a”

Set phase[2] = “a”

Set Amt = 0

Set test = 0

Set CusIndex = 1

Set Menu = 0

Set TransCount = 1

Set HasEnoughMoney = false

Set nothing = “nothing”

While Count < 5

Customer[Count] = “a”

Username[Count] = “a”

Password[Count] = “a”

SavBal[Count] = “a”

CheBal[Count] = “a”

Count++

End While

Count = 1

End Declare

//

// assigns all the data from the flat-file database to parallel arrays in the program

LoadData

Redirect\_Input(FinalProjectMakin.txt)

While Count < 5

Input Customer[Count]

Input Username[Count]

Input Password[Count]

Input SavBal[Count]

Input CheBal[Count]

Count++

End While

Set Count = 1

Redirect\_Input(false)

End LoadData

//

// opening screen asking to enter username and password

GeneralWelcome

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* Welcome to our ATM. "

Write "\* Enter your username and "

Write "\* password to begin. "

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

End GeneralWelcome

//

// a more personal greeting including the customer’s name

UserWelcome

Clear\_Console

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* Thanks for coming in today"

Write "\* " + Customer[CusIndex]

Write "\* Click OK to begin."

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write “Click OK to continue”

Input nothing

End UserWelcome

//

// controls the general process flow of validating and verifying log in input

LogIn

Set phase[1] = “username”

NumberCheck

IF Count != 4 THEN

NotNumberCheck

IF Count != 4 THEN

Phase[1] = “password”

NumberCheck

IF Count != 4 THEN

NotNumberCheck

END IF

END IF

END IF

End LogIn

//

// output section informing the user they failed to log in

LogInFail

Clear\_Console

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* You have entered the”

Write "\* incorrect " + phase[1] + " 3 times. "

Write "\* Please see a bank teller"

Write "\* for assistance."

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

End LogInFail

//

// validates input as being or not being a number, collects menu selection input, collects

// transaction amounts

NumberCheck

// 3 tries to enter not a number for username and password

IF phase[1] == “username” OR “password” THEN

Write “Please enter your “ + phase[1] + “ to continue.”

Input test

//continue to get input until they enter a string or try 3 times

WHILE Is\_Number(test) OR Count != 4 THEN

IF Count < 3 THEN

Write “Sorry, that’s the incorrect “ + phase[1] + “. You have + (3 – Count) + “ tries remaining.”

Set Count++

Write “Please enter your “ + phase[1] + “ to continue.”

Input test

//

ELSE

//

Write “Sorry, that’s the incorrect “ + phase[1] + “. This is your last attempt.”

Set Count++

Write “Please enter your “ + phase[1] + “ to continue.”

Input test

END IF

END IF

//returns to LogIn if it was in the username or password phase

ELSE

IF Phase[1] == “menu” OR phase[2] == “mainmenu” THEN

IF phase[2] == “mainmenu” THEN

//MainMenu input and validation

//happens when phase[2] = “mainmenu”

Write "Enter Menu selection"

Input Menu

WHILE NOT Is\_Number(Menu) OR Menu < 0 OR Menu > 6

Write "Invalid selection. You must choose”

Write “option 1, 2, 3, 4, or 5 by entering the”

Write “numbers 1, 2, 3, 4, or 5 and pressing”

Write “enter."

Input Menu

END WHILE

ELSE

//input and validation of account selection for all transaction types

//happens when phase[1] = “menu”

Write "1- Checking 2- Savings 3- Main Menu"

Input Menu

WHILE NOT Is\_Number(Menu) OR Menu < 0 OR Menu >3

Write "You must choose option 1, 2, or 3 by”

Write “entering the numbers 1, 2, or 3 and”

Write “pressing enter."

Input Menu

END WHILE

END IF

ELSE

//input and validation for Amt in any transaction

//called from Deposit, Withdrawal, or Transfer

Write "How much would you like to " + phase[2] + " ($500 limit)?"

Input Amt

WHILE NOT Is\_Number(Amt) OR ABS(Amt) != Amt OR Amt > 500 OR Amt = 0

IF Amt > 500 THEN

Write "You entered $" + Amt + " which is more than the $500 limit."

Write "Please enter an amount under $500."

Input Amt

END IF

Write "You must enter a number here. Letters or negative/zero amounts are not allowed."

Input Amt

END IF

END IF

End NumberCheck

//

// once validated as not being a number, determines if the username and password are correct and

// establishes CusIndex

NotNumberCheck

IF phase[1] == “username”

//Username verification section

Phase[1] is now set to “username”

//unless all tries have been used, check the username entered against allowed values

//drops from loop and establishes correct CusIndex value when match found

WHILE Count < 4 OR test != Username[CusIndex]

IF Count != 4 OR Username[CusIndex] != test THEN

WHILE Username[CusIndex] != test

CusIndex++

END WHILE

IF test != Username[CusIndex] THEN

IF Count < 3 THEN

Write "Sorry, that's the incorrect " + phase[1] + ". You have " + ( 3 - Count) + " tries remaining."

Count++

// wrong answer, got get new input

NumberCheck

ELSE

Write "Sorry, that's the incorrect " + phase[1] + ". This is your last attempt."

Count++

// wrong answer, go get new input

NumberCheck

END IF

END IF

END IF

CusIndex++

END WHILE

IF test == Username[CusIndex] THEN

Set Count = 1

// once a match has been found, reset Count so the password iteration gets 3 chances

END IF

ELSE

//Password verification section

//phase[1] is now set to “password”

WHILE Count != 4 AND test != Password[CusIndex]

IF test != Password[CusIndex] THEN

//if count = 4, this section will be bypassed IF Count < 3 THEN

Write "Sorry, that's the incorrect " + phase[1] + ". You have " + ( 3 - Count) + " tries remaining."

Count++

// wrong answer, got get new input

NumberCheck

ELSE

IF Count = 3 THEN

Write "Sorry, that's the incorrect " + phase[1] + ". This is your last attempt."

Count++

// wrong answer, go get new input

NumberCheck

END IF

END IF

END IF

END WHILE

END IF

End NotNumberCheck

//

// generates output when a user returns to the main menu, completes three transactions,

// or quits the program, and displays the prompt to enter an amount for a transaction

ActionMenu

IF Menu == 3 THEN

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* No transaction made”

Write "\* Click OK to continue”

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "Click OK to continue"

Input nothing

ELSE

// chooses between end program options or transaction account selection

IF phase[1] == “mainmenu” THEN

// chooses between manually quitting or having 3 transactions

IF TransCount == 4 THEN

//completed 3 transactions

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* You have completed three"

Write "\* transactions in this session. To "

Write "\* continue you must log in again. "

Write "\* Thank you for using the”

Write "\* ATM Service.”

Write "\* Have a nice day.”

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

ELSE

//chose option 5 to quit program

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* Thank you for using our ATM. "

Write "\* Have a nice day”

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

END IF

//prompt for amount to deposit, withdraw, or transfer

ELSE

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* You may " + phase[2] + " any amount up to $500. "

IF phase[2] == “transfer” THEN

Write "\* Select an account to transfer from. "

ELSE

Write "\* Select an account for " + phase[2]”.”

END IF

Write "\* 1- Checking”

Write "\* 2- Savings”

Write "\* 3- Return to main menu”

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

END IF

END IF

End ActionMenu

//

// provides choices for available transaction types or to quit the program

MainMenu

Clear\_Console

WHILE TransCount != 4 AND Menu != 5

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* 1 Deposit”

Write "\* 2 Withdrawal”

Write "\* 3 Balance Inquiry”

Write "\* 4 Transfer Balance”

Write "\* 5 Log Out”

Write "\* Select an option (1 - 5)"

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Set phase[2] = “mainmenu”

NumberCheck

IF Menu == 1 THEN

Clear\_Console

Deposit

ELSE

IF Menu == 2 THEN

Clear\_Console

Withdrawal

ELSE

IF Menu == 3 THEN

Clear\_Console

ShowBalance

ELSE

IF Menu == 4 THEN

Clear\_Console

Transfer

ELSE

Clear\_Console

END IF

END IF

END IF

END IF

END WHILE

//set phase for ActionMenu to present correct program ending dialog

Set phase[1] = “mainmenu”

End MainMenu

//

// controls deposit transactions

Deposit

Clear\_Console

Set phase[1] = “menu”

Set phase[2] = “deposit”

ActionMenu

NumberCheck

IF Menu == 3 THEN

Clear\_Console

ActionMenu

ELSE

Set phase[1] = “amt”

NumberCheck

END IF

//Skip this loop if returning to the main menu

WHILE Menu != 3

IF Amt > Floor(Amt) THEN

NoCoins

Clear\_Console

Deposit

ELSE

Clear\_Console

IF Menu == 1 THEN

Set CheBal[CusIndex] = CheBal[CusIndex] + Amt

ShowBalance

ELSE

Set SavBal[CusIndex] = SavBal[CusIndex] + Amt

ShowBalance

END IF

TransCount++

END IF

Set Menu = 3

END WHILE

Clear\_Console

End Deposit

//

// controls withdrawal transactions

Withdrawal

Clear\_Console

Set phase[1] = “menu”

Set phase[2] = “withdraw”

// display account selection options

ActionMenu

//select account to withdraw from

NumberCheck

IF Menu == 3 THEN

Clear\_Console

//display return to main menu dialog

ActionMenu

ELSE

Set phase[1] = “amt”

// get and validate amount to withdraw

NumberCheck

//check if they have enough money

HasEnoughMoney

END IF

WHILE HasEnoughMoney == True

IF Amt > Floor(Amt) THEN

NoCoins

Withdrawal

ELSE

Clear\_Console

IF Menu == 1 THEN

Set CheBal[CusIndex] = CheBal[CusIndex] – Amt

ShowBalance

TransCount++

ELSE

Set SavBal[CusIndex] = SavBal[CusIndex] – Amt

ShowBalance

TransCount++

END IF

END IF

END WHILE

End Withdrawal

//

// controls transfer transactions

Transfer

Clear\_Console

Set phase[1] = “menu”

Set phase[2] = “transfer”

// display account selection options

ActionMenu

//select account to withdraw from

NumberCheck

IF Menu == 3 THEN

Clear\_Console

//display return to main menu dialog

ActionMenu

ELSE

Set phase[1] = “amt”

// get and validate amount to withdraw

NumberCheck

//check if they have enough money

HasEnoughMoney

END IF

WHILE HasEnoughMoney == True

IF Menu == 1 THEN

Set CheBal[CusIndex] = CheBal[CusIndex] – Amt

Set SavBal[CusIndex] = SavBal[CusIndex] + Amt

ELSE

Set SavBal[CusIndex] = SavBal[CusIndex] – Amt

Set CheBal[CusIndex] = CheBal[CusIndex] + Amt

END IF

ShowBalance

TransCount++

END WHILE

End Transfer

//

// output that shows the available balance of both accounts

ShowBalance

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* Checking balance is $" + CheBal[CusIndex]

Write "\* Savings balance is $" + SavBal[CusIndex]”

Write "\* Click OK to continue.”

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Input nothing

Clear\_Console

End ShowBalance

//

// determines if accounts have a sufficient balance to support the chosen transaction

HasEnoughMoney

Set HasEnoughMoney = true

IF Amt > CheBal[Cusindex] AND Menu = 1 OR Amt > SavBal[CusIndex] AND Menu = 2 THEN

Clear\_Console

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* You have insufficient funds”

Write "\* to " + phase[2] + " that amount.”

Write "\* Checking balance is $" + CheBal[CusIndex] + ".”

Write "\* Savings balance is $" + SavBal[CusIndex] + ".”

Write "\* Click OK to continue.”

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Input nothing

Set HasEnoughMoney = false

Clear\_Console

IF phase[2] = “withdrawal” THEN

Withdrawal

ELSE

Transfer

END IF

End HasEnoughMoney

//

// output telling the user they must use whole bill amounts

NoCoins

Clear\_Console

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "\* You may not " + phase[2] + " coins at this. "

Write "\* ATM. Please " + phase[2] + " bills only.”

Write "\* Click OK to continue.”

Write "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

Write "Click OK to continue"

Input nothing

Clear\_Console

End NoCoins

//

// sends all the data back out to the flat-file database to keep track of changes between sessions

UpdateServer

Redirect\_Output(FinalProjectMakin.txt)

While Count < 5

Write Customer[Count]

Write Username[Count]

Write Password[Count]

Write SavBal[Count]

Write CheBal[Count]

Count++

End While

Set Count = 1

Redirect\_Output(false)

End UpdateServer