Introduction to C - Programming Assignment #2

Due: 03/14/14

Objective

1. To give students practice in using loops and branching statements

Program A : To develop a simpler version of Banking system

Moon Trust Bank wants to develop a small banking system for a specific customer. The system is supposed to keep track of all deposits and amounts withdrawn over the period of one month, October, 2004. It is specified that the current balance of the customer on the morning of day 1 is $2000.00. The bank encourages the customer to keep his balance as high as possible. Towards this end, the bank provides an incentive, in the form of paying interest on the minimum balance maintained over the month

The banking system should provide the following options:

1) Deposit funds (credit transaction)
2) Withdraw funds (debit transaction)
3) Print statement of account
4) Compute interest and exit the program

.

The program should prompt the user to enter an option. It is guaranteed that the user will enter a valid option.

Option1:

Prompt the user to enter a date. It is guaranteed that the user will enter a date between 1 and 31.

Prompt the user to enter the amount that he wants to credit to his account. It is guaranteed that the user will enter a positive value.

Save the information about the day and the balance using two variables.

Display the balance after crediting the current amount.

Option 2:

Prompt the user to enter a date. It is guaranteed that the user will enter a date between 1 and 31.

Prompt the user to enter the amount he wants to withdraw. It is guaranteed that the user will enter a positive value.

If it exceeds the balance, print the balance and prompt him to enter a smaller amount. It is guaranteed that next time the user will not enter a value more than the current balance. Update and print the current balance.

Option 3:
Prompt for the date.
Print the current balance, total number of credit transactions and total number of debit transactions made so far in the month.

Note: *You are guaranteed that the user will enter the day of each transaction in non-decreasing order. (So, when executing option 3, you won't ever have to go "back in time" to determine the balance at a prior date.)*

Note*: Also, at the end of option 1, 2 and 3 ask the user for his/her next option*

Option 4:
Don't prompt the user for the date. It is assumed that option 4 is given only on the last day of the month.
Compute interest for the month, based on the minimum balance of the month at the rate of 2% per year.
Print number of credit transactions, number of debit transactions in the month.
Calculate and print the interest.
Print the final balance.
Print "Good Bye!" and quit the program

You have to keep track of the minimum balance variable and update it after each transaction.

Output Sample
Here is one sample output of running the program. Note that this test is NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given. The user input is given in *italics.*

Sample Run Program A
Your current balance is $2000.00
Please indicate your option: *1*
Please enter a valid date from 1 to 31 : *6*
How much do you want to deposit? *1000.0*
Your current balance is $ 3000.00

Please indicate your option: *2*
Please enter a valid date from 6 to 31 :*10*
How much do you want to withdraw? *1300.00*
Your current balance is $ 1700.00

Please indicate your option: *1*
Please enter a valid date from 10 to 31 :*18*
How much do you want to deposit? *600.00*
Your current balance is $ 2300.00

Please indicate your option: *3*
Please enter a valid date from 18 to 31 :*24*

Your current balance is $ 2300.00
Number of credit transactions: 2
Number of debit transactions: 1

Please indicate your option: *4*

Statement of Account for October, 2004

Number of credit transactions: 2
Number of debit transactions: 1
Interest computed: $ 2.83
Final balance: $ 2302.83

Good Bye!

Program B : Taking care of valid dates in the Banking System.

This program builds up on Program A. In this part we want to make sure that the user enters a valid date of the month, and further that the date can not be less than the dates entered previously by the user.

The banking system should provide the following options:
1) Deposit funds (credit transaction)
2) Withdraw funds (debit transaction)
3) Print statement of account
4) Compute interest and exit the program

The program should prompt the user to enter an option. It is guaranteed that the user will enter a valid option.

For options 1, 2 and 3, ask the user to enter a valid date of transaction. The user will enter an integer. If the user enters a value less than 1 or more than 31, prompt him to enter a valid date again. The valid dates for various transactions must appear in increasing order. Thus, after a transaction is carried out on a particular day, say 10^{th} day, for the next transaction, the user can not enter a date integer less than 10. He is allowed to enter 10, as more than one transaction can take place on the same day. The balance is to be updated after every (credit or debit) transaction.

The options are to be handled as mentioned below:

Option1:
Prompt the user to enter a valid date between the date of last transaction and end of the month. If the last transaction was done on 6th October, then ask the user to input a date from 6 to 31.
Check that the valid date is not less than previous transaction date.
More than one transaction can take place on the same day.
Prompt the user to enter the amount that he wants to credit to his account. It is guaranteed that the user will enter a positive value.
Save the information about the day and the balance using two variables. Display the balance after crediting the current amount.

Option 2:
Prompt the user to enter a valid date. Check that the valid date is not less than previous transaction date. More than one transaction can take place on the same day.
Prompt the user to enter the amount he wants to withdraw. It is guaranteed that the user will enter a positive value.
If it exceeds the balance, print the balance and prompt him to enter a smaller amount. It is guaranteed that the next time the user will not enter a value more than the current balance.
Update and print the current balance.

Option 3:
Prompt for the valid date.
If the date entered is 31, carry out the steps mentioned below for option 4, otherwise print the current balance, total number of credit transactions and total number of debit transactions made so far in the month.

***Note**: At the end of option 1, 2 and 3 ask the user for his next option.*

Option 4:
Don't prompt the user for the date. It is assumed that option 4 is given only on the last day of the month. Compute interest for the month, based on the minimum balance of the month at the rate of 2% per year. Print the number of credit transactions and the number of debit transactions in the month. Calculate and print the interest. Print the final balance. Print "Good Bye!" and quit the program

You have to keep track of the minimum balance variable and update it after each transaction. You also have to keep track of the old transaction date and the new transaction date, and ensure that the new date is not smaller than the old date.

Output Sample Program B
Here are two sample outputs of running the program. Note that this test is NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given. The user input is given in *italics.*

Sample Run #1

Your current balance is $2000.00
Please indicate your option: *1*
Please enter a valid date from 1 to 31 : *6*
How much do you want to deposit? *1000.0*
Your current balance is $ 3000.00

Please indicate your option: *2*
Please enter a valid date from 6 to 31 :*42*
Please enter a valid date from 6 to 31 :*1*
Please enter a valid date from 6 to 31 :*10*
How much do you want to withdraw? *4200.00*
You can not withdraw more than $3000.00.
How much do you want to withdraw? *1300.00*
Your current balance is $ 1700.00

Please indicate your option: *1*
Please enter a valid date from 10 to 31 :*18*
How much do you want to deposit? *600.00*
Your current balance is $ 2300.00

Please indicate your option: *3*
Please enter a valid date from 18 to 31 :*24*

Your current balance is $ 2300.00
Number of credit transactions: 2
Number of debit transactions: 1

Please indicate your option: *4*

Statement of Account for October, 2004

Number of credit transactions: 2
Number of debit transactions: 1
Interest computed: $ 2.83
Final balance: $ 2302.83

Good Bye!

Sample Run #2

Your current balance is $2000.00

Please indicate your option: *1*
Please enter a valid date from 1 to 31 : *3*
How much do you want to deposit? 700.00
Your current balance is $ 2700.00

Please indicate your option: *1*
Please enter a valid date from 3 to 31 : *2*
Please enter a valid date from 3 to 31 : *6*
How much do you want to deposit? 300.00
Your current balance is $ 3000.00

Please indicate your option: *2*
Please enter a valid date from 6 to 31 :*42*
Please enter a valid date from 6 to 31 :*1*
Please enter a valid date from 6 to 31 :*10*
How much do you want to withdraw? *1300.00*
Your current balance is $ 1700.00

Please indicate your option: *3*
Please enter a valid date from 10 to 31 :*14*

Your current balance is $ 1700.00
Number of credit transactions: 2
Number of debit transactions: 1

Please indicate your option: *1*
Please enter a valid date from 14 to 31 :*18*
How much do you want to deposit? *600.00*
Your current balance is $ 2300.00

Please indicate your option: *3*
Please enter a valid date from 18 to 31 :*31*

Statement of Account for October, 2004

Number of credit transactions: 3
Number of debit transactions: 1
Interest computed: $ 2.83
Final balance: $ 2302.83

Good Bye!

Write a program that asks the user to enter a positive odd integer, n.
If the integer is 0 or less, print out a message saying the integer is too small and ask the user to reenter the integer.
If the integer is even, print out a message saying that the integer must be odd and ask the user to reendter the integer.
When the user enters a valid input (continue prompting them until they do) then print out a diamond with n rows. Here is an example with n = 5:

```
  *
 ***
*****
 ***
  *
```

Notice that not only must the number of stars on each row be accurate, but so must the number of spaces.

Sample Output

```
Please enter a positive odd integer.
-3
Sorry, that integer is too small.
Please enter a positive odd integer.
4
Sorry, the integer you enter must be odd.
Please enter a positive odd integer.
7
   *
  ***
 *****
*******
 *****
  ***
   *
```

Restrictions

Your programs should include a header comment with the following information: your name, course number, section number, assignment title, and date. Also, make sure you include ample comments throughout your code describing the major steps in solving the problem.

Please name your solution to part A: bank.c
Please name your solution to part B: bankplus.c
Please name your solution to part C: diamond.c

Grading Details

Your program will be graded upon the following criteria:

1) Your correctness
2) Your programming style and use of white space. (Even if you have a plan and your program works perfectly, if your programming style is poor or your use of white space is poor you could get 10% or 15% deducted from your grade.)
3) Compatibility to either Codeblocks or Dev C++. (If your program does not compile in either of these environments, you will get a **sizable** deduction from your grade.)