

## APS 105 – Computer Fundamentals

Lab #4: Methods

Fall 1999

(To be completed *before* your lab period, week of October 4)

### Objective

To introduce methods as a way to organize your program, and to practice the basics of Java programming.

### 1 A Bank Account Management Program

You are to construct a program that is to be used to keep track of the balances of two bank accounts: a savings account and a checking account. The program will allow for deposits, withdrawals and interest payments to be made on either account.

To begin with, the program will initialize the balance of each account to \$0. The program will then continuously prompt the user to select an action. The choices will be to make a deposit, to make a withdrawal, to calculate and deposit interest, or to quit the program. If either of the first three options are selected, the user is asked which account the action is applied to. **Each of the deposit, withdrawal and interest calculations actions should be implemented as a separate method that takes the current balance of the account as one of its parameters and returns the modified balance as its result.** Each method may have other parameters, in addition to the current balance, if you desire.

You must not use *global* variables. That is, all variables referenced must be declared within the body of a method (between the { and } characters) or as a parameter to the method (between the ( and ) characters that occur after the method name). Furthermore, the two variables which you will use to store the current balance of each account *must be declared in the body of main.* **You will be penalized if you do not structure your program in this way.**

The *deposit method* will prompt for a deposit amount, and then return the new account balance. If the deposit is a negative value, ask for a new amount.

The *withdrawal method* will prompt for a withdrawal value, and return the new account balance. If the withdrawal is negative or greater than the account balance, ask for new amount.

The *interest method* will calculate the interest earned on the account and return the new balance. Since interest should only be calculated at regular intervals, the interest calculation would be used only by bank employees. To prevent an unauthorized user from asking for interest, *this command should be protected by a numerical password.* The password number can be a constant hard-coded into your program; when the user requests an interest calculation, your program should prompt the user for the password, and check it.

The two accounts have separate interest rates, which you may hard-code in the program as constants. It is best to pass this as a parameter to your interest method. To calculate the interest, apply the interest rate to the *lowest account balance since the last time interest was applied.* Note that no interest will be earned the

1

first time, because the opening balance is always zero. However, the bank employee may ask for interest to be credited as often as desired.

An example of the use of the program is shown below. The program should accept either upper or lower case letters to specify termination, e.g., either Q or q for (Q)uit.

```
--- Bank Account Management Program ---
Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? D
(S)avings or (C)hequing? s
Amount of deposit? 1200
Savings account balance: $1200.00

Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? I
Please enter special password: 76567
Interest Access Allowed.
(S)avings or (C)hequing? c
2.5% interest applied to $0.00 = $0.00
Chequing account balance: $0.00

Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? d
(S)avings or (C)hequing? c
Amount of deposit? 2000
Chequing account balance: $2000.00

Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? I
Please enter special password: 77887
Interest Access Denied.

Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? I
Please enter special password: 76567
Interest Access Allowed.
(S)avings or (C)hequing? s
1.5% interest applied to $0.00 = $0.00
Savings account balance: $1200.00

Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? I
Please enter special password: 76567
Interest Access Allowed.
(S)avings or (C)hequing? s
1.5% interest applied to $1200.00 = $18.00
Savings account balance: $1218.00

Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? I
Please enter special password: 76567
Interest Access Allowed.
(S)avings or (C)hequing? s
1.5% interest applied to $1218.00 = $18.27
Savings account balance: $1236.27

Interaction: (W)ithdrawal, (D)eposit, (I)nterest or (Q)uit? q
Savings account balance: $1236.27
Chequing account balance: $2000.00

End Of Program
```

2

## 2 Hints and Notes

The bank account should not hold fractions of a penny after applying interest, and users should not be allowed to deposit or withdraw fractions of a penny. To properly account for this, **store the bank account balance as an integer** representing the number of cents in the account (rather than the number of dollars). However, when printing the account balance, it must be printed in dollars (don't worry if you can't print out all of the trailing zeros after the decimal). Likewise, deposits and withdrawals should be done in dollar units. For example, inside your program, convert to cents by multiplying by 100 and *casting* the result to an `int` as follows:

```
double dollars = StdIn.getDouble();
int cents = (int)(dollars * 100);
```

In addition to the routines `StdIn.getInt()` and `StdIn.getDouble()`, which return an `int` and `double` type, respectively, you will also need to use `StdIn.getChar()`, which returns a *char* type. Just like `int` and `double` variables, you can create `char` variables. These `char` variables can be compared to other `char` variables, or compared to single characters which you must place inside single quotes in your program.

The `Average`.java program from lab 3 used `char` variables, as does the following example. Notice the use of single quotes to represent one character, and how both `Q` and `q` must be handled separately.

```
// This program writes a prompt to the screen,
// asking the user to type Q to quit. When the
// user finally types Q and ENTER, the program
// prints another message and then quits.
class PromptForQChar
{
    public static void main( String[] args )
    {
        char inputChar;
        do
        {
            System.out.print("Enter Q to quit: ");
            inputChar = StdIn.getChar();
        } while( inputChar != 'Q' && inputChar != 'q' );
        System.out.println("We're done now, thanks for typing Q.");
    }
}
```

3

## 3 Solution to Lab 2, Part 4

```
// Ask for dollar and cent's values.
// Compute how many coins are needed
// to make up that value.
class Change
{
    public static void main( String[] args )
    {
        int dollars, cents;

        System.out.println( "A Change Making Program" );
        System.out.print( "How many dollars? " );
        dollars = StdIn.getInt();
        System.out.print( "How many cents? " );
        cents = StdIn.getInt();

        // compute total value
        // in case cents > 99, cents < 0, etc.
        int value = dollars * 100 + cents;

        // recompute dollars and cents,
        // in case cents > 99, cents < 0, etc.
        dollars = value / 100;
        cents = value % 100;

        int loonies = dollars / 2;
        int quarters = dollars / 25;
        int dimes = (cents%25) / 10;
        int nickels = ((cents-quarters*25)%10) / 5;
        int pennies = (cents%5);

        System.out.println();
        System.out.print( "Total value $" );
        System.out.print( dollars );
        System.out.print( ".*" );
        System.out.println( cents );

        System.out.println( cents );
        System.out.print( "You will need * );
        System.out.print( " loonies );
        System.out.println( " Toonies" );
        System.out.print( "You will need * );
        System.out.print( " loonies );
        System.out.println( " loonies" );
        System.out.print( "You will need * );
        System.out.print( " quarters );
        System.out.println( " Quarters" );
        System.out.print( "You will need * );
        System.out.print( " quarters );
        System.out.println( " quarters" );
        System.out.print( "You will need * );
        System.out.print( " dimes );
        System.out.println( " Dimes" );
        System.out.print( "You will need * );
        System.out.print( " nickels );
        System.out.println( " Nickels" );
        System.out.print( "You will need * );
        System.out.print( " pennies );
        System.out.println( " Pennies" );
    }
}
```

4