Austin Peay State University, Tennessee Spring 2016
CSCI 2070: Programming in Selected Languages I
Dr. Chenchutta Jackson

CSCI 2070 Assignment 4

Q1: **Savings Account Class** [25 points]
Note: For this question, name your java class `<my name>SavingsAccount.` For example, my java class would be named: `ChenchuttaJacksonSavingsAccount`. Take note of requirements for arguments and returns.
Create class **SavingsAccount**. Use a static variable annualInterestRate to store the annual interest rate for all account holders. Each object of the class should have a private instance variable savingsBalance indicating the amount the saver currently has on deposit.
Methods -
calculateMonthlyInterest to calculate the monthly interest by multiplying the savingBalance by annualInterestRate divided by 12- this interest should be added to savingsBalance.
withdraw to provide the functionality that will allow a user to withdraw from the savings account. If the withdrawal will cause the balance to go below $100.00 a prompt should tell the user that her account will be closed and a check will be sent to her from the bank
A static method **modifyInterestRate** that sets the annualInterestRate to a new value.

Write a program to test class <your name>SavingsAccount.
Instantiate two SavingAccount objects, one with a balance of 5000.00 and the other with 2500.00, respectively. Set annualInterestRate to 5%, then calculate the monthly interest and print the new balances for both savers. Then set the annualInterestRate to 7%, calculate the next month's interest and print the new balances for both savers. Also test the **withdraw** method by causing one of the accounts to go below $100.

Hence, the java filename should be `ChenchuttaJacksonSavingsAccount.java`.
You also need to write a separate program called `<my name>SavingsDemo.java` to test the class.
Important: If you do not put `<my name>` to the above mentioned fields (class name and filename), **you will get 0 points for the question.**

Estimated time: 2 hours

Q2: **Guessing Game GUI** [25 points]
Write a GUI application that plays "guess the number" as follows: Your application chooses the number to be guessed by selecting an integer at random in the range 1-2000. The application displays the following in a label:

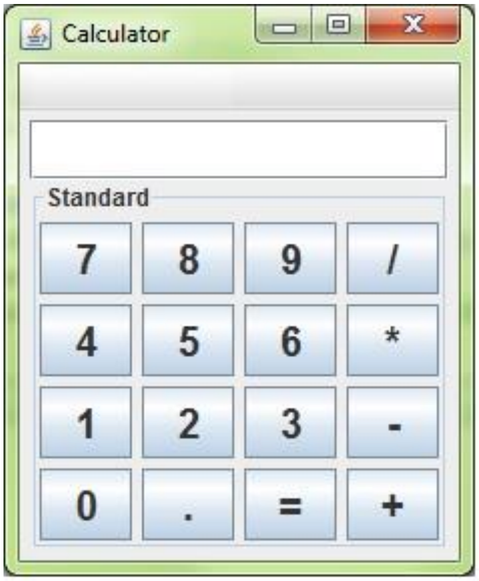
> I have a number between 1 and 2000. Can you guess my number?
> Please enter your first guess.

A JTextField should be used to input the guess. As each guess is input, the background color should change to either red or blue. Red indicates that the user is getting "warmer", and blue indicates the user is getting "colder". A JLabel should display either "Too High" or "Too Low" to help the user zero in on the correct answer. When the user gets the correct answer, "Correct" should be displayed, and the JTextField used for input should be changed to be uneditable. A JButton should be provided to allow the user to play the game again. When the JButton is clicked a new random number should be generated and the input JTextField changed to be editable.

Note: For this question, name your java class `<my name>GuessGameGUI`. For example, my java class would be named: `ChenchuttaJacksonGuessingGameGUI`.
Hence, the java filename should be `ChenchuttaJacksonGuessGameGUI.java`. (You may use one or two java files.)
Important: If you do not put `<my name>` to the above mentioned fields (class name and filename), **you will get 0 points for the question.**

Estimated time: 3 hours

Q3: **Calculator GUI** [25 points]
`<Your full name>`'s Calculator has the following function:
Given two numbers and an operator calculate the answer for +, -, *, and /.
Do not allow division by zero.



Create a **GUI application** that

- Display `<Your full name>`'s Basic Calculator
- Display the mathematic expression or equation the user entered in the text field
- Display the result in the area where the user inserted the equation(in the text field)
- Calculate and display the answer to `<Your full name>`'s Basic Calculator.

Note: For this question, name your java class `<my name>Calculator`. For example, my java class would be named: `ChenchuttaJacksonCalculator`.
Hence, the java filename should be `ChenchuttaJacksonCalculator.java`. (You may use one or two java files.)
Important: If you do not put `<my name>` to the above mentioned fields (class name and filename), **you will get 0 points for the question.**

Estimated time: 3 hours

Q4: **Profit Class** [25 points]

`<Your full name>`'s Space Travel Company needs to store the company profit for each of the 12 months in 2015. Write a class that stores the profit for each of the 12 months in 2015 into an array of **doubles**. Profit cannot be negative (yes, you always make money). The program should have getters, setters and the methods that return the following:

- The total profit for the year
- The average monthly profit
- The month with the highest profit
- The month with the lowest profit

Demonstrate the class in a separate program

- Display `<Your full name>`'s Space Travel Company
- Ask for user input, use setter methods to store the input values
- Profit cannot be negative (yes, you always earn money)
- Use getter methods to retrieve data
- Demonstrate the above mentioned four methods (total profit, average profit, highest profit, lowest profit)

Note: For this question, name your java class `<my name>Profit`. For example, my java class would be named: `ChenchuttaJacksonProfit`.
Hence, the java filename should be `ChenchuttaJacksonProfit.java`.
You also need to write a separate program call `<my name>ProfitDemo.java` to test the class.
Important: If you do not put `<my name>` to the above mentioned fields (class name and filename), **you will get 0 point for the question.**

Estimated time: 4 hours

Submission instructions:
You need to compile the above programs (questions) separately, and provide **two test cases** (if applicable) for each program (question). Do a screen capture of the input and related output for each test case. Use any graphic editing software (e.g. Microsoft Paint, Adobe Fireworks) to cut out the program input and output (from the screen capture), paste them into a word document under a related question number, save the document as a pdf file. A sample input/output (screen capture) can be found at the end of this document.
You need to submit the following:
1. A pdf file containing the screen captures of program input and output of all test cases, name the file **lastname_firstname_assignment04.pdf**.
2. All java files (.java files only). Zip your java files into a single zip file (or rar file) **lastname_firstname_assignment04.zip**.

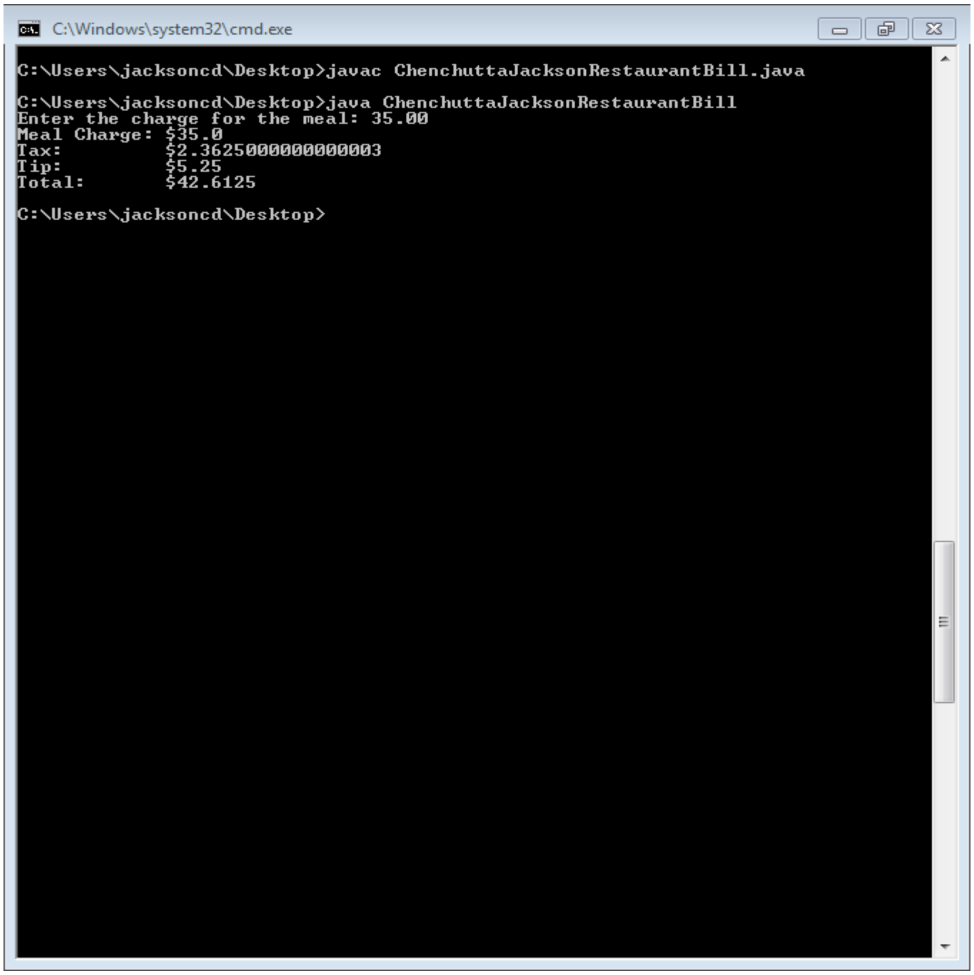
Please submit electronic copy (the above mentioned **two files**) to D2L digital dropbox.

Grading guidelines (programming questions):
Your programs will be judged on several criteria, which are shown below.

- Correctness (50%): Does the program compile correctly? Does the program do what it's supposed to do?
- Design (20%): Are operations broken down in a reasonable way (e.g. classes and methods)?
- Style (10%): Is the program **indented** properly? Do variables have **meaningful names**?
- Robustness (10%): Does the program handle erroneous or unexpected input gracefully?
- Documentation (10%): Do all program files begin with a **comment** that identifies the author, the course code, and the program date? Are all the classes, methods and data fields clearly **documented (comments)?** Are unclear parts of code **documented (comments)?** (Some items mentioned may not apply to some languages)

A program that does not compile will get at most **50% of the possible points**.

Sample input/output (screen captures)
Question X, test case 1, input/output:



```
C:\Users\jacksoncd\Desktop>javac ChenchuttaJacksonRestaurantBill.java

C:\Users\jacksoncd\Desktop>java ChenchuttaJacksonRestaurantBill
Enter the charge for the meal: 35.00
Meal Charge: $35.0
Tax:         $2.3625000000000003
Tip:         $5.25
Total:       $42.6125

C:\Users\jacksoncd\Desktop>
```

Screen capture must be readable by the instructor, or 0 point will be given for the question.
Please note that you can use more than one screen captures for each test case.