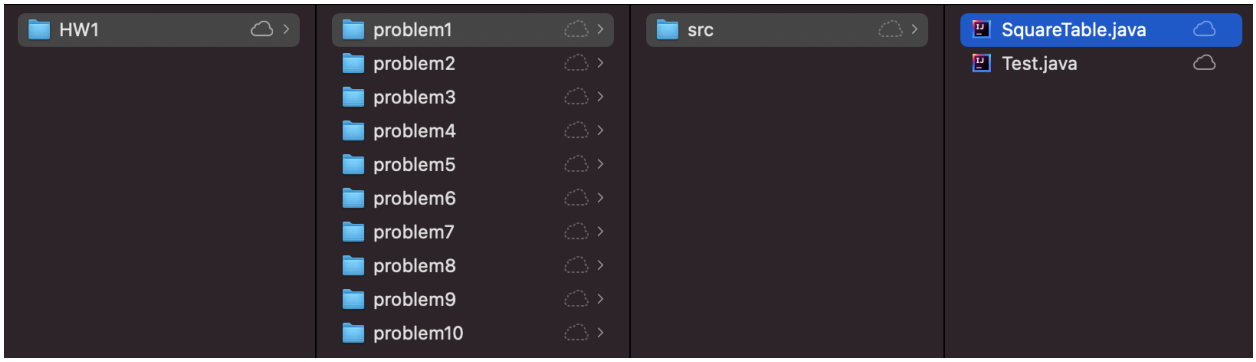
AY 2021 Assignment 1 [6 Marks]

Date / Time	17 September 2021 – 1 October 2021 23:59
Course	**[M1522.600] Computer Programming**
Instructor	Youngki Lee

- You can refer to the Internet or other materials to solve the assignment, but you ***SHOULD NOT*** discuss the question with anyone else and need to code **ALONE**.
- We will use the automated copy detector to check the possible plagiarism of the code between the students. The copy checker is reliable so that it is highly likely to mark a pair of code as the copy even though two students quickly discuss the idea without looking at each other's code. Of course, we will evaluate the similarity of a pair compared to the overall similarity for the entire class.
- We will do the manual inspection of the code. In case we doubt that the code may be written by someone else, we reserve the right to request an explanation about the code. We will ask detailed questions that cannot be answered if the code is not written by yourself.
- If one of the above cases happens, you will get 0 marks for the assignment and may get a further penalty. Please understand that we will apply these methods for the fairness of the assignment.
- Download and unzip “HW1+SKELETON.zip” file from the Autolab. The file contains the skeleton codes for 10 questions (in “/problemX” directory for the question #X).
- **Do not modify the overall directory structure**. Simply fill in the codes in appropriate files. It is okay to add new directories or files if needed.
- Do not use any external libraries. It is also not allowed to use JAVA Collection Framework.
- Assume all inputs are correct **(except for Question 3)**. This means that you do not have to handle wrong inputs, and the trailing spaces will be ignored.
- Contact TAs if you have any questions.

Submission Guidelines

1. For submission, compress the entire “HW1” directory in a single zip file.
2. The directory structure of your submitted file (after unzipping) should look like the following. When you extract the zip file, there must be the HW1/ directory.



3. Name the compressed file “20XX-XXXXX.zip” (your student ID).
4. Submit your code on the Autolab.
5. Double-check if your final zip file is properly submitted.
6. Note that you will get 0 marks for the wrong submission format.

Assignment 1 Overview.

1. The goal of this assignment is to implement simple Java programs.
2. All inputs and outputs are console inputs and outputs.
3. Question 1-8 are assigned with 0.5 mark each. Question 9 and 10 are assigned with 1 mark, respectively.
4. You can test your code with Autolab. See ‘Simple Autolab Guide.pdf’ on eTL.
5. Note that Autolab provides useful test cases but you may want to use additional test cases of your own. We will use a more comprehensive set of test cases for evaluation.

Question 1: Squares Table [0.5 Marks]

Objective: Write a program that prints out all squares that are less or equal to the input number N.

Note:

1. Outputs always start with **'1 times 1 = 1'**
2. Output format is **'k times k = k^2'**

Target Function: `public static void printSquareTable(int n)` (in SquareTable.java)
Input: int N (1 <= N <= 900)
Output: Square table corresponds to input number N.

Input	Output
49	1 times 1 = 1 2 times 2 = 4 3 times 3 = 9 4 times 4 = 16 5 times 5 = 25 6 times 6 = 36 7 times 7 = 49
23	1 times 1 = 1 2 times 2 = 4 3 times 3 = 9 4 times 4 = 16

Question 2: Sum of Fibonacci Numbers [0.5 Marks]

Objective: Write a program that prints the sum of N smallest Fibonacci numbers.

Description: Fibonacci Numbers, commonly denoted as F_n, form a sequence called the Fibonacci sequence. The sequence starts with the two numbers, 0 and 1, and each number in the sequence is the sum of its two preceding ones, that is, $F_0 = 0$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$ for n > 1. Print $F_0 \sim F_{N-1}$ given N as the i

Reference: https://en.wikipedia.org/wiki/Fibonacci_number

Target Function: `public static void printFibonacciNumbers(int n)` (in FibonacciNumbers.java)
Input: Integer N (1 <= N <= 40)
Output: The first N Fibonacci Numbers from the smallest to the largest and the sum of those N Fibonacci Numbers. If the sum has more than five digits, print only its last five digits (including leading zeros).

Input	Output
5	0 1 1 2 3 sum = 7
30	0 1 … 317811 514229 last five digits of sum = 46268

Question 3: Drawing Figure [0.5 Marks]

Objective: Write a program that prints the appropriate star(*) pattern for a given number.

Note:

1. If there are two or more stars in a row, there is a space between the adjacent stars.
2. There is no space before the first star and after the last star in the longest row.
3. The length of every row is the same as the longest row. This means that each row except for the longest one has spaces after the last star.

Target Function: `public static void drawFigure(int n)` (in DrawingFigure.java)
Input: Integer N (1 <= N <= 100)
Output: The corresponding star pattern as below.

Input	Output
1	<pre> * * * * *</pre>

2	<pre> *
	  * * *
	* * * * *
	  * * *
	    *</pre>

3	<pre> *
	    * * *
	  * * * * *
	* * * * * * *
	  * * * * *
	    * * *
	      *</pre>

4	<pre> *
	      * * *
	    * * * * *
	  * * * * * * *
	* * * * * * * * *
	  * * * * * * *
	    * * * * *
	      * * *
	        *</pre>

Question 4: Character Counter [0.5 Marks]

Objective: Write a program that

1. Takes an arbitrary string composed of alphabets.
2. Counts the number of alphabets that are in the input string.
3. Prints the alphabet and the corresponding count in an increasing character order. (Do NOT print out alphabets that have never appeared in the input string).
4. Prints uppercase first and then lowercase in the same line if there are both upper and lower case letters.

Target Function: `public static void countCharacter(String str)` (in CharacterCounter.java)
Input: String str (1 <= length <= 200)
Output: Character counts with an increasing character order. See an example below.

Input	Output

`bbbaAabaccHadAHbcbHHAdcbbeHaH`	`A: 3 times, a: 5 times`
	`b: 8 times`
	`c: 4 times`
	`d: 2 times`
	`e: 1 times`
	`H: 6 times`

Question 5: Number Counter [0.5 Marks]

Objective: Write a program that counts the frequency of each number (0~9) in the multiplication of three input numbers.

Description:

1. There are three input numbers. Each number is in range 1 <= N <= 999.
   (e.g., 214, 731, 200)
2. Multiply the input numbers.
   (e.g., 214 × 731 × 200 = 31,286,800)
3. Count the frequency of each number (0~9) from the multiplicated result.
   (e.g., 0: 2 times, 1: 1 times, 2: 1 times, 3: 1 times, 6: 1 times, 8: 2 times)
4. Print the counted frequency in the increasing order of the numbers. Do NOT print the numbers that have never been counted.
5. Print the length of the multiplication result.

Target Function: `public static void countNumbers(String str0, String str1, String str2)` (in NumberCounter.java)

Input: Three numbers per input String.

Output: Number counts with the increasing order of the numbers, followed by the length of the multiplication results.

Input	Output
927 281 350	0: 2 times 1: 2 times 4: 1 times 5: 1 times 7: 1 times 9: 1 times length: 8

Question 6: Prime Numbers [0.5 Marks]

Objective: Write a program that prints prime numbers between two input numbers, m and n, inclusive.

Description: A prime number (or a prime) is a natural number greater than 1 that is not a product of two smaller natural numbers. The smallest prime number is 2, and the second minority is 3. Print prime numbers in ascending order. If there is no prime number within the range, just print out nothing.

Reference: https://en.wikipedia.org/wiki/Prime_number

Target Function: `public static void printPrimeNumber(int m, int n)` (in PrimeNumbers.java)
Input: Natural number m and n (1 <= m < n <= 1000)
Output: Prime numbers in an ascending order.

Input	Output
5 29	5 7 11 13 17 19 23 29
200 202	

Question 7: Decreasing String [0.5 Marks]

Objective: Write a program that finds the longest substring where each alphabet in the substring is in a descending order.

Description: A substring is a contiguous sequence of characters within a string. Among substrings, there are decreasing substrings, where the alphabet on the left is always bigger than the one on the right. Find the length of the longest decreasing substring. If a substring contains multiple identical alphabets, the substring is not a decreasing substring.
(e.g., Input: “abdfedccbgdcba” ⇒ Longest decreasing substring: “gdcba” ⇒ Output: 5)
(Note: “fedccb” is NOT the longest decreasing substring. A substring with the length of 1 is also considered a decreasing substring.)

Target Function: `public static void printLongestDecreasingSubstringLength(String inputString)` (in DecreasingString.java)
Input: A string that consists of small letters. (1 <= input string length <= 200)
Output: Length of the longest decreasing substring.

Input	Output
`abcdaeca`	`3`
`abdfedccbgdcba`	`5`
`abab`	`2`
`z`	`1`

Question 8: Matrix Flip [0.5 Marks]

Objective: Write a program that flips (upside down and left to right) the input matrix. For example, flip the 3 x 3 input matrix on the left to generate the output matrix on the right.

Input Matrix	Output Matrix
AAA ABB BBB	BBB BBA AAA

Description: Implement a method that prints out the flipped matrix.

Target Function: `public static void printFlippedMatrix(char[][] matrix)` (in MatrixFlip.java)
Input: M, N, and a M by N size matrix composed of 2D char arrays. (1 <= M, N <= 100)
Output: Flipped matrix

Input	Output
3 3 AAA ABB BBB	BBB BBA AAA
4 5 BBBBC CBBBB BBBBB CCCCC	CCCCC BBBBB BBBBC CBBBB

Question 9: Fractional Number [1 Marks]

Objective: Write a program that calculates the input equation, which consists of four fundamental arithmetic operations.

Description:

1. The input equation will be composed of two numbers and an operator.
2. Two numbers and an operator are separated by spaces.
3. The input number can be two types.
    a. Fractional number: two natural numbers with '/' between them (ex. 2/3)
    b. Natural numbers (ex. 1, 2, 3, …) (1 <= N <= 999)
4. There are 4 operators: '+', '-', '*', '/'
5. All input numbers are positive, but the result can be negative.
6. The result should be an irreducible fraction. (You can use FractionalNumber.gcd())
7. If the result is negative, put '-' first. DO NOT make the denominator minus.
   (-2/3 ⇒ Good, 2/-3 ⇒ Bad)
8. If the result is a natural number, DO NOT use '1' as the numerator.
   (3 ⇒ Good, 3/1 ⇒ Bad)

Useful Resources:

- String: https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html

Target Function: `public static void printCalculationResult(String equation)` (in FractionalNumber.java)
Input: Equation to solve. Format: <Number><Space><Operator><Space><Number>
Output: Calculated result. The result should be an Irreducible Fraction.

Input	Output
`2/3 + 3/4`	`17/12`
`6/9 - 2`	`-4/3`
`1/2 + 1/2`	`1`

Question 10: Card Game Simulator [1 Marks]

Objective: Write a program that simulates a card game.

Description:

1. There are 20 unique cards with 10 numbers (0 ~ 9) and 2 shapes (O, X).
2. Player A and B start the game with 10 cards each. (Input contains this information)
    a. Input consists of 2 lines.
    b. The first line indicates Player A's card list, and the second line indicates Player B's card list. (See Example Input and Output)
3. The two players take turns playing cards.
4. The players choose the card to play with the following rules.
    a. The game starts with Player A. Player A uses the card with the largest number. If there are two cards (O, X) with the largest number, Player A will choose the card with shape X among the two cards.
    b. After that, if the next player has a card that has the same number as the previous player's, that card is used.
    c. If the next player does not have a card that has the same number as the previous player's but has the same shape card, the player will choose the same shape card with the largest number.
    d. If the next player does not have both of the same number cards and the same shape cards, the current player wins the game.
    e. If the two players used all of the cards, Player B wins.
5. Simulate the game by printing the sequence of the cards.
6. Print out the result message when the winner is decided.

Useful Resources:

- String: https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html
- PrintStream: https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/io/PrintStream.html
(Type of System.out is PrintStream)

Target Function: `private static void simulateCardGame(String inputA, String inputB)` (in CardGameSimulator.java)
Input: The first String ⇒ Player A's card list ,
The second String ⇒ Player B's card list.
Output: Simulated card game progress and result.

Input	Output

`00 3X 40 4X 7X 10 6X 50 60 9X`
`2X 20 30 90 5X 1X 80 70 8X 0X`
	`Player A: 9X`
	`Player B: 90`
	`Player A: 60`
	`Player B: 80`
	`Player A: 50`

	Player B: 5X Player A: 7X Player B: 7O Player A: 4O Player B: 3O Player A: 3X Player B: 8X Player A: 6X Player B: 2X Player A: 4X Player B: 1X Player A: 1O Player B: 2O Player A: 0O Player B: 0X Player B wins the game!
9X 3O 1X 1O 8O 3X 0O 0X 5O 6X 6O 2O 5X 9O 8X 2X 4X 4O 7O 7X	Player A: 9X Player B: 9O Player A: 8O Player B: 8X Player A: 6X Player B: 6O Player A: 5O Player B: 5X Player A: 3X Player B: 7X Player A: 1X Player B: 4X Player A: 0X Player B: 2X Player B wins the game!
0O 1O 2O 3O 4O 5O 6O 7O 8O 9O 0X 1X 2X 3X 4X 5X 6X 7X 8X 9X	Player A: 9O Player B: 9X Player B wins the game!
0O 1O 2O 3O 4O 5O 6O 7O 9X 9O 0X 1X 2X 3X 4X 5X 6X 7X 8X 8O	Player A: 9X Player B: 8X Player B wins the game!