CSE1141 Fall 2020

Programming Assignment #4

Due Date: 10/01/2021 – 23:59 (No Extension)

1. Write a program that calculates the invoice of each flat in the apartment building. There are N flats in a building and apartment heating is the central system. 30% of bill will be shared equally among the flats, the rest (70%) will be shared according to the consumption of each flat.

Sample Calculation:

- Suppose that an apartment building has 3 flats
- Consumption of each flat (m^3): 12.8, 23, 9.2
- Total Bill: 320.40 TL
  - 30% of bill: 96.12 TL
  - 70% of bill: 224.28 TL
- Then, the bill for each flat should be calculated as follows:
  - The bill for the Flat #1: 32.04 + 63.80 = 95.83 TL
  - The bill for the Flat #2: 32.04 + 114.63 = 146.67 TL
  - The bill for the Flat #3: 32.04 + 45.85 = 77.89 TL

Your program must have the following methods:

`public static void main(String[] args)`

- Main method will take the inputs from the user.
- Then, it will invoke **`calculateTheInvoice`** () and **`printBills`**() methods, respectively.

`public static double[] calculateTheInvoice (double[] flats, double totalBill)`

- You should calculate the bill of each flat based on the given sample calculation scenario.
- This method should take the following parameters:
  - A double array *flats* that indicates the consumption of each flat.
  - A double value *totalBill* that contains the total consumption of the whole apartment building.

- Then, the method should return a double array, which contains the calculated bill for each flat, to the main() method.

```
public static void printBills (double[] bills)
```

- You should print the values to the console display.

First input (*N*) is the number of flats in the apartment building. It is followed by *N* inputs for *N* flat consumption, and the last input is for the total bill.

Input Format:	N C1 C2 C3 .. Cn TotalBill
Sample Input:	3 12.8 23 9.2 320.40

Here,

- *N*: the number of flats
- *C1, C2, .. Cn*: Consumption of *each* flat
- *TotalBill*: The total Bill for the apartment building.

Example Run 1 (Input: `3 12.8 23 9.2 320.40`)

```
3 12.8 23 9.2 320.40
Flat #1: 95.83
Flat #2: 146.67
Flat #3: 77.89
```

Example Run 2 (Input: `15 12 14.2 15.87 21.4 19 13 8.1 11 15.14 16 23.14 27 5.98 7.18 17 812.90`)

```
15 12 14.2 15.87 21.4 19 13 8.1 11 15.14 16 23.14 27 5.98 7.18 17
812.90
Flat #1: 46.47
Flat #2: 52.0
Flat #3: 56.21
Flat #4: 70.13
Flat #5: 64.09
Flat #6: 48.98
Flat #7: 36.65
Flat #8: 43.95
Flat #9: 54.37
Flat #10: 56.54
Flat #11: 74.51
```

```
Flat #12: 84.23
Flat #13: 31.31
Flat #14: 34.33
Flat #15: 59.05
```

Example Run 3 (Input: 5 44.02 0 17 21.01 7.56 210.82)

```
5 44.02 0 17 21.01 7.56 210.82
1.Flat: 85.15
2.Flat: 12.64
3.Flat: 40.65
4.Flat: 47.25
5.Flat: 25.1
```

2. Write a program that will determine whether or not it is valid per the Luhn formula. The *Luhn algorithm* is a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers. The task is to check if a given string is valid.

   Validating a Number

   - Strings of length 1 or less are not valid.
   - Spaces are allowed in the input.
   - All other non-digit characters are disallowed.

   Example 1: valid credit card number

   Suppose that the following number is given as the input:

   4539 1488 0343 6467

   a. The first step of the Luhn algorithm is to double every second digit, starting from the right. We will be doubling

      4_3_ 1_8_ 0_4_ 6_6_

   b. If doubling the number results in a number greater than 9 then subtract 9 from the product. The results for our example:

      8569 2478 0383 3437

   c. Then, calculate the sum all of the digits:

      8+5+6+9+2+4+7+8+0+3+8+3+3+4+3+7 = 80

   d. If the sum is evenly divisible by 10, then the number is valid.
      80 is divisible by 10, and the quotient is 8 (even)

Then, the given number is valid!

Example 2: invalid credit card number

Suppose that the following number is given as the input:

8273 1232 7352 0569

- Double the second digits, starting from the right

  7253 2262 5312 0539

- Sum the digits

  7+2+5+3+2+2+6+2+5+3+1+2+0+5+3+9 = 57

- 57 is not evenly divisible by 10, so this number is not valid!

Your program must have the following methods:

`public static void main(String[] args)`

- Main method will take the input from user.
- Then it will invoke **`validateNumber`**() method and print the result.

`public static boolean validateNumber (String number)`

- Check if a given number (String) is valid or not.
- Then, the method should return the result (boolean) to the main() method.

Example Run 1

```
7634 78KS
Invalid Input !
```

Example Run 2

```
7789!
Invalid Input !
```

Example Run 3

```
4539 1488 0343 6467
DNumber:4_3_1_8_0_4_6_6_
LNumber:8569247803833437
Number is Valid
```

Example Run 4

```
8273 1232 7352 0569
DNumber:8_7_1_3_7_5_0_6_
LNumber:7253226253120539
Number is Invalid
```

Example Run 5

```
42 123 4598
DNumber:_2_2_4_9_
LNumber:441438598
Number is Invalid
```

Example Run 6

```
3 89 23 1234
DNumber:_8_2_1_3_
LNumber:379432264
Number is Valid
```

3. Write a program that takes an input letter and outputs it in a diamond shape. Given a letter, it prints a diamond starting with 'A', with the supplied letter at the widest point.

   Your program should satisfy the following requirements:

   - The first row contains one 'A'.
   - The last row contains one 'A'.
   - All rows, except the first and last, have exactly two identical letters.
   - The diamond is horizontally symmetric.
   - The diamond is vertically symmetric.
   - The diamond has a square shape (width equals height).
   - The letters form a diamond shape.
   - The top half has the letters in ascending order.
   - The bottom half has the letters in descending order.

Examples

In the following examples, spaces are indicated by - characters.

Diamond for letter 'A':

```
A
```

Diamond for letter 'C':

```
..A..
.B.B.
C...C
.B.B.
..A..
```

Diamond for letter 'E':

```
....A....
...B.B...
..C...C..
.D.....D.
E.......E
.D.....D.
..C...C..
...B.B...
....A....
```

Your program must have the following methods:

`public static void main(String[] args)`

- Main method will take the input letter from the user.
- Then, it will invoke **`constructDiamond`**() method.
- Lastly, it will invoke **`printDiamond`**() method.

`public static char[][] constructDiamond (char letter)`

- This method should take a char letter and construct the diamond shape for the given letter in a two dimensional char array.
- The size of your two dimensional array is determined based on the given letter.
- This method should return the two dimensional array to the main() method

`public static void printDiamond (char[][] diamond)`

- This method should take a two dimensional char array and print the content of it.

Example Run 1

```
Enter a Letter: 7
Invalid Input !
```

Example Run 2

```
Enter a Letter: *
Invalid Input !
```

Example Run 3

```
Enter a Letter: A
A
```

Example Run 4

```
Enter a Letter: C
..A..
.B.B.
C...C
.B.B.
..A..
```

Example Run 5

```
Enter a Letter: d
...A...
..B.B..
.C...C.
D.....D
.C...C.
..B.B..
...A...
```

Example Run 6

```
Enter a Letter: AC
Invalid Input !
```

Important Notes:

- Your program will be tested with an auto-grader. So it should take the input exactly the same in the example and it should print the output exactly the same in the example. Otherwise, your program may fail.
- You should exactly print the outputs with 2 digits after the decimal point.
- You shouldn't print any messages before input taking.
- Your program should execute correctly for different test cases.
- Please do not write any package name at the beginning of your code.

- Only parts selected from the selected questions will be graded. So if you send only one program, you might get a grade of 0 based on our evaluation.

Submission Instructions

Please zip and submit all your files using filename YourNumberHW4.zip (ex: 150713852HW4.zip) to Canvas system (under Assignments tab).

Your zip file should contain the followings:

1. Java source code for Problem 1 (Pro1_yourNumber.java)
2. Java class file for Problem 1 (Pro1_ yourNumber.class)
3. Java source code for Problem 2 (Pro2_ yourNumber.java)
4. Java class file for Problem 2 (Pro2_ yourNumber.class)
5. Java source code for Problem 3 (Pro3_ yourNumber.java)
6. Java class file for Problem 3 (Pro3_ yourNumber.class)

Notes:

1. Write a comment at the beginning of each program to explain the purpose of the program. **Write your name and student ID as a comment**. Include necessary comments to explain your actions.
2. Select meaningful names for your variables.
3. You are allowed to use the materials that you have learned in lectures & labs.
4. Do not use the things that you did not learn in the course.
5. Each student should submit his/her own homework. You can discuss with your peers about the homework but you are not allowed to exchange code or pseudocode. This also applies to material found on the web. Should some submitted homework assignments be identical or suspected to be identical, all involved parties will get a grade of **ZERO** from all homework. In case of any forms of cheating or copying, both giver and receiver are equally culpable and suffer equal penalties.
6. No late submission will be accepted.