Homework #5 – COP3330
Spring 2022

INSTRUCTION:

1. **Topics:** Chapters 9 , 10 ,and 11
2. **Total questions:** 8 question and each worth 15 points.
3. **Submission:** Use the same instruction as the previous HW. Put UML diagrams in a separate folder called UMLs.
4. **Due date:** March 27, 2022 (11:59 pm)
5. **Early submission:** March 27, 2022 (11:59 pm) (5 points)

Q1: (STOPWATCH) Design a class named StopWatch. The class contains:

- Private data fields startTime and endTime with getter methods.
- A no-arg constructor that initializes startTime with the current time.
- A method named start() that resets the startTime to the current time.
- A method named stop() that sets the endTime to the current time.
- A method named getElapsedTime() that returns the elapsed time for the stopwatch in milliseconds.

Draw the UML diagram for the class and then implement the class. Write a test program that measures the execution time of sorting 100,000 numbers using selection sort.

Q2: (Algebra: 2 * 2 linear equations) Design a class named LinearEquation for a 2 * 2 system of linear equations:

$$\begin{aligned} ax + by &= e \\ cx + dy &= f \end{aligned} \quad x = \frac{ed - bf}{ad - bc} \quad y = \frac{af - ec}{ad - bc}$$

The class contains:

- Private data fields a, b, c, d, e, and f.
- A constructor with the arguments for a, b, c, d, e, and f.
- Six getter methods for a, b, c, d, e, and f.

■ A method named isSolvable() that returns true if ad - bc is not 0.

■ Methods getX() and getY() that return the solution for the equation.

Draw the UML diagram for the class and then implement the class. Write a test program that prompts the user to enter a, b, c, d, e, and f and displays the result.

If ad - bc is 0, report that "The equation has no solution." Here is a sample run:

```
Enter a, b, c, d, e, f: 9.0 4.0 3.0 -5.0 -6.0 -21.0 Enter
x is -2.0 and y is 3.0
```

```
Enter a, b, c, d, e, f: 1.0 2.0 2.0 4.0 4.0 5.0 Enter
The equation has no solution
```

Q3:(The Location class) Design a class named Location for locating a maximal value and its location in a two-dimensional array. The class contains public data fields row, column, and maxValue that store the maximal value and its indices in a two-dimensional array with row and column as int types and maxValue as a double type.
Write the following method that returns the location of the largest element in a two-dimensional array:

```
public static Location locateLargest(double[][] a)
```

The return value is an instance of Location. Write a test program that prompts the user to enter a two-dimensional array and displays the location of the largest element in the array. Here is a sample run:

```
Enter the number of rows and columns in the array:  3 4 Enter
Enter the array:
23.5 35 2 10 Enter
4.5 3 45 3.5 Enter
35 44 5.5 9.6 Enter
The location of the largest element is 45 at (1, 2)
```

Q4: (Game: ATM machine) Use the Account class created in Programming Exercise 9.7 to simulate an ATM machine. Create ten accounts in an array with id 0, 1, . . . , 9, and initial balance $100. The system prompts the user to enter an id. If the id is entered incorrectly, ask the user to enter a correct id. Once an id is accepted, the main menu is displayed as shown in the sample run. You can enter a choice 1 for viewing the current balance, 2 for withdrawing

money, 3 for depositing money, and 4 for exiting the main menu. Once you exit, the system will prompt for an id again. Thus, once the system starts, it will not stop.

```
Enter an id: 4 Enter

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 Enter
The balance is 100.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 2 Enter
Enter an amount to withdraw: 3 Enter

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 Enter
The balance is 97.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 3 Enter
Enter an amount to deposit: 10 Enter

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 Enter
The balance is 107.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 4 Enter

Enter an id:
```

Q5: (The MyDate class) Design a class named MyDate. The class contains:

■ The data fields year, month, and day that represent a date. month is 0-based, i.e., 0 is for January.
■ A no-arg constructor that creates a MyDate object for the current date.
■ A constructor that constructs a MyDate object with a specified elapsed time since midnight, January 1, 1970, in milliseconds.
■ A constructor that constructs a MyDate object with the specified year, month, and day.
■ Three getter methods for the data fields year, month, and day, respectively.
■ A method named setDate(long elapsedTime) that sets a new date for the object using the elapsed time.

Draw the UML diagram for the class and then implement the class. Write a test program that creates two MyDate objects (using new MyDate() and new MyDate(34355555133101L)) and displays their year, month, and day.
(Hint: The first two constructors will extract the year, month, and day from the elapsed time. For example, if the elapsed time is 561555550000 milliseconds, the year is 1987, the month is 9, and the day is 18. You may use the GregorianCalendar class discussed in Programming Exercise 9.5 (in the text) to simplify coding.)

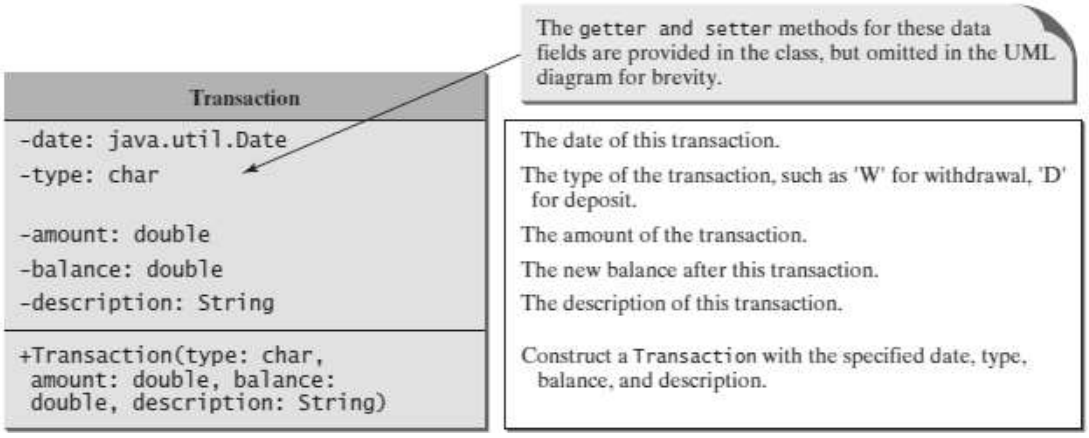
Q6: (Implement the StringBuilder class) The StringBuilder class is provided in the Java library. Provide your own implementation for the following methods (name the new class MyStringBuilder2):

■public MyStringBuilder2();
■public MyStringBuilder2(char[] chars);
■public MyStringBuilder2(String s);
■public MyStringBuilder2 insert(int offset, MyStringBuilder2 s);
■public MyStringBuilder2 reverse();
■public MyStringBuilder2 substring(int begin);
■public MyStringBuilder2 toUpperCase();

Q7: (The Person, Student, Employee, Faculty, and Staff classes) Design a class named Person and its two subclasses named Student and Employee.
Make Faculty and Staff subclasses of Employee. A person has a name, address, phone number, and email address. A student has a class status (freshman, sophomore, junior, or senior). Define the status as a constant. An employee has an office, salary, and date hired. Use the MyDate class defined in Programming Exercise 10.14 (in the text) to create an object for date hired. A faculty member has office hours and a rank. A staff member has a title. Override the toString method in each class to display the class name and the person's name.
Draw the UML diagram for the classes and implement them. Write a test program that creates a Person, Student, Employee, Faculty, and Staff, and invokes their toString() methods.

Q8: (New Account class) An Account class was specified in Programming Exercise 9.7. Design a new Account class as follows:

■ Add a new data field name of the String type to store the name of the customer.
■ Add a new constructor that constructs an account with the specified name, id, and balance.
■ Add a new data field named transactions whose type is ArrayList that stores the transaction for the accounts. Each transaction is an instance of the Transaction class. The Transaction class is defined as shown in Figure below.



■ Modify the withdraw and deposit methods to add a transaction to the transactions array list.
■ All other properties and methods are the same as in Programming Exercise 9.7. Write a test program that creates an Account with annual interest rate 1.5%, balance 1000, id 1122, and name George. Deposit $30, $40, and $50 to the account and withdraw $5, $4, and $2 from the account. Print an account summary that shows account holder name, interest rate, balance, and all transactions.