Exercise1

1- Create a class called FullTimeEmployee that includes three instance variables—a first name (type String), a last name (type String), and hour rate (type double). Provide a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the hour rate is not positive, do not set its value. Write a method in the class to calculate the yearly salary and name it calcYearlySalary. Write a test app named FullTimeEmployeeTest that demonstrates class FullTimeEmployee’s capabilities. Create two

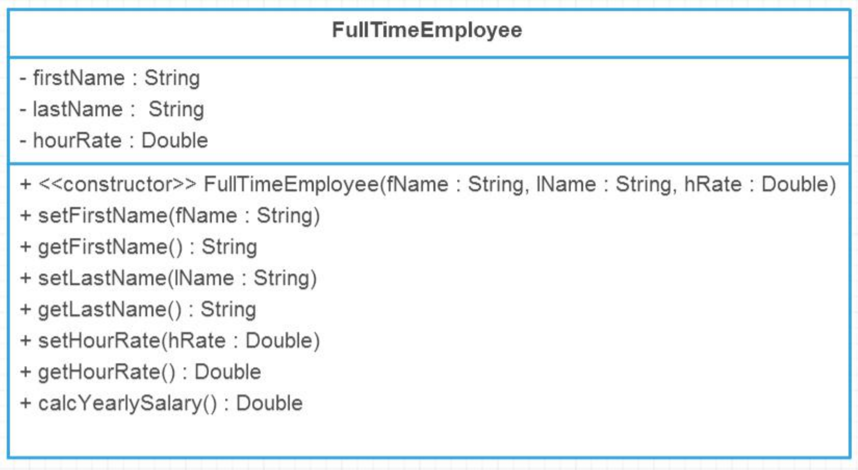
FullTimeEmployee objects and display each object’s yearly salary.

Hint 1: Full time employee works 40 hours per week.

Hint 2: Number of weeks in a year is 52.

Hint 3: Yearly salary is given by the formula: Hour rate * number of hours per week * number of weeks in a year.

The UML diagram for the class FullTimeEmployee is:



FullTimeEmployee

- firstName : String
- lastName : String
- hourRate : Double

+ <<constructor>> FullTimeEmployee(fName : String, lName : String, hRate : Double)
+ setFirstName(fName : String)
+ getFirstName() : String
+ setLastName(lName : String)
+ getLastName() : String
+ setHourRate(hRate : Double)
+ getHourRate() : Double
+ calcYearlySalary() : Double

2- Create a class called Invoice that include four instance variables—an item number (type String), an item description (type String), a quantity of the item being purchased (type int) and a price per item (double). Your class should have a constructor that initializes the four instance variables. Provide a set and a get method for each instance variable. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0.0. In addition, provide a method named getInvoiceAmount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a double value.

Write a test app named InvoiceTest that demonstrates class Invoice’s capabilities.

Draw UML class diagram for the Invoice class.

Exercise2

1. Without using subroutines, sums 5 different numbers, which previously stored in 5 different registers, in the memory. Save your program to a new file, assemble it, and run it.

   Tips: Start with a register with a pre-loaded value, your choice. Then save 5 difference values, your choice again, into 5 different registers. Then do the additions while saving the result back to the initial register. **DON'T use subroutines.**

2. Use 5 different subroutines to subtract 5 different numbers, which previously stored in 5 different registers, in the memory. The initial value must be negative, so your final result must be negative too after the 5 subtractions. Save your program to a new file, assemble it, and run it. **Please use subroutines.**

3. Conduct OR operation with 2 registers. You can use subroutines or not, it is up to you. **Tips: De Morgan**

4. Write two loops for this question. The first loop runs 20 times and keeps adding one to a value stored in a register. Once this loop is finished, the program will jump to a second loop that runs 20 times again and keeps subtracting one from the same value stored in the same register.
   Tips: Use a counter

5. Write a FSM chart for all the questions above. You should have 4 FSM charts. **You can do them on paper- handwritten.**

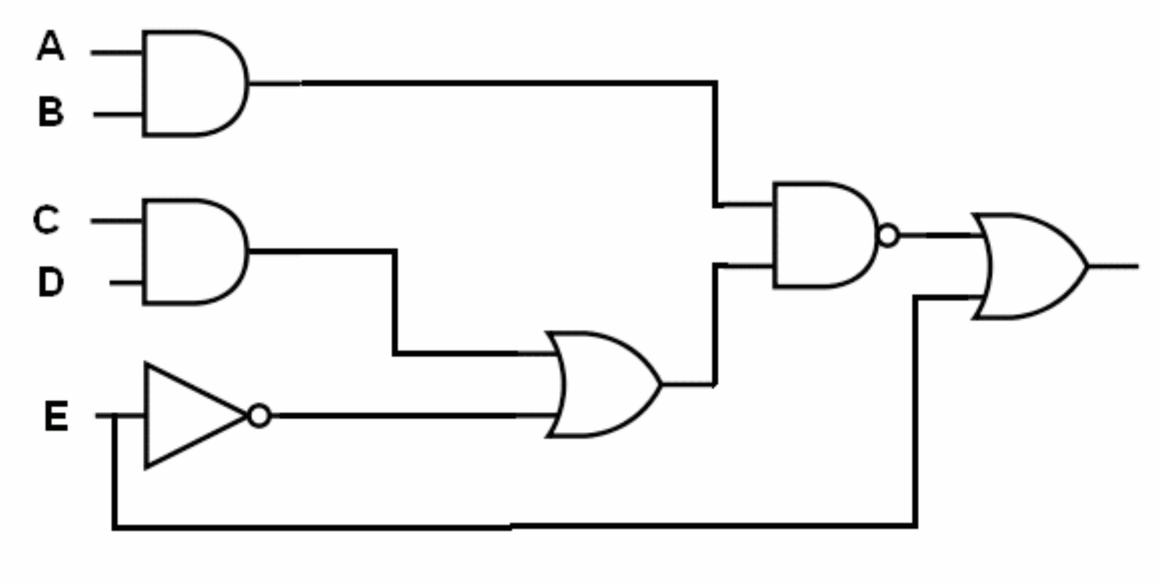
Remember ***Internal Commentary*** and ***Good Spacing*!!!**

Exercise3

1. Write a Java application that saves 10 integers from user's input into an 1D array and after that it could find the largest of the 10 integers.

2. Write a Java application that takes 20 sets of 2 characters from the user. These two characters should be saved into a 2D array. And then display the characters from the 2D array line by line to terminal using the printf or println function.

3. Create a program that could work out the following Boolean Expression:



4. Write an UML for all three programs.