Lab Exercises No.1
Accelerated Introduction to Computer Science Fall 2023
Course CS 201 02

Name: Harlee Ramos **Due Date:** 09/01/2023

Exercise 1.1. Type in the below program code, the black text., and then submit the test case. Java code is case-sensitive, so be careful about the upper and lower cases.

```
public class Exercise1 {
    public static void main(String[] args) {
        System.out.printf("Hello World");

    }
}
```

No code is needed for the rest of this lab Remember the process from class, showcase your inputs, process, and outputs in each problem.

Problem Solving with Pseudocode

Exercise 1.2. A teenager gets retained by a neighborhood association to distribute fliers, collect dues, and do miscellaneous chores. Just to make sure they are around when needed, they get 40 dollars a month (they don't have to work for that). Plus, they get $11.25 per hour for any time they actually work in a month. Given how long they work in a month, calculate how much they earn.

```
BEGIN

  DECLARATION OF VARIABLES
  VARIABLES WITH SET VALUE:           MONTHLY PAYMENT              double monPay = 40.00;
                                      BONUS PAYMENT PER HOUR       double hourPay = 11.25;
  VARIABLES THAT NEED USER INPUT:     WORKED HOURS                 double workedHrs;
  VARIABLES FOR STORAGE RESULT:       MONTHLY EARNING              double monEarning;

  PRINT ("Introduce the quantity of worked hours: ") ;

  USER INPUT THE VALUE FOR workedHrs;

  PROCEDURE monEarning= monPay + hourPay*workedHrs;

  PRINT ("The monthly earnings are: $",  monEarning) ;

END
```

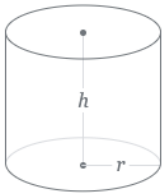
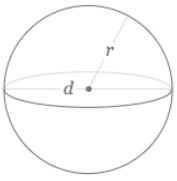
Exercise 1.3. Consider the 3-dimensional barbell shown. a) Find the volume of the figure if the radius of each sphere is given, and the length of the bar connecting them is given, and the diameter of the bar is given (all in the same units) b) Find the surface area of the figure.

Source: Google Images

IF WE CONSIDER THAT BOTH SPHERES HAVE THE SAME WEIGHT AND RADIUS THEN:

```
BEGIN
  DECLARATION OF VARIABLES:
    VARIABLES THAT NEED USER INPUT:    RADIUS OF SPHERE            double radSphere;
                                       LENGTH OF THE BAR           double legBar;
                                       DIAMATER OF THE BAR         double diaBar;
    VARIABLES FOR STORAGE VALUE:       VOLUME OF SPHERES           double volSphere;
                                       RADIUS OF THE BAR           double radiBar;
                                       VOLUME OF BAR               double volBar;
                                       SURFACE AREA OF SPHERES     double surfAreaS;
                                       SURFACE AREA OF BAR         double surfAreaB;
                                       VOLUMEN OF THE FIGURE       double volFigure;
                                       SURFACE AREA OF THE FIGURE  double surfAreaFigure;

  PRINT NOTE FOR THE USER ("Input a numeric value in the same unit of distance in the following cases");

  PRINT                              ("Introduce the radius of the sphere") ;
  USER INPUT THE VALUE FOR           radSphere;

  PRINT                              ("Introduce the length of the bar") ;
  USER INPUT THE VALUE FOR           legBar;

  PRINT                              Introduce the diameter of the bar") ;
  USER INPUT THE VALUE FOR           diaBar;

  PROCEDURE CALCULATION OF BAR'S RADIUS          radiBar = diaBar/2;
  CALCULATION OF SPHERE'S VOLUME          volSphere = (4/3)*pi*(radSphere)^3;

  CALCULATION OF BAR'S VOLUME        volBar = pi *(radiBar)^2  * legBar;

  CALCULATION OF SURFACE AREA OF SPHERE      surfAreaS = 4*pi * 2*(radSphere)^2 ;

  CALCULATION OF SURFACE AREA OF BAR surfAreaB = 2 * pi* radiBar*  legBar + 2* pi *(radiBar)^2;

  CALCULATION OF VOLUME OF FIGURE    volFigure = 2 * volSphere + volBar;

  CALCULATION OF SURF AREA OF FIGURE surfAreaFigure = 2 * surfAreaS + surfAreaB;

  PRINT ("The volume of the figure is "+ volFigure) ;
  PRINT ("The surface area of the figure is "+ surfAreaFigure) ;

END
```

Used formulas to develop the pseudocode

CYLINDER		SPHERE	
	SURFACE AREAN $A = 2\pi rh + 2\pi r^2$ VOLUME $V = \pi r^2 h$		SURFACE AREA $A = 4\pi r^2$ VOLUME $V = \frac{4}{3}\pi r^3$

Source: Google

Exercise 1.4. Most stop watches allow you to display the time elapsed as a number of seconds, or as hours : minutes: seconds. So, 5437 seconds or 1 hour:30 minutes:37 seconds Given an integer number of seconds, calculate the equivalent integer number of hours, integer number of minutes, and integer number of seconds.

```
BEGIN
       DECLARATION OF VARIABLES:
          VARIABLES THAT NEED USER INPUT:    INTEGER NUMBER OF SECONDS            int inSeconds;
          VARIABLES FOR STORAGE VALUE:              EXCHANGE TO HOURS             int excHours;
                                                    EXCHANGE TO MINUTES           int excMinut;
                                                    EXCHANGE TO SECONDS           int excSeconds

       PRINT                                        ("Introduce the integer number in seconds") ;

       USER INPUT THE VALUE FOR                     inSeconds;

       PROCEDURE      excHours = inSeconds/3600;

                      excMinut = (inSeconds-(3600 * excHours))/60;

                      excSeconds = (inSeconds-(3600 * excHours)-(excMinut * 60)) ;

       PRINT   ("The equivalent in hours is: "+ excHours) ;

       PRINT   ("The equivalent in minutes is: "+ excMinut) ;

       PRINT   ("The equivalent is seconds is: "+ excSeconds) ;

END
```

Exercise 1.5. A person suffering from type II diabetes injects insulin based on measurements of their blood sugar level. If the blood sugar level is less than 115, they don't need to inject any insulin at all. For a value of 115, they inject 1 unit of insulin. For every additional increase of 20 in her blood sugar level, they get one additional unit of insulin. (Thus, for a blood-sugar level of 134, they get 1 unit; for a blood-sugar level of 135, they get 2 units.) Create a formula for calculating the insulin injections and output a table that shows the number of units of insulin injected for a user input range of blood sugar values from 115 upwards.

```
BEGIN
  DECLARATION OF VARIABLES:
    VARIABLES THAT NEED USER INPUT:    USER BLOOD SUGAR LEVEL              double user_bloodsug;

    VARIABLES FOR STORAGE VALUE:       NUMBER OF INSULIN INJECTIONS        double num_injection;


  PRINT  ("Introduce the patient blood level of sugar");

  USER INPUT THE VALUE FOR     user_bloodsug;

  PROCEDURE
         IF user_bloodsug > 134;
                CALCULATE num_injection = (user_bloodsug - 134)/20;
                THEN ROUND UP with Math.ceil (num_injection);
                PRINT ("The patient need "+ num_injection + "units of insulin") ;

         ELSE IF 115 <= user_bloodsug <=134;
                       CALCULATE     num_injection = 1;
                       PRINT ("The patient need "+ num_injection + "units of insulin") ;

                              ELSE user_bloodsug < 115;
                                     CALCULATE     num_injection = 0;
                                     PRINT ("The patient does not need an insulin shoot") ;
END
```

Exercise 1.6. You would like to calculate your GPA (GPA = points Earned/credit Hours Completed) after this term is over. You already know how many credits you have completed prior to this term and your GPA prior to this term. And you are estimating that you will get all As (each worth 4) this term for 15 more hours (4*15=60 points earned this term). What will your new GPA be?

```
BEGIN
 DECLARATION OF VARIABLES:
 VARIABLES THAT NEED USER INPUT:     POINTS EARNED LAST TERM                      double earnPoints1;
                                     CREDIT HOURS COMPLETED IN THE LAST TERM      double credHcomple1;
                                     NEW TERM POINTS ESTIMATION                   double earnPoints2;
                                     NEW TERM CREDIT HOURS                        double credHcomple2;
 VARIABLES FOR STORAGE VALUE:        ESTIMATION OF GPA                            double gpa_estimat;

 PRINT  ("Input the points earned in last the term");
 USER INPUT THE VALUE FOR    earnPoints1;

 PRINT  ("Input the CREDIT HOURS COMPLETED IN THE LAST TERM");
 USER INPUT THE VALUE FOR    credHcomple1;

 PRINT  ("Input your estimated points for this term");
 USER INPUT THE VALUE FOR    earnPoints2;

 PRINT  ("Input your credit hours for this term");
 USER INPUT THE VALUE FOR    credHcomple2;

 PROCEDURE    gpa_estimat =  (earnPoints1 + earnPoints2)/(credHcomple1 + credHcomple2);
 PRINT ("Your estimated GPA will be: ");


END
```

Exercise 1.7. Most banks provide change counting machines for their customers. Given the number of pennies, nickels, dimes and quarters, calculate the total value of all this change.

```
BEGIN
       DECLARATION OF VARIABLES:
       VARIABLES THAT NEED USER INPUT:      int val_pennies;
                                            int val_nickels;
                                            int val_dimes ;
                                            int val_quarters ;
       VARIABLES FOR STORAGE VALUE:         double val_change;

PRINT    ("Input the integer number of pennies");
USER INPUT THE VALUE FOR     val_pennies;

PRINT    ("Input the integer number of nickels");
USER INPUT THE VALUE FOR     val_nickels;

PRINT    ("Input the integer number of dimes");
USER INPUT THE VALUE FOR     val_dimes;

PRINT    ("Input the integer number of quarters");
USER INPUT THE VALUE FOR     val_quarters;


PROCEDURE  val_change = (val_pennies*0.01)+(val_nickels*0.05)+(val_dimes*0.10)+(val_quarters*0.25);

PRINT   ("TOTAL CHANGE: $"+ val_change);

END
```

Exercise 1.8. Clothing stores sometimes have sales during the year to attract customers and clear inventory. Calculate the final sale price of an item by deducting the sale percentage from the original price and then adding the tax percentage. Testing Problems

```
BEGIN
 DECLARATION OF VARIABLES:
  VARIABLES THAT NEED USER INPUT:    ORIGINAL PRICE OF THE PRODUCT        double ori_price;
                                     SALE PERCENTAJE FOR THE PRODUCT      double sale_percen;
                                     PERCENTAJE OF TAXES                  double tax_percen;
  VARIABLES FOR STORAGE VALUE:       GROSS SALE PRICE WITHOUT TAXES       double gross_sprice;
                                     FINAL SALE PRICE OF PRODUCT          double sale_price;
  PRINT        ("Input the original price of the product ");
  USER INPUT THE VALUE FOR   ori_price;

  PRINT        ("Input the sale percentage in decimals");
  USER INPUT THE VALUE FOR   sale_percen;

  PRINT        ("Input the taxes percentage in decimals");
  USER INPUT THE VALUE FOR   tax_percen;

  PROCEDURE   gross_sprice = ori_price* sale_percen;
               sale_price = gross_sprice *(1+ tax_percen);

 PRINT ("SUBTOTAL: $ "+ gross_sprice);
 PRINT ("TOTAL: $ "+ sale_price) ;
END
```

Testing Problems

Exercise 1.9. The bank wants to be sure you can afford to pay them back before they give you a mortgage. One way they consider your ability to repay is by making sure your total debt doesn't exceed a certain percentage of your income, usually 36-42%. This percentage is called the debt ratio. Given your monthly income (a real number) and other monthly debt payment(a real number), you can use the following formulas to determine the lower and upper limits on your monthly mortgage payments.

- Lower limit = (36% of your income) minus your other monthly debt
- Upper limit = (42% of your income) minus your other monthly debt

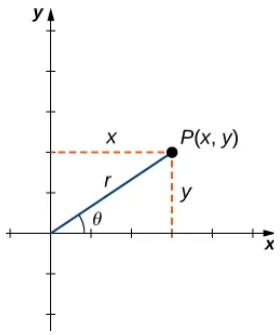
PSEUDOCODE

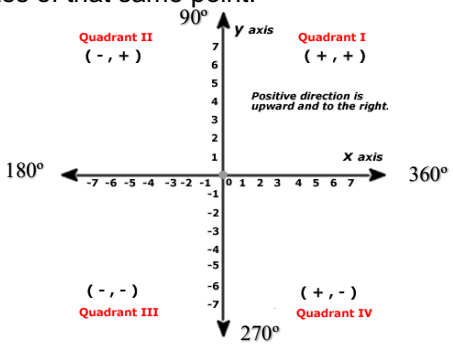
```
BEGIN

  DECLARATION OF VARIABLES:
  VARIABLES THAT NEED USER INPUT:     MONTHLY INCOME                          double montInco;
                                       OTHER MONTHLY DEBT PAYMENT             double montDebt;
  VARIABLES WITH SET VALUE:RECOMMENDED DEBT RATIO FOR LOWER LIMIT             double ratioDebt1= 0.36;
                RECOMMENDED DEBT RATIO       FOR UPPER LIMIT                  double ratioDebt2= 0.42;
  VARIABLES FOR STORAGE VALUE: LOWER LIMIT OF MONTHLY MORTGAGE PAYMENTS       double lowerLi;
                       UPPER LIMIT OF MONTHLY MORTGAGE PAYMENTS               double upperLi;

  PRINT  ("Input your monthly income");
  USER INPUT THE VALUE FOR     montInco;

  PRINT  ("Input the sum of other monthly payments that you may have");
  USER INPUT THE VALUE FOR     montDebt;


  PROCEDURE     lowerLi = ratioDebt1* montInco  - montDebt;
                upperLi = ratioDebt2* montInco  - montDebt;

        PRINT   ("YOUR LOWER LIMIT OF MONTHLY MORTGAGE PAYMENTS IS: $"+ lowerLi);
        PRINT   ("YOUR UPPER LIMIT OF MONTHLY MORTGAGE PAYMENTS IS: $"+ upperLi) ;
END
```

TEST

```
montDebt = 375.55
montInco = 3400.00

ratioDebt1= 0.36
ratioDebt2= 0.42

lowerLi = 0.36* 3400.00  - 375.55  = 848.45
upperLi = 0.42* 3400.00  - 375.55  = 1052.45

YOUR LOWER LIMIT OF MONTHLY MORTGAGE PAYMENTS IS: $848.45
YOUR UPPER LIMIT OF MONTHLY MORTGAGE PAYMENTS IS: $1052.45
```

Exercise 1.10. A point in the two-dimensional plane can be represented by its cartesian coordinates x and y or by its polar coordinates theta (angle) and r (radius). Write a program given a radius and angle (in degrees) of a point to calculate and display the x and y cartesian coordinates of that same point.





Source: Google Images

PSEUDOCODE

```
BEGIN
  DECLARATION OF VARIABLES:
  VARIABLES THAT NEED USER INPUT:          POLAR COORDINATE ANGLE      double  angleValue;
                                           POLAR COORDINATE RADIUS     double radiusValue;
  VARIABLES FOR STORAGE VALUE:             CARTESIAN COORDENATE x      double  coorXval;
                                           CARTESIAN COORDENATE y      double  coorYval;
  PRINT  ("Input the polar coordinate angle: ");
  USER INPUT THE VALUE FOR     angleValue;

  PRINT  ("Input the polar coordinate radius: ");
  USER INPUT THE VALUE FOR     radiusValue;

  PROCEDURE     coorXval = radiusValue * cos (angleValue);
                coorYval = radiusValue * sin (angleValue);
  PRINT("The cartesian coordinate of the inputted values is: ("+ coorXval + (",")+coorYval + ")");

        IF 0 < angleValue <= 90 ;
          PRINT("It is displayed in Quadrant I");

          ELSE IF 90 < angleValue <= 180 ;
            PRINT("It is displayed in Quadrant II");

          ELSE IF 180 < angleValue <= 270 ;
            PRINT("It is displayed in Quadrant III");

          ELSE 270 < angleValue <= 360 ;
            PRINT("It is displayed in Quadrant IV");
END
```

TEST

CONDITION 1 0 < angleValue <= 90	CONDITION 2 90 < angleValue <= 180
angleValue = 30 radiusValue = 5 coorXval = 5 * cos (30) = 4.3 coorYval = 5 * sin (30) = 2.5 The cartesian coordinate of the inputted values is:(4.3, 2.5) It is displayed in Quadrant I	angleValue = 145 radiusValue = 6 coorXval = 6 * cos (145) = -4.9 coorYval = 6 * sin (145) = 3.4 The cartesian coordinate of the inputted values is:(-4.9, 3.4) It is displayed in Quadrant II
CONDITION 3 180 < angleValue <= 270	**CONDITION 4 270 < angleValue <= 360**
angleValue = 200 radiusValue = 2 coorXval = 2 * cos (200) = -1.9 coorYval = 2 * sin (200) = -0.7 The cartesian coordinate of the inputted values is:(,)	angleValue = 350 radiusValue = 10 coorXval = 10 * cos (350) = 9.8 coorYval = 10 * sin (350) = -1.7 The cartesian coordinate of the inputted values is:(9.8, -1.7)

It is displayed in Quadrant III	It is displayed in Quadrant IV

Exercise 1.11. The rate your high-rise building is charged for waste and recycling collection is based on the total weight of the waste (which must be dumped in a landfill) and the total weight of the items that can be recycled. If the recycle items weight is greater than or equal to the waste weight, you are charged $10 per 100 pounds of waste. Otherwise you are charged $10 per 100 pounds of waste plus recycle weight.

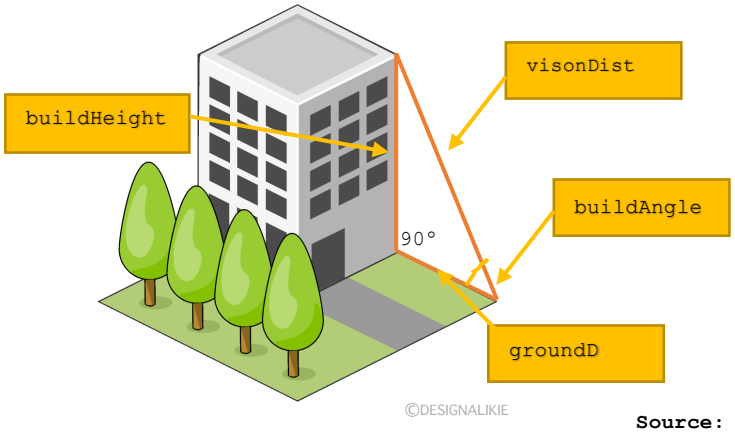
PSEUDOCODE

```
BEGIN
  DECLARATION OF VARIABLES:
  VARIABLES THAT NEED USER INPUT:     NON-RECYCLABLE WASTE WEIGHT          double wasteWei;
                                        RECYCLABLE WASTE WEIGHT            double recyWei;
   VARIABLES WITH SET VALUE:          CHARGED AMOUNT                       double chargAmo = 10;
   VARIABLES FOR STORAGE VALUE:              TOTAL CHARGE                  double tot_charge;
                                                                           double dif_waste;
                                                                           double dif_recy;
  PRINT  ("Input the weight in pounds of the non-recyclable waste");
  USER INPUT THE VALUE FOR    wasteWei;

  PRINT  ("Input the weight in pounds of the recyclable waste");
  USER INPUT THE VALUE FOR    recyWei;

  PROCEDURE     dif_waste =  wasteWei/100;
                dif_recy =   recyWei/100;

                   IF wasteWei <= recyWei;
                          CALCULATE tot_charge = chargAmo* dif_waste;
                          PRINT       ("Your waste charge is: $"+ tot_charge) ;

                   ELSE wasteWei > recyWei;
                          CALCULATE tot_charge = chargAmo* (dif_waste + dif_recy) ;
                          PRINT ("Your waste charge is $:"+ tot_charge) ;
END
```

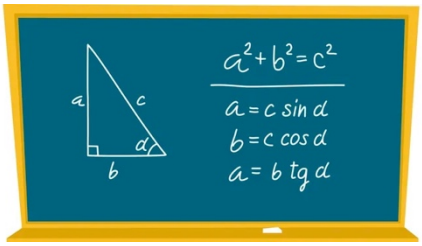
TEST

Case 1 wasteWei <= recyWei	Case 2 wasteWei > recyWei
wasteWei = 312.04	wasteWei = 359.20
recyWei = 359.20	recyWei = 312.04
chargAmo = 10.00	chargAmo = 10.00
dif_waste = 250.00/100 = 3.59	dif_waste = 359.20/100 = 3.59
dif_recy = 312.04/100 = 3.12	dif_recy = 312.04/100 = 3.12
tot_charge = 10.00 * 3.59 = 35.90	tot_charge = 10.00* (3.59 + 3.12) = 67.10
Your charge is: $ 35.90	Your charge is: $ 67.10

Exercise 1.12. You can use trigonometry to find the height of a building as shown. Suppose you can measure the angle theta (plus or minus 3 degrees) between the line of sight to the top of the building and the ground, and you can measure the distance d to the building. Calculate an upper and lower estimate for the height of the building.



Pythagoras Theorem Formulas



Source: Google Images

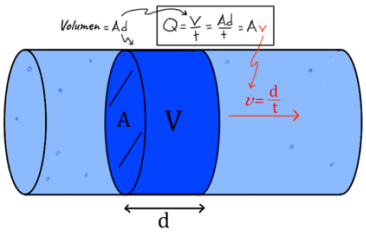
PSEUDOCODE

```
BEGIN
        DECLARATION OF VARIABLES:
        VARIABLES THAT NEED USER INPUT: GROUND DISTANCE TO THE BUILDING              double groundD;
                                        ANGLE BETWEEN THE TOP OF THE BUILDING AND GROUND double buildAngle;
        VARIABLES FOR STORAGE VALUE: ESTIMATED HEIGHT OF THE BUILDING 1             double buildHeight1;
                                     ESTIMATED HEIGHT OF THE BUILDING 2             double buildHeight2;
                                AIR DISTANCE BETWEEN THE TOP BUILDING AND GROUND double visionDist1;
                                AIR DISTANCE BETWEEN THE TOP BUILDING AND GROUND double visionDist2;

        PRINT    ("Input the ground distance to the building ");
        USER INPUT THE VALUE FOR      groundD;

        PRINT    ("Input the angle in grade value between the top of the building and ground ");
        USER INPUT THE VALUE FOR      buildAngle;

        PROCEDURE
                              visionDist1 = groundD / cos (buildAngle + 3°) ;
                              visionDist2 = groundD / cos (buildAngle - 3°) ;

                              buildHeight1 = visonDist1 * sin(buildAngle + 3°) ;
                              buildHeight2 = visonDist2 * sin(buildAngle - 3°) ;


        PRINT ("The estimated height plus three degrees for the building is: "+ buildHeight1 +" units") ;
        PRINT ("The estimated height minus three degrees for the building is: "+ buildHeight2+" units") ;
END
```

TEST

```
groundD = 400
buildAngle = 34

visionDist1 = 400 / cos (34 + 3) = 500.85
visionDist2 = 400 / cos (34 - 3) = 466.65

buildHeight1 = 500.85 * sin (34 + 3) = 301.41
buildHeight2 = 500.85 * sin (34 - 3) = 247.96

The estimated height plus three degrees for the building is: 301.41 units
The estimated height minus three degrees for the building is: 247.96 units
```

Exercise 1.13. Given the constant flow rate of a faucet into the sink in volume/sec, the volume of the sink, and the constant drain rate of the drain in volume/sec, determine if the faucet is left running when (if ever) the sink will overflow. Output when it will overflow in seconds, or a message stating it will not overflow.



Source: Google Images

SUPPOSITIONS

Considering that the flow Q in the faucet's tube and the drain's tube is constant, then

SINK VOLUME = FAUCET VOLUMEN – DRAIN VOLUME

SINK VOLUME =ΔVOLUME

SINK VOLUME = ΔQ* ΔT

We can say that Δt will represent the time of draining the sink, then

SINK VOLUME = (FAUCET FLOW RATE – DRAIN RATE)*DRAINING TIME

$$DRAINING\ TIME = \frac{SINK\ VOLUME}{FAUCET\ FLOW\ RATE\ -\ DRAIN\ FLOW\ RATE}$$

PSEUDOCODE

```
BEGIN
DECLARATION OF VARIABLES:
        VARIABLES THAT NEED USER INPUT: Q CONSTANT FLOW RATE OF A FAUCET    double faucetQ;
                                        Q CONSTANT DRAIN RATE OF THE DRAIN  double drainQ;
                                        VOLUME OF THE SINK                  double sinkVol;
VARIABLES FOR STORAGE VALUE:            DRAINING TIME                       double drainingTime;

PRINT    ("Input the Q constant of the flow rate of a faucet into the sink: ");
USER INPUT THE VALUE FOR      faucetQ;

PRINT    ("Input the Q constant of the flow drain rate of the drain: ");
USER INPUT THE VALUE FOR      drainQ;

PRINT    ("Input the volume of the sink: ");
USER INPUT THE VALUE FOR      sinkVol;

PROCEDURE      drainingTime = sinkVol/(faucetQ - drainQ);

// IN THIS CONDITION drainingTime WILL RETURN A NEGATIVE VALUE
        IF faucetQ <= drainQ;                OR  IF drainingTime <= 0;
                PRINT ("THE SINK WILL NO OVERFLOW") ;

// IN THIS CONDITION drainingTime WILL RETURN A POSITIVE VALUE
        ELSE IF faucetQ > drainQ;            OR  ELSE IF drainingTime > 0
                PRINT ("THE SINK WILL OVERFLOW") ;
```

EXERCISE TEST

CASE 1 **faucetQ <= drainQ OR drainingTime <= 0**	**CASE 2** **faucetQ > drainQ OR drainingTime > 0**
faucetQ =0.002 drainQ = 0.015 sinkVol = 1.5 drainingTime = 1.5 / (0.002-0.015) = -115.38 THE SINK WILL NO OVERFLOW	faucetQ =0.015 drainQ = 0.002 sinkVol = 1.5 drainingTime = 1.5 / (0.015-0.002) = 115.38 THE SINK WILL OVERFLOW