

PMR: this assignment again was hard to build. I used the how much a car consumes in gas for the example. I had to rely on the internet to help.

/\*\*

\* Project title: 08.08 Arrays of Objects

\*

\* Purpose of Project: Calculate gas used.

\*

\* @version 11/26/2019

\*

\* @author Anika Jallipalli

\*/

public class AnnualFuelUse {

private int fillup, days, startMiles, endMiles, distance;

private double gallonsUsed, mpg, price, cost;

AnnualFuelUse(int fill, int day, int startMi, int endMi, double galUsed, double pricePerGal) {

fillup = fill;

days = day;

startMiles = startMi;

endMiles = endMi;

gallonsUsed = galUsed;

price = pricePerGal;

}

public int getFillup() {

return fillup;

}

public int getDays() {

return days;

}

public int getStartMiles() {

return startMiles;

}

public int getEndMiles() {

return endMiles;

}

public double getGalsUsed() {

return gallonsUsed;

}

public double getPrice() {

return price;

}

public int calcDistance() {

distance = endMiles - startMiles;

return distance;

}

public double calcMPG() {

mpg = ((double)distance) / gallonsUsed;

return mpg;

}

public double calcCost() {

cost = gallonsUsed \* price;

return cost;

}

}

**TESTER**

/\*\*

\* Project title: 08.08 Arrays of Objects

\*

\* Purpose of Project: Calculate gas used.

\*

\* @version 11/26/2019

\*

\* @author Anika Jallipalli

\*/

public class AnnualFuelUseTester {

public static void main(String[] args) {

int fillUp1 = 1;

int fillUp2 = 2;

int fillUp3 = 3;

int days2 = 9;

int days3 = 14;

int startMiles1 = 110000;

int endMiles1 = 110100;

int endMiles2 = 110150;

int endMiles3 = 110200;

double gallonsUsed1 = 11.7;

double gallonsUsed2 = 10.5;

double gallonsUsed3 = 11.4;

double price1 = 1.99;

double price2 = 1.89;

double price3 = 2.09;

AnnualFuelUse[] years = {

new AnnualFuelUse(fillUp1, fillUp1, startMiles1, endMiles1, gallonsUsed1, price1),

new AnnualFuelUse(fillUp2, days2, endMiles1, endMiles2, gallonsUsed2, price2),

new AnnualFuelUse(fillUp3, days3, endMiles2, endMiles3, gallonsUsed3, price3)

};

//Distance

int maxDistance = Integer.MIN\_VALUE;

for (AnnualFuelUse year12 : years) {

if (year12.calcDistance() > maxDistance) {

maxDistance = year12.calcDistance();

}

}

int minDistance = Integer.MAX\_VALUE;

for (AnnualFuelUse year11 : years) {

if (year11.calcDistance() < minDistance) {

minDistance = year11.calcDistance();

}

}

//MPG

double maxMPG = Double.MAX\_VALUE;

for (AnnualFuelUse year10 : years) {

if (year10.calcMPG() < maxMPG) {

maxMPG = year10.calcMPG();

}

}

double minMPG = Double.MAX\_VALUE;

for (AnnualFuelUse year9 : years) {

if (year9.calcMPG() < minMPG) {

minMPG = year9.calcMPG();

}

}

//Price

double maxPrice = Double.MAX\_VALUE;

for (AnnualFuelUse year8 : years) {

if (year8.getPrice() < maxPrice) {

maxPrice = year8.getPrice();

}

}

double minPrice = Double.MIN\_VALUE;

for (AnnualFuelUse year7 : years) {

if (year7.getPrice() < minPrice) {

minPrice = year7.getPrice();

}

}

//Total Distance

int totalDistance = 0;

for (AnnualFuelUse year6 : years) {

totalDistance += year6.calcDistance();

}

//Total Gallons

int totalGalsUsed = 0;

for (AnnualFuelUse year5 : years) {

totalGalsUsed += year5.getGalsUsed();

}

//Total Cost

double totalCost = 0;

for (AnnualFuelUse year4 : years) {

totalCost += year4.calcCost();

}

//Yearly Distance

int yearlyDistance = 0;

for (AnnualFuelUse year3 : years) {

yearlyDistance += (year3.calcDistance() \* 96);

}

//Yearly Gallons

double yearlyGallons = 0;

for (AnnualFuelUse year2 : years) {

yearlyGallons += (year2.calcMPG()) / 2.5;

}

//Yearly Cost

double yearlyCost = 0;

for (AnnualFuelUse year1 : years) {

yearlyCost += (year1.calcCost() \* 96);

}

System.out.println("Fill Up Days Start Miles End Miles Distance " +

"Gallons Used MPG Price Cost");

for (AnnualFuelUse year : years) {

System.out.printf("%4d %8d %12d %12d %8d %12.2f %12.1f %8.2f %8.2f \n", year.getFillup(), year.getDays(),

year.getStartMiles(), year.getEndMiles(), year.calcDistance(),

year.getGalsUsed(), year.calcMPG(), year.getPrice(), year.calcCost());

}

System.out.println();

System.out.printf("Minimum: %3d %25.1f %8.2f\n", minDistance, minMPG, minPrice);

System.out.printf("Maximum: %3d %25.1f %8.2f\n", maxDistance, maxMPG, maxPrice);

System.out.println("Totals: " + totalDistance + " " + totalGalsUsed + " " + totalCost);

System.out.println("Annual Projection: " + yearlyDistance + " " + yearlyCost + " " + yearlyGallons + " " + yearlyCost);

}

}