**Objectives:**
- Translate a problem description into an algorithm of mathematical operations; then write a program from that
- Use the IPOS mindset when developing the algorithm (Input-Processing-Output-Storage)
- Use some conventional programming practices (for asgn 1 and future assignments)
  - top-comment in program, including name, lab section/day/time, asgn #, app name & brief description
  - descriptive naming (and camel case) for variables, methods and classes
- Use variables for intermediate storage (for easier testing) with appropriate data types
- Use hard-coded “input” data for testing, then convert program to use normal user-provided input data
- Use arithmetic operators, assignment operator (=), casting (double), rounding
- Be mindful of operator precedence, intermediate results, rounding/truncating when using mathematical equations
- Use incremental development to complete the project
- Print output report to console (print & println methods & concatenation)

**PROJECT OVERVIEW**
This app allows a user to calculate the cost of commuting from home to the CEAS building (Floyd Hall) at the Parkview campus in order to take a class, say CS1110. To simplify, only 3 options are presented: driving alone vs. ride-sharing with 1 other person vs. ride-sharing with 2 other people. Results show the cost per person for the semester, plus a few other intermediate calculation results. The total cost for the semester depends on:
- the number of trips made (i.e., number of class sessions per week & number of weeks in semester)
- gas costs - i.e., car’s average mpg (miles per gallon) & current gas price per gallon
- distance driven - (NOTE: allow for a “round trip”, since data specified is just “1-way”)
- other driving costs (e.g., insurance, tires, maintenance & repairs, depreciation, parking permit. etc.) are NOT INCLUDED for this assignment.

**INPUT**
During stage 1 of testing your program, hard-code the input data values in the program.
(That way you can more easily check your output results with the specs and with other people in class).
However, for stage 2 for the final project (which you’ll submit for grading), the program needs to be “dynamic” and get its input values from the user for all 5 variables.
For example,
- Stage 1: pricePerGallon variable is initialized as 2.12 inside the program
- Stage 2: pricePerGallon is declared as a variable (and initialized to 0)
  - program asks user to enter value for price per gallon of gas
  - program reads’ that value from the keyboard-input and stores it in the variable, pricePerGallon

Stage 1: Declare the variables for the data items you’ll need from the user, and initialize them to these designated values. Use reasonably short, but descriptive variable names (with appropriate Java-style capitalization format, i.e. camel case naming).
- Number of days per week the class meets: 3
- Number of weeks in the semester: 15
- Average MPG (Miles Per Gallon): 18
- Price of gas (per gallon): $2.12
- 1-way distance to CEAS/Parkview: 23

**OUTPUT to the “console” (output window)**
The output is a report summarizing the input data used, some intermediate results, and the output results that were calculated. See the sample output below. Use the EXACT format/wording/spacing/labelling/... shown in the sample output. **DO NOT HARDCODE ANY** of data values (numbers) in the report itself (or in the intermediate results - the PROGRAM MUST do the calculations, not YOU as a human user. Note that rounding should be done so monetary amounts always show up as 2 decimal places.

**HINT:** use printf(“%.2f”, var_name); to print values with 2 decimal place accuracy.
CALCULATIONS
Your PROGRAM MUST CALCULATE the results – do NOT HARDCODE the results data values or any intermediate values. So declare some additional variables (besides the input variables) to store each result your program will need for the final report. Each amount to be calculated/printed must have its own descriptive variable.

FYI: ONE round trip would be calculated as:

\[
\frac{2.12}{18 \text{ miles}} = \frac{2.12}{(2 \times 23) \text{ miles}}
\]

So $??? = $ what? Do NOT hardcode any of the actual numbers above in your calculations. Use VARIABLE NAMES for these amounts in your formulas. And the variables contain the above values (stage 1: because the variables are initialized to those values; stage 2: because the user provided those values when prompted).

SAMPLE OUTPUT (use this EXACT format – and NO HARDCODING of the data itself)

>>> RIDE SHARE APP <<<

Input data from user
----------------------
Class meetings: 15 weeks/semester, 3 days/week
Gas price: $2.12 per gallon
Car efficiency: averages 18 mpg
Distance (home to CEAS): 23 miles 1-way

Intermediate results
----------------------
Number round trips/semester: 45
Total miles/semester: 2070
Gas cost:
  for 1 round-trip: $5.42
  total for semester: $243.90

Cost for the semester
----------------------
1 person alone: $243.90
2 people sharing: $121.95 (per person)
3 people sharing: $81.30 (per person)

APPENDIX A
Incremental Development

There are several ways to develop/write a program. The most commonly-used software development method is incremental development, where the product is designed, implemented and tested incrementally (i.e., a little more is added each time) until the product is finished and working correctly. It includes the general rule: “always keep the program in a working state, where it produces SOME viewable output”. That is,

1. Add some code to carry out the next step in the algorithm
2. “Compile” it (in Java/NetBeans, “Build” it) and correct all syntax errors
3. Run program and check that the output is correct thus far
4. Repeat steps 1,2,3 until the program is complete and does what the user/client needs it to do.