CIS 121 Assignment – Reading Data from a File. Practice Pass by Value Functions. For each problem develop an IPO within this document. Save the document and upload to Blackboard for grading. Then develop a C++ program for each problem and upload the .cpp file to Blackboard.

1. Read in auto make, model, MSRP (manufacturer suggested resale price) and sales price from a file. See auto.txt below. For each auto (line of data) compute the savings to be MSRP – sales price and sales tax in a single function.

   Pass to the function MSRP, sales price by value, sales price and sales tax by reference. The function will:

   1. Compute savings = MSRP – Sales Price
   2. Compute sales tax = Sales Price * 0.07

   For each auto (line of data) display make, model, MSRP, sales price, savings and sales tax. Sum the savings.

   After all autos (lines of data processed) display sum of savings.

   Auto.txt

   Honda Accord 25000.00 22000.00

   Honda CRV 30000.00 28000.00

   Toyota Corrola 28000.00 25000.00

   Ford Fusion 23000.00 20000.00

Input	Process	outout
Auto make	Call function compute_savings_and_tax(MSRP, sales_price, savings, sales_tax)	Display make, model MSRP, SALES PRICE, SAVINGS, SALES TAX
model	Compute savings = MSRP – sales_price	Display total sum of savings
MSRP	Compute sales_tax = sales_price * 0.07	
Sales price		

2. Read in the grocery item, quantity and cost per item from a file. See grocery.txt below. For each item determine the extended price (quantity * cost per item) within a function.

   Pass to the function quantity and price. The function will:
   3. Compute extended price = quantity * price
   4. Return extended price

   For each item (line of data) display item, quantity, cost per item and extended price.

   Also, sum the extended price.

   After all lines of data are processed, compute tax of 7% (sum of extended price * 0.07). Compute total receipt to be sum of extended price + tax. Display sum of extended price, tax and total receipt.

Grocery.txt

Banana 10 0.50

Oranges 5 0.10

Cookies 15 0.30

Hamburgers 4 1.00

Buns 4 0.50

Soda 10 0.99

input	Process	Output
Item	Call function compute_extended_price(quantity, price) Compute extended_ price = quantity * price Accumulate sum of extended prices	Display item, quantity, cost per item, extended price
quantity		Compute tax = sum * 0.07 Compute total receipt = sum + tax Display sum, tax, total receipt

Cost per item		

3. Read gallons of gas used and miles travelled from a file. See trips.txt below. For each trip compute miles per gallon within a function.

   Pass to the function gallons and miles. Within the function:
   1. Compute mpg = miles / gallons.
   2. Return mpg.

   For each trip (line of data) display gallons, miles and mpg.

   Sum miles travelled and gallons of gas used.

   After all data is processed (end of file) display sum of gallons and sum of miles travelled.

Trips.txt

20 500

10 600

5 50

10 350

input	Process	Output
Gallons, miles	Call compute_mpg(gallons, miles) Compute mpg = miles / gallons	Display gallons, miles, mpg; sum miles and gallons

4. Read in last name and annual salary from a file. See empl.txt below. For each employee (line of data) compute their bi-weekly pay within a function.

   Pass to the function the annual salary. The function will:

   1. Compute bi-weekly salary = annual salary / 26
   2. Return bi-weekly salary

For each line display last name, annual salary and bi-weekly salary. Sum the annual salaries. Count the number of employees (each line of data).

After all employees processed compute average annual salary (total annual salary / count of employees). Display sum of annual salary, count of employees and average annual salary.

Empl.txt

Jones 50000.00

Adams 65000.00

Baker 45000.00

Smith 75000.00

input	Process	Output
Employee last name	Call function compute_biweekly_salary(annual_salary) Compute biweekly_salary = annual_salary / 26	Display last name, annual salary, bi-weekly salary
annual salary		Sum annual salaries, count employees
		c ompute and display average annual salary

5. The input to this program is last name, student code and credits taken. See file student.txt below.

   You are to read each line of data into your program (one line at a time of course). Write a function to calculate tuition owed and course fees in a single function. You must pass to the function student code and credits taken by value and tuition owed and course fees by reference.

   Within the function compute tuition owed and course fees:

   1. Determine cost per credit hour. In district students (student code == 'I') pay $250.00 per credit hour. Out of district students (student code == 'O') pay $500.00 per credit hour.
   2. Calculate tuition owed to be credit hours * cost per credit hour
   3. Compute course fees as 10% of the total tuition.

   Also, within the program sum the tuition owed for all students. Count the number of students. For each student (line of data), display last name, student code, cost per credit hour, credits taken, tuition owed and course fees. Additionally, sum the tuition owed.

After all students are processed (end of loop), calculate the average tuition cost per student (sum of tuition owed / number of students). Display sum of tuition owed, number of students and average tuition cost per student.

Student.txt

Jones I 15.00

Baker O 20.00

Davis I 10.00

Michaels O 12.00

Baez I 12.00

input	Process	Output
Last name, student code, credits	Call function compute_tuition_and_fees(code, credits, tuition_owed, course_fees)	Display last name, code, cost per credit, credits taken, tuition owed, course fees
	Determine cost per credit: I = $250, O = $500	Sum tuition owed, count students
	Compute tuition owed: tuition_owed = credits * cost_per_credit	Compute and display average tuition per student
	Compute course fees: course_fees = tuition_owed * 0.10	