**CSC 1101 – Problem Solving and Programming Laboratory**

**Lab 16 – [your name]**

**25 points – Due March 25, 11pm**

**a)** Save this document with your name and the homework number somewhere in the file name.

**b)** Type/paste your answers into the document.

**c)** Submit this document and your .cpp file(s) to the Canvas item where you downloaded this document. Do not submit a zip file but individually attach your files.

**1) [12 points]** You've been hired by *Wheeled Wrens* to write a C++ console application that estimates bicycle speed in miles per hour. Use a validation loop to prompt for and get from the user the whole-number wheel diameter of a bicycle in inches in the range 10-50. Then use a validation loop to prompt for and get from the user the real-number of rotations the bike wheel makes per pedal rotation in the range 0.1-10. Then use a validation loop to prompt for and get from the user the whole-number pedal rotations per minute in the range 1-120. Format the following four outputs into two formatted columns:

● The three inputs.

● The estimated speed based on the three inputs.

The first column is a left-justified label. The second column is a right-justified value. Use formatted output manipulators to print the output. Declare and use global constants for the column widths. The output should look like this:

Get a bicycle speed estimate (y or n)? y

Enter the wheel diameter in inches (10-50): 27

Enter the number of wheel rotations per pedal rotation (0.1-10): 4

Enter the number of pedal rotations per minute (1-120): 40

Wheel diameter (inches): 27

Wheel-to-pedal ratio: 4.0

Pedal rotations (per minute): 40

Bike speed (miles per hour): 12.9

Get another bicycle speed estimate (y or n)?

Create value function **speedEstimate** that takes the wheel diameter, wheel-to-pedal-ratio, and pedal rotations per minute as inputs and returns the real-number speed estimate. The following three global constants are useful in calculating the speed estimate:

● PI = 3.14159

● MILES\_PER\_INCH = 1. / (12 \* 5280)

● MINUTES\_PER\_HOUR = 60

Format any real numbers to one decimal place. Continue to prompt the user for estimates until they enter sentinel value “n”. Use this data for the last run:

|  |  |  |
| --- | --- | --- |
| Wheel diameter | Wheel-to-pedal ratio | Pedal rotations per minute |
| 60 |  |  |
| 24 | 20 |  |
|  | 5 | 200 |
|  |  | 34 |
| 27 | 2.5 | 40 |
| 30 | 10 | 17 |

*[your program code here]\**

**//==========================================================**

**//**

**// Title: Wheeled Wrens**

**// Course: CSC 1101**

**// Lab Number: Lab 16-01**

**// Author: Trevor Trusty**

**// Date: 3/21/2019**

**// Description:**

**// C++ console application that estimates bicycle speed**

**// in miles per hour.**

**//==========================================================**

**#include <conio.h> // For function getch()**

**#include <cstdlib> // For several general-purpose functions**

**#include <fstream> // For file handling**

**#include <iomanip> // For formatted output**

**#include <iostream> // For cin, cout, and system**

**#include <string> // For string data type**

**using namespace std; // So "std::cout" may be abbreviated to "cout"**

**enum Option{YES, NO};**

**//==========================================================**

**// Globals**

**//==========================================================**

**const int COL1 = 33;**

**const int COL2 = 4;**

**const double PI = 3.1459;**

**const double MILES\_PER\_INCH = 1. / (12 \* 5280);**

**const int MINUTES\_PER\_HOUR = 60;**

**//==========================================================**

**// Prototypes**

**//==========================================================**

**Option optionConvert(char);**

**double speedEstimate(double, double, double);**

**int main()**

**{**

**// Declare variables**

**double wheelDiameter;**

**double perPedal;**

**double perMinute;**

**Option myOption; // YES or NO**

**char myChar; // User input for sentinel value**

**// Show application header**

**cout << "Welcome to Wheeled Wrens!" << endl;**

**cout << "--------------------------" << endl << endl;**

**//Prompt for sentinel value**

**cout << "Get a bicycle speed estimate (y or n)? ";**

**cin >> myChar;**

**myOption = optionConvert(myChar); // Changes input char into type 'option' for sentinel value**

**while (myOption != NO) // Sentinel Loop**

**{**

**cout << "Enter the wheel diameter in inches (10-50): ";**

**cin >> wheelDiameter;**

**while (wheelDiameter < 10 || wheelDiameter > 50)**

**{**

**cout << "Error: Diameter should be between 10 and 50 inches" << endl;**

**cout << "Enter the wheel diameter in inches (10-50): ";**

**cin >> wheelDiameter;**

**}**

**cout << "Enter the number of wheel rotations per pedal rotation (0.1-10):";**

**cin >> perPedal;**

**while (perPedal < 0.1 || perPedal > 10)**

**{**

**cout << "Error: Rotations should be between 0.1 and 10 rotations." << endl;**

**cout << "Enter the numsber of wheel rotations per pedal rotation (0.1-10):";**

**cin >> perPedal;**

**}**

**cout << "Enter the number of pedal rotations per minute (1-120): ";**

**cin >> perMinute;**

**while (perMinute < 1 || perMinute > 120)**

**{**

**cout << "Error: Rotations should be between 1 and 120 rotations.";**

**cout << "Enter the number of pedal rotations per minute (1-120): ";**

**cin >> perMinute;**

**}**

**//Show table**

**cout << fixed << setprecision(1);**

**cout << setw(COL1) << left << "Wheel diameter (inches):";**

**cout << setw(COL2) << right << wheelDiameter << endl;**

**cout << setw(COL1) << left << "Wheel-to-pedal ratio:";**

**cout << setw(COL2) << right << perPedal << endl;**

**cout << setw(COL1) << left << "Pedal rotations (per minute):";**

**cout << setw(COL2) << right << perMinute << endl;**

**cout << setw(COL1) << left << "Bike speed (miles per hour):";**

**cout << setw(COL2) << right << speedEstimate(wheelDiameter, perPedal, perMinute) << endl;**

**cout << "Get another bicycle speed estimate (y or n)? ";**

**cin >> myChar;**

**myOption = optionConvert(myChar); // Convert new response to be checked again by sentinal loop**

**}**

**// Show application close**

**cout << "\nEnd of PI Throwers" << endl << endl;**

**// Pause before application window closes**

**cout << "Press any key to exit ..." << endl;**

**\_getch();**

**return 0;**

**}**

**//==========================================================**

**// End of Execution**

**//==========================================================**

**//==========================================================**

**// optionConvert**

**//==========================================================**

**Option optionConvert(char x)**

**{**

**if (x == 'y')**

**{**

**return YES;**

**}**

**else**

**return NO; // For simplicity, any response except y will end the application**

**}**

**//==========================================================**

**// speedEstimate**

**//==========================================================**

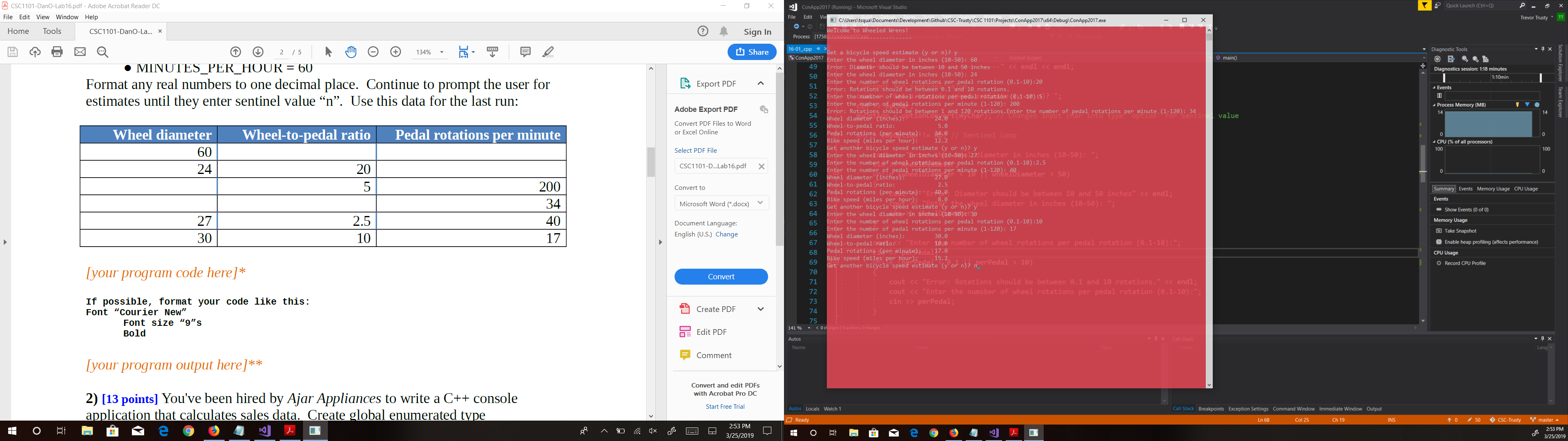
**double speedEstimate(double x, double y, double z)**

**{**

**return x \* y \* z \* PI \* MILES\_PER\_INCH \* MINUTES\_PER\_HOUR;**

**}**

*[your program output here]\*\**



**2) [13 points]** You've been hired by *Ajar Appliances* to write a C++ console application that calculates sales data. Create global enumerated type **appliance\_type** with values:

UNKNOWN, washer, dryer, refrigerator, microwave, toaster, stove

Create function **convertedToApplianceString** that takes an appliance\_type variable, and matches and returns an equivalent string value. If there is no match, return string “UNKNOWN”. Create function **convertedToApplianceType** that takes a string variable, and matches and returns an equivalent appliance\_type value. If there is no match, return value UNKNOWN. In the main function, attempt to open output file **AppliancesSold.txt**. If the file doesn’t open, show an error message and end the application. If the file opens, write column headers (and eventually data) to the file per the following specification:

| Field | Width | Justification |
| --- | --- | --- |
| Appliance | 16 | left |
| Wholesale price | 12 | right |
| Retail price | 12 | right |
| Profit | 12 | right |
| Commission | 12 | right |
| Sales tax | 12 | right |
| Total | 12 | right |

See sample application **Text file input and output per file specification** on Canvas. Prompt for and get from the salesperson the following information:

● Appliance name – the product to be sold. Use a validation loop and function convertedToApplianceType to insure the appliance name is valid.

● Wholesale price – the price *Ajar Appliances* paid for the appliance. Use a validation loop to insure that the price is at least zero.

● Retail price – the price *Ajar Appliances* is selling the appliance for. Use a validation loop to insure that the price is at least the wholesale price.

Format the following seven outputs into two formatted columns:

● The three inputs.

● The profit *Ajar Appliances* will earn selling the appliance (retail – wholesale).

● The 2% commission the *Ajar Appliances* salesperson will earn selling the appliance (profit \* 0.02).

● The 6% Michigan sales tax on the appliance (retail \* 0.06).

● The total cost to the customer (retail + sales tax).

The first column is a left-justified label. The second column is a right-justified value. Use formatted output manipulators to print the output. Declare and use global constants for the 2% commission rate, the 6% sales tax rate, and the column widths. Format any real numbers to two decimal places. Write the same seven values to file **AppliancesSold.txt** in one line per the above file specification. Continue to prompt the user for sales until they enter sentinel value “n”. The output should look like this:

Welcome to Ajar Appliances

--------------------------

Get a sale (y or n)? y

Enter an appliance (washer, dryer, refrigerator, microwave, toaster, stove): dry

Error: invalid appliance name of 'dry'.

Enter an appliance (washer, dryer, refrigerator, microwave, toaster, stove): dryer

Enter the wholesale price (>= $0): -200

Error: the wholesale price must be at least $0.

Enter the wholesale price (>= $0): 400

Enter the retail price (>= $400.00): 200

Error: the retail price must be at least $400.00.

Enter the retail price (>= $400.00): 500

Appliance: dryer

Wholesale price ($): 400.00

Retail price ($): 500.00

Profit ($): 100.00

Commission (2%): 2.00

Sales tax (6%): 30.00

Total cost: 530.00

Get another sale (y or n)?

The start of your AppliancesSold.txt file should look like this:



Use these inputs for your last run:

| Appliance | Wholesale price | Retail price |
| --- | --- | --- |
| sink |  |  |
| stove | -100, 300 | 200, 500 |
| dryer | 750 | 775 |
| washer | 1100 | 1199 |
| n (sentinel value) |  |  |

*[your program code here]\**

**If possible, format your code like this:**

**//==========================================================**

**//**

**// Title: Ajar Appliances**

**// Course: CSC 1101**

**// Lab Number: Lab 16-02**

**// Author: Trevor Trusty**

**// Date: 3/21/2019**

**// Description:**

**// C++ console application that calculates sales data.**

**//**

**//==========================================================**

**#include <conio.h> // For function getch()**

**#include <cstdlib> // For several general-purpose functions**

**#include <fstream> // For file handling**

**#include <iomanip> // For formatted output**

**#include <iostream> // For cin, cout, and system**

**#include <string> // For string data type**

**using namespace std; // So "std::cout" may be abbreviated to "cout"**

**//==========================================================**

**// Globals**

**//==========================================================**

**enum appliance\_type{UNKNOWN, washer, dryer, refrigerator, microwave, toaster, stove};**

**const string SALES\_OUT = "AppliancesSold.txt";**

**const int COLFMT1 = 32;**

**const int COLFMT2 = 6;**

**//==========================================================**

**// Prototypes**

**//==========================================================**

**string convertedToApplianceString(appliance\_type);**

**appliance\_type convertedToApplianceType(string);**

**int main()**

**{**

**// Declare variables**

**int x;**

**ofstream outFile;**

**// Show application header**

**cout << "Welcome to Wheeled Wrens!" << endl;**

**cout << "--------------------------" << endl << endl;**

**outFile.open(SALES\_OUT);**

**if (!outFile.is\_open())**

**{**

**cout << "Error: Could not open " << SALES\_OUT << endl;**

**}**

**else**

**{**

**outFile << setw(COLFMT1) << left << "Appliance"**

**<< setw(COLFMT2) << right << "Wholesale price"**

**<< setw(COLFMT2) << right << "Retail price"**

**<< setw(COLFMT2) << right << "Profit"**

**<< setw(COLFMT2) << right << "Commissio"**

**<< setw(COLFMT2) << right << "Sales tax"**

**<< setw(COLFMT2) << right << "Total" << endl;**

**for (int i = 0; i < x; i++)**

**{**

**}**

**}**

**// Show application close**

**cout << "\nEnd of PI Throwers" << endl << endl;**

**// Pause before application window closes**

**cout << "Press any key to exit ..." << endl;**

**\_getch();**

**return 0;**

**}**

**//==========================================================**

**// End of Execution**

**//==========================================================**

**//==========================================================**

**// convertedToApplianceString**

**//==========================================================**

**string convertedToApplianceString(appliance\_type x)**

**{**

**if (x == washer)**

**return "washer";**

**else if (x == dryer)**

**return "dryer";**

**else if (x == refrigerator)**

**return "refrigerator";**

**else if (x == microwave)**

**return "microwave";**

**else if (x == toaster)**

**return "toaster";**

**else if (x == stove)**

**return "stove";**

**else if (x == UNKNOWN)**

**return "UNKNOWN";**

**}**

**//==========================================================**

**// convertedToApplianceType**

**//==========================================================**

**appliance\_type convertedToApplianceType(string x)**

**{**

**if (x == "washer")**

**return washer;**

**else if (x == "dryer")**

**return dryer;**

**else if (x == "refrigerator")**

**return refrigerator;**

**else if (x == "microwave")**

**return microwave;**

**else if (x == "toaster")**

**return toaster;**

**else if (x == "stove")**

**return stove;**

**else if (x == "UNKNOWN")**

**return UNKNOWN;**

**}**

*[your program output here]\*\**

*[your AppliancesSold.txt file here]\*\*\**