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

**Lab 22 – rory lange**

**25 points – Due April 19, 11pm**

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

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

**c)** Submit the following documents to the Canvas assignment link where you downloaded this document:

✓ This document.

✓ Your .cpp files renamed to .txt.

Submit the documents separately, not as one .zip file.

You've been hired again by *Part Pretenders* to enhance the great C++ console application you wrote for Lab 20. Expand the string array **parts** size from eight to ten. Keep the initializer list that stores eight car parts in the array. Use one word per part. Don't store the parts in alphabetical order. Create the following functions:

**void addPart(string parts[], int &partCount)**

This void function takes the **parts** array and current part count. It checks if there is room to add a part to the **parts** array. If not, it prints a message. If so, it prompts for and gets a part from the user, adds it to the next open spot in the array, updates the part count, and prints a "part added" message. It also prints a reminder to resort the data since a part has been added to the list. Since **partCount** is a reference variable, the updated count is returned to the calling function.

**void deletePart(string parts[], int &partCount)**

This void function takes the **parts** array and current part count. It checks if there is a part to delete from the **parts** array. If not, it prints a message. If so, it prints a "part removed" message and updates the part count. Since **partCount** is a reference variable, the updated count is returned to the calling function.

Change the following function:

**int menuOption()**

This value function presents the following menu to the user:

Part Pretenders Menu

1 - List car parts

2 - Sort car parts

3 - Search car parts

4 - Add car part

5 - Delete car part

9 – Exit

Enter an option:

It then prompts for and gets from the user an option, and returns it. Here are the option descriptions:

**1 – List car parts** calls function listParts.

**2 – Sort car parts** calls function sortParts.

**3 – Search car parts** calls function searchParts.

**4 – Add part** calls function addPart.

**5 – Delete part** calls function deletePart.

Since you now have a partially-filled array, add an integer count to function main that keeps track of the number parts in the array. Continue to read and process menu options until the user enters the sentinel value of 9. The output for add and delete should look like:

Welcome to Part Pretenders

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

Part Pretenders Menu

1 - List car parts

2 - Sort car parts

3 - Search car parts

4 - Add car part

5 - Delete car part

9 - Exit

Enter an option: 1

Part List

Windshield

Bumper

Mirror

Panel

Transmission

Filter

Tire

Grille

Part count: 8

Part Pretenders Menu

1 - List car parts

2 - Sort car parts

3 - Search car parts

4 - Add car part

5 - Delete car part

9 - Exit

Enter an option: 4

Enter the part to add: Trunk

Part 'Trunk' added to list.

Remember to resort list before you search.

Part Pretenders Menu

1 - List car parts

2 - Sort car parts

3 - Search car parts

4 - Add car part

5 - Delete car part

9 - Exit

Enter an option: 1

Part List

Windshield

Bumper

Mirror

Panel

Transmission

Filter

Tire

Grille

Trunk

Part count: 9

Part Pretenders Menu

1 - List car parts

2 - Sort car parts

3 - Search car parts

4 - Add car part

5 - Delete car part

9 - Exit

Enter an option: 5

Part 'Trunk' removed from list.

Part Pretenders Menu

1 - List car parts

2 - Sort car parts

3 - Search car parts

4 - Add car part

5 - Delete car part

9 - Exit

Enter an option: 1

Part List

Windshield

Bumper

Mirror

Panel

Transmission

Filter

Tire

Grille

Part count: 8

Part Pretenders Menu

1 - List car parts

2 - Sort car parts

3 - Search car parts

4 - Add car part

5 - Delete car part

9 - Exit

Enter an option: 9

End of Part Pretenders

Run the program using the following menu option order:

1

2

1

4

1

2

1

5

1

//==========================================================

//

// Title:      Part Pretenders

// Course:     CSC 1101

// Lab Number: 20-1

// Author:     rory lange

// Date:       4/15/21

// Description:

//   This C++ console application manages part names.  It

// declares string array parts of size eight, and stores

// eight parts in it.  It presents the following menu to the

// user:

//

//   Part Pretenders Menu

//   1 - List car parts

//   2 - Sort car parts

//   3 - Search car parts

//   9 – Exit

//

//   Enter an option :

//

// It continues to read and process menu options until the

// user enters the sentinel value of 9.

//

//==========================================================

#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

//==========================================================

const int ARRAY\_SIZE = 10;

//==========================================================

// binarySearchParts

//==========================================================

int binarySearchParts(string parts[], int arraySize, string key)

{

  // Declare variables

  int min = 0;

  int index;

  int max = arraySize - 1;

  // Loop to find key

  while (min <= max)

  {

    // Get midpoint of array

    index = (min + max) / 2;

    // Test if key found

    if (parts[index] == key)  // Guess is right on

      return index;

    else if (parts[index] < key)  // Guess is too low

      min = index + 1;

    else

      max = index - 1;

  }

  // Return index

  return -1;

}

//==========================================================

// menuOption

//==========================================================

int menuOption()

{

  // Declare variables

  int option;

  // Show menu and get option

  cout << "\nPart Pretenders Menu" << endl;

  cout << "1 - List car parts" << endl;

  cout << "2 - Sort car parts" << endl;

  cout << "3 - Search car parts" << endl;

  cout << "4 - Add car parts" << endl;

  cout << "5 - Delete car parts" << endl;

  cout << "9 - Exit" << endl;

  cout << "\nEnter an option: ";

  cin >> option;

  return option;

}

//==========================================================

// listParts

//==========================================================

void listParts(string parts[], int arraySize)

{

  // Loop to list parts

  cout << endl << "Part List" << endl;

  for (int i = 0; i < arraySize; i++)

    cout << parts[i] << endl;

  cout << endl << "Part count: " << arraySize << endl;

}

//==========================================================

// searchParts

//==========================================================

void searchParts(string parts[], int arraySize)

{

  // Declare variables

  string key;

  int index;

  cout << "\nEnter a part to search for (no spaces): ";

  cin >> key;

  index = binarySearchParts(parts, arraySize, key);

  if (index != -1)

    cout << "'" << key << "' found at index "

    << index << "." << endl;

  else

    cout << "'" << key << "' NOT found." << endl;

}

//==========================================================

// sortParts

//==========================================================

void sortParts(string parts[], int arraySize)

{

  // Declare variables

  string value;

  int spot;

  // Loop to test each value

  for (int i = 1; i < arraySize; i++)

  {

    // Loop to find spot to place value

    value = parts[i];

    spot = i - 1;

    while (spot >= 0 && parts[spot] > value)

    {

      parts[spot + 1] = parts[spot];

      spot = spot - 1;

    }

    // Place value in spot

    parts[spot + 1] = value;

  }

  // Print sorted message

  cout << endl << "Parts sorted." << endl;

}

//==========================================================

// addPart

//==========================================================

void addPart(string parts[], int &partCount) {

  string part;

  if (partCount < ARRAY\_SIZE) {

    cout << "Enter a part to add: ";

    cin >> part;

    parts[partCount] = part;

    partCount++;

    cout << "Don't forget to sort the list before searching." << endl;

  }

  else

    cout << "ERROR: too many parts in list." << endl;

}

//==========================================================

// deletePart

//==========================================================

void deletePart(string parts[], int &partCount) {

  if (partCount > 0) {

    cout << "part '" << parts[partCount-1] << "' removed" << endl;

    parts[partCount] = "";

    partCount--;

  }

  else

    cout << "ERROR: no parts to delete." << endl;

}

//==========================================================

// main

//==========================================================

int main()

{

  // Declare variables

  // Put your own 8 car parts in for use to start

  string parts[ARRAY\_SIZE] =

    {"wheels", "transmission", "dashboard", "mirrors", "airbags", "engine", "driveterrain", "windows"};

  int option;

  int count = 8;

  // Show application header

  cout << "Welcome to Part Pretenders" << endl;

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

  // Loop to process options

  option = menuOption();

  while (option != 9)

  {

    // Handle option

    switch (option) {

      // List parts

    case 1:

      listParts(parts, count);

      break;

      // Sort parts

    case 2:

      sortParts(parts, ARRAY\_SIZE);

      break;

      // Search parts

    case 3:

      searchParts(parts, ARRAY\_SIZE);

      break;

      //add parts

    case 4:

      addPart(parts, count);

      break;

      //delete parts

    case 5:

      deletePart(parts, count);

      break;

      // Handle invalid option

    default:

      cout << endl << "Error: unknown option of '" << option

        << "'." << endl;

    }

    // Get next option

    option = menuOption();

  }

  // Show application close

  cout << "\nEnd of Part Pretenders" << endl;

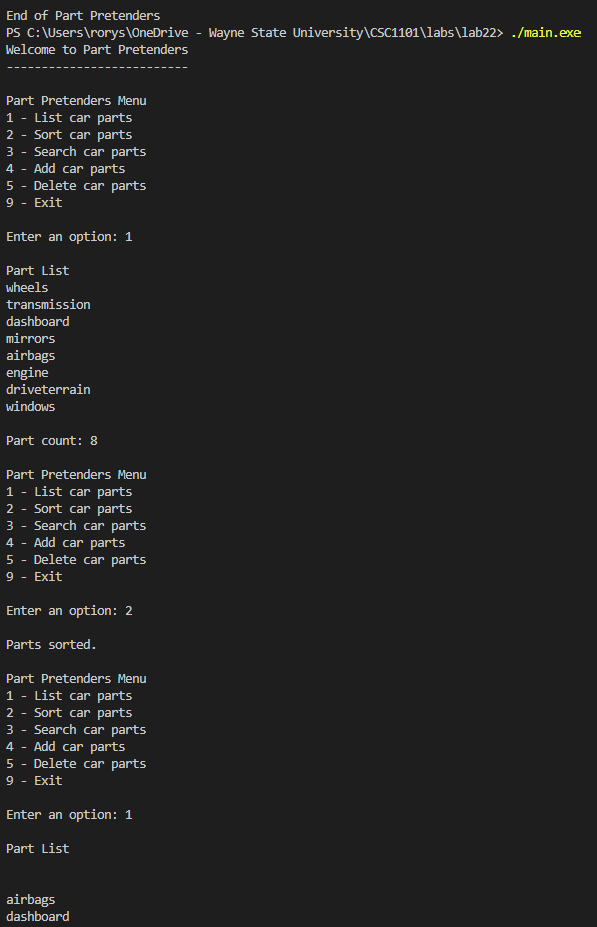
}

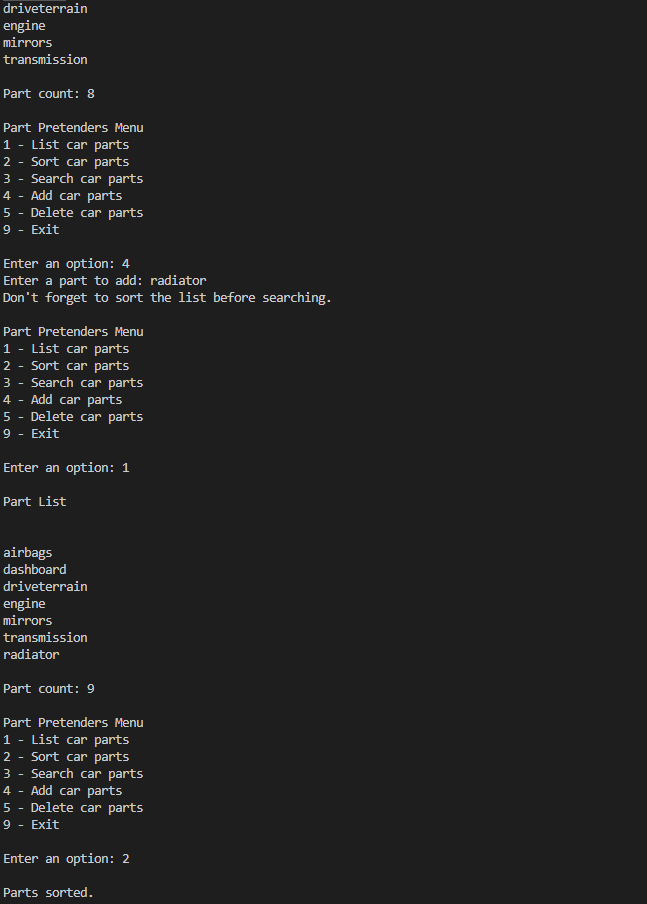
**If possible, format your code like this:**

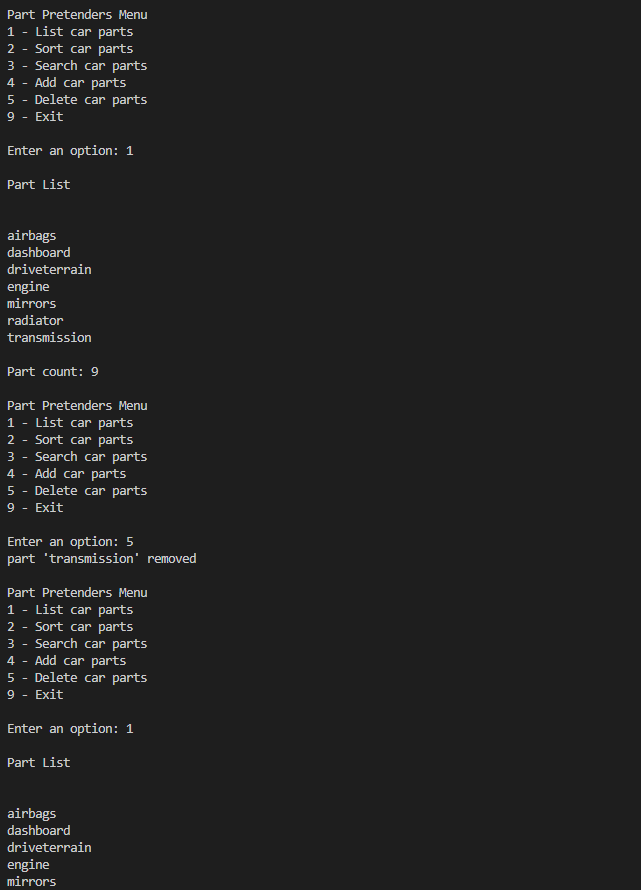
**Font “Courier New”**

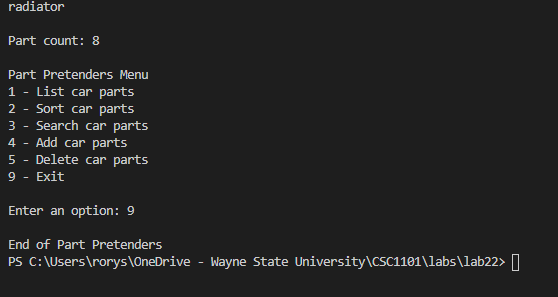
**Font size “9”s**

**Bold**









**\* Copying-and-pasting C++ code to a Word document**

**macOS**

1) From within the C++ program, press **command-A** and press **command-C**.

2) From within the Word document, press **command-V**.

**Windows**

1) From within the C++ program, press **CTRL-A** and press **CTRL-C**.

2) From within the Word document, press **CTRL-V**.

**\*\* Copying-and-pasting C++ console application output to a Word document**

**macOS**

1) From the C++ console, press **shift-command-4-space**.

2) From within the Word document, **command-V**.

**Windows**

1) From the C++ console, press **ALT-PrintScreen**.

2) From within the Word document, press **CTRL-V**.