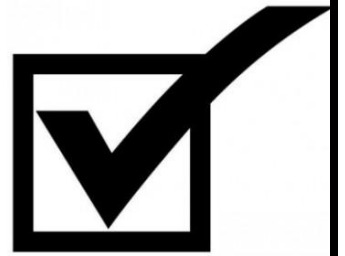


**BUSINESS
PROFESSIONALS**
of AMERICA
Giving Purpose to Potential



C++ PROGRAMMING

(335)

REGIONAL 2026

PRODUCTION:

Regional_MenuProject

_____ (540 points)

Test Time: 90 minutes

GENERAL GUIDELINES.

Failure to follow any of these rules may result in disqualification:

1. **Submission Requirements:** Contestants must submit this test booklet along with any printouts.
2. **Permitted Items:** Only the equipment, supplies, and materials specified for this event are allowed in the testing area. Previous BPA tests and sample tests (whether handwritten, photocopied, or typed) are not permitted.
3. **Electronic Devices:** Electronic devices will be monitored according to ACT standards.

EXAM GUIDELINES.

1. Your name and/or school name should *not* appear on work you submit for grading.
2. Create a folder on the flash drive provided using your contestant number as the name of the folder.
3. Copy your entire solution/project into this folder.
4. Submit your entire solution/project so that the graders may open your project to review the source code.
5. Ensure that the files required to run your program are present and will execute on the flash drive provided.

*Note that the flash drive letter may *not* be the same when the program is graded as it was when you created the program. Use relative file paths, not absolute file paths.

The graders will *not* alter your source code. Submissions that do *not* contain source code will *not* be graded. The project must include an executable program file.

Commenting for Source Code Review:

- Certain sections of your code will be graded. These gradable blocks of code can range from creating data structures, method algorithms, exception handling, and class construction.
 - **Code Commenting Requirements:** Clear and concise comments must be included for all major components of the program, specifically:
 - **Input:** Document how data is received or acquired, including user inputs, file reads, the format of the input, API requests, etc.
 - **Processing:** Provide comments that explain the logic, algorithms, or transformations applied to the input data.
 - **Output:** Describe how and where the final results are produced, the format of the output, whether through user interfaces, files, or other systems.
 - These comments should enhance understanding of the program's flow and intent, making it easier for others to read, maintain, and debug the code.
- The grading rubric contains a section called Source Code Review: in this section are listed descriptions of all the graded programming concepts.
- Each gradable item and any significant programming area should be commented. There will be points awarded for proper commenting.

C++ Regional

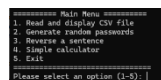
You will be developing a console application with C++. The goal of this application is to create a simple menu system that will perform various actions to demonstrate your programming skills at the regional level competition. Your program will need to capture all the important data from the user and/or file as well as handle any exceptions or data entry errors.

Menu Options You Will Be Programming

When the program begins, you will see a menu appear in the console. There are 5 selections that the user will be able to select. Each menu item can be programmed independently, which means you can pick and choose which menu items you want to program. Below the image (Figure 1) is a description of each menu item. Your menu should handle any invalid inputs that are not option 1 to 5.

General Notes for All Menu Options:

1. The program should gracefully handle any invalid input by prompting the user to try again without crashing.
2. All numeric inputs should reject letters, symbols, decimals, and negative values where integers are expected.
3. User inputs must be validated to ensure they fall within appropriate ranges or match required formats.
4. Each menu option should have its own `function/method.



```
===== Main Menu =====
1. Read and display CSV file
2. Generate random passwords
3. Reverse a sentence
4. Simple calculator
5. Exit
Please select an option (1-5): |
```

Figure 1

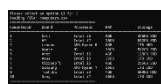
Menu Option 1: Read and Display CSV File

Task Description:

The student must write a function named “readCSV”. This function will take in no information and will return no information. It will read, parse and display the contents of the provided CSV file (**computer.csv**) in a structured, tabular format. The output should include column headers and rows of data. They need to ensure proper formatting based and handle any file-related errors, such as the file being missing or inaccessible.

Expected Behaviors:

- If the file is missing or unreadable, the program should display an appropriate error message.
- Data must be displayed in a neatly aligned table with headers as shown in figure 2.



```
===== Read CSV File =====
make    model    year    price    mpg
Ford    Focus    2012    12500    24
Chevrolet    Malibu    2011    11500    26
Toyota    Camry    2010    10500    24
Honda    Accord    2009    9500     24
Nissan    Altima    2008    8500     24
Ford    Mustang    2007    7500     24
Chevrolet    Silverado    2006    6500     24
Toyota    Prius    2005    5500     24
Honda    Civic    2004    4500     24
Ford    Taurus    2003    3500     24
Nissan    Sentra    2002    2500     24
Chevrolet    Cobalt    2001    1500     24
Toyota    Corolla    2000    1000     24
Honda    Odyssey    1999    500      24
Ford    Explorer    1998    0        24
===== End of CSV File =====
```

Figure 2

Menu Option 2: Generate Random Passwords

Task Description:

The student needs to write a function named “randomPassword” to generate a user provided number of secure random passwords. The number must be between 1 and 9 inclusively. Each password should meet complexity requirements (e.g., include uppercase letters, lowercase letters, digits, and symbols).

Expected Behaviors:

- Prompt the user to re-enter valid input if the input is non-integer or out of range of 1 - 9.
- Generate passwords that meet the complexity criteria and display them. Each password needs to be 8 characters long. The password can consist of any of the following characters:
abcdefghijklmnopqrstuvwxyz1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ
WZ!@#%\$



Menu Option 3: Reverse a Sentence

Task Description:

The student must write a function named “reverseSentence”. This function will take in no information and will return no information. It will prompt the user to input a sentence, reverse the characters in the sentence, and display the reversed string. Punctuation and spaces must also be reversed. See figure 3.

Expected Behaviors:

- If the user enters no input, prompt them to enter a valid sentence.
- Display the reversed sentence, including spaces and punctuation.



Menu Option 4: Simple Calculator

Task Description:

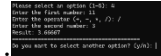
The student will write a function named “simpleCalculator”. This function will take in no information and will return no information. This function will implement a simple calculator that performs addition, subtraction, multiplication, or division operations based on user input. The programmer can assume that all numeric data has a floating point. See figure 4.

Improper Input:

- Invalid operators (anything other than +, -, *, /).
- Non-numeric values for the operands.
- Division by zero.

Expected Behaviors:

- Prompt the user to re-enter valid input for invalid operators or non-numeric operands.
- For division display an error if the user attempts to divide by zero and return to the main menu



Menu Option 5: Exit

Task Description:

The student will write a function named “exitProgram”. This function will take in no information, but it will return a boolean value on whether or not to exit the program. This function will be call from the menu after each iteration of the menu.

Expected Behaviors:

- Closes the program or returns the menu.

Requirements/Explanations:

1. You must use the provided C++ file: **Regional_MenuProject.cpp**
2. The computers.csv file has been given to you; you are responsible for placing it in a location that allows the program to import the data from a relative location so that your program files can be uploaded into the grader computer and operate as intended.

Rubric:

Solution and Project		
The C++ project file is correctly named and present on the flash drive in a single folder with your contest ID		10
Program Execution (<i>If the program does not execute, then the remaining items in this section receive a score of zero</i>). All menu options must work according to the instructions.		
Main Menu (appearance and performance)		20
#1 Read and Display CSV File in tabular form that is evenly spaced and aligned correctly		40
#1 Invalid Input for Read and Display CSV File		10
#2 Generate Random Passwords		50
#2 Invalid Input for Generate Random Passwords		20
#3 Reverse a Sentence		30
#3 Invalid Input for Reverse a Sentence		10
#4 Simple Calculator		30
#4 Invalid Input for Simple Calculator		20
#5 Exit		20
Subtotal		/260 Points

Source Code Review		
Program is properly commented (See Commenting for Source Code Review above)		50
Main menu is validated and handles all user inputs		40
#1 Read and Display the contents CSV File: Reads and parsed all data in the CSV file. Outputs the values in tabular for as shown in the sample provided		50
#2 Generate Random Passwords: Creates the number of passwords requested by the user. Uses characters provided in the instructions.		50
#3 Reverse a Sentence: Reverses the string entered by the user. Checks for an empty string.		30
#4 Simple Calculator: Validates user input. Calculates the answer based on user input		50
#5 Exit: Asks the user if they would like to select another option. Returns a Boolean value.		10
Subtotal		/280 Points
Total Points		/540 Points

```

1  ✓ #include <iostream>
2  ✓ #include <fstream>
3  ✓ #include <sstream>
4  ✓ #include <vector>
5  ✓ #include <string>
6  ✓ #include <iomanip>
7  ✓ #include <ctime>
8  ✓ #include <random>
9  ✓ //include <stdlib>
10 ✓ #include <algorithm> // For std::shuffle, std::reverse, and std::sort
11 ✓ #include <limits>    // For numeric_limits
12 using namespace std;
13
14
15 ✓ // SC9: The switch statement and how it maps operators to corresponding calculations.
16 // Function to perform a simple calculator operation
17 ✓ void simpleCalculator() {
18     double num1, num2;
19     char operation;
20     cin.ignore(80, '\n');
21     // Input numbers and operator
22     cout << "Enter the first number: ";
23     cin >> num1;
24     cout << "Enter the operator (+, -, *, /): ";
25     cin >> operation;
26     cout << "Enter the second number: ";
27     cin >> num2;
28
29     // Perform calculation
30     switch (operation) {
31     case '+':
32         cout << "Result: " << num1 + num2 << "\n";
33         break;
34     case '-':
35         cout << "Result: " << num1 - num2 << "\n";
36         break;
37     case '*':
38         cout << "Result: " << num1 * num2 << "\n";
39         break;
40     case '/':
41         if (num2 == 0) {
42             cout << "Error: Division by zero is not allowed.\n";
43         }
44         else {
45             cout << "Result: " << num1 / num2 << "\n";
46         }
47         break;
48
49     default:
50         cout << "Error: Invalid operator. Please use +, -, *, /, or %.\n";
51     }
52     cout << "=====\n";
53 }

```

```

54
55 // SC12: The use of the reverse algorithm on the sentence string.
56 // Function to reverse a sentence
57 void reverseSentence() {
58     cin.ignore(); // Clear the input buffer
59     cout << "Enter a sentence: ";
60     string sentence;
61     getline(cin, sentence);
62
63     // Reverse the sentence
64     reverse(sentence.begin(), sentence.end());
65
66     cout << "Reversed sentence: " << sentence << "\n";
67     cout << "=====\n";
68 }
69
70 // SC14: The formatting logic used to display the CSV data in a table format.
71 // Function to read and display CSV file in an organized table format
72 void readAndDisplayCSV(const string& filePath) {
73     ifstream file(filePath);
74
75     if (!file.is_open()) {
76         cerr << "Error: Unable to open the file at " << filePath << "\n";
77         return;
78     }
79     bool isHeader = true;
80     cout << "Reading file: " << filePath << "\n";
81     cout << "=====\n";
82     string line;
83     vector<vector<string>> data; // To store all rows and their parsed values
84
85     while (getline(file, line)) {
86         stringstream ss(line); // Create a stringstream from the current line
87         string cell;
88         vector<string> row_data; // To store values of the current row
89
90         // Parse cells using comma as delimiter
91         while (getline(ss, cell, ',')) {
92             row_data.push_back(cell);
93         }
94         data.push_back(row_data); // Add the parsed row to the main data structure
95         cout << left;
96         cout << setw(15) << row_data[0] << setw(15) << row_data[1]
97             << setw(15) << row_data[2] << setw(15) << row_data[3] <<
98             setw(15) << row_data[4] << endl;
99         if (isHeader)
100         {
101             cout << "-----" << endl;
102             isHeader = false;
103         }
104     }
105     cout << "=====\n";
106     file.close();
107 }
108
109

```



```

110     // Function to generate a random password
111     void randomPassword() {
112         const string possible = "abcdefghijklmnopqrstuvwxyz1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ!@#$%";
113         srand((unsigned)time(NULL));
114         int random = rand() % 10;
115         int numPasswords;
116         do
117         {
118             cout << "Enter the number of passwords to generate: ";
119             cin >> numPasswords;
120             if (numPasswords <= 0 or numPasswords > 10)
121             {
122                 cout << "Illegal entry: Please enter a number between 1 and 8\n";
123             }
124         } while (numPasswords <= 0 or numPasswords > 10);
125         string password;
126         for (int lcv1 = 0; lcv1 < numPasswords; lcv1++)
127         {
128             for (int lcv2 = 0; lcv2 < 8; lcv2++)
129             {
130                 random = rand() % 67;
131                 password += possible[random];
132             }
133             cout << password << endl;
134             password = "";
135         }
136         cout << "=====\n";
137     }
138
139
140
141
142     bool exit()
143     {
144         bool ans;
145         char answer;
146         // Ask the user if they want to continue
147         do
148         {
149             cout << "Do you want to select another option? (y/n): ";
150             cin >> answer;
151             if (answer != 'y' and answer != 'Y' and answer != 'n' and answer != 'N')
152             {
153                 cout << "Please enter either a 'y' or 'n'." << endl;
154             }
155         } while (answer != 'y' and answer != 'Y' and answer != 'n' and answer != 'N');
156
157         if (answer == 'y' or answer == 'Y')
158             return false;
159         return true;
160     }
161
162

```

```

164         // SC17: The loop logic for repeatedly displaying the menu and handling user choices.
165     int main() {
166         bool done = false;
167         int choice;
168         while (!done) {
169             system("CLS");
170             cout << "===== Main Menu =====\n";
171             cout << "1. Read and display CSV file\n";
172             cout << "2. Generate random passwords\n";
173             cout << "3. Reverse a sentence\n";
174
175             cout << "4. Simple calculator\n";
176             cout << "5. Exit\n";
177
178             cout << "=====\n";
179             cout << "Please select an option (1-5): ";
180
181             // Variable to store user's choice
182
183
184             cin >> choice;
185
186             // Handle menu options
187             if (choice == 1) {
188                 // Path to the CSV file
189                 string filePath = R"(computers.csv)";
190                 readAndDisplayCSV(filePath);
191                 done = exit();
192             }
193             else if (choice == 2) {
194                 randomPassword();
195                 done = exit();
196
197             else if (choice == 3) {
198                 reverseSentence();
199                 done = exit();
200             }
201
202             else if (choice == 4) {
203                 simpleCalculator();
204                 done = exit();
205             }
206             else if (choice == 5) {
207                 done = true;
208             }
209
210             else if (choice >= 6 or choice < 1) {
211
212                 cout << "Invalid selection. Please enter a number between 1 and 5.\n";
213             }
214         }
215         return 0;
216     }

```