Lab 2-1 Up to Speed Requirement Details

The focus of these problems will be working with information extracted from a

municipal government data feed containing bids submitted for auction of property.

The data set is provided in two comma-separated files:

1. eBid\_Monthly\_Sales.csv (larger set of 17,937 bids)
2. eBid\_Monthly\_Sales\_Dec\_2016.csv (smaller set of 179 bids)

This assignment is designed to quickly get up to speed with the C++ language and the development environment you will be using throughout this course. We will build a simple console program that uses a menu to enable testing of the logic you will complete. In this version the following menu is presented when the program is run:

|  |
| --- |
| **Menu:**  **1. Enter a Bid**  **2. Load Bids**  **3. Display All Bids**  **9. Exit**  **Enter choice:** |

The Lab2-1.cpp program is partially completed - it contains token placeholders such as:

* ?type?
* ?variable?
* ?retval? (return value)

Replace these tokens with appropriate C++ identifiers so that the program will compile and run correctly.

Setup: Begin by creating a new C++ Project with a Project Type of "Hello World C++ Project".

1. Name the project Lab2-1, remember to pick the correct compiler in **Toolchains** and click Finish. This will create a simple Lab2-1.cpp source file under the **/src** directory.
2. Download the lab files and copy them to the following directories:
3. **/Lab2-1**
   1. eBid\_Monthly\_Sales (.csv file)
   2. eBid\_Monthly\_Sales\_Dec\_2016 (.csv file)
4. **/src**
   1. Lab 2-1.cpp (replacing the existing auto-generated one)
   2. CSVparser.cpp
   3. CSVparser.hpp

Remember to right-click on the project in the Project Explorer pane on the left and **'Refresh'** the project so it adds all the new files to the correct folder underneath.

1. Because this activity uses C++ 11 features you must follow the instructions under “C++ Compiler Version” in the *C++ Development Installation guide* to add **-std=c++11** compiler switch to the Miscellaneous settings.

Task 1: Reference the CSVParser library

In order to use the CSVParser library functionality in your program you must declare it (or "bring in") those function definitions so they will be compiled into your executable program.

Task 2: Define a vector data structure to hold a collection of bids.

Task 3: Create a data structure and add to the collection of bids.

1. Each bid read from the input file must be stored in a Bid structure and that structure added to the vector.

Task 4: Define a vector to hold all the bids.

1. Within the main() method we need a vector defined to hold all the bids returned from the loadBids() method.

Task 5: Complete the method call to load the bids.

Task 6: Loop and display the bids read.

1. Create a loop to iterate over every bid in the vector and pass each bid instance to the displayBid() method.
2. Define a timer variable
3. Initialize a timer variable before loading bids
4. Calculate elapsed time and display result

Here is sample output from running the completed program to illustrate the separate activities:

|  |  |  |  |
| --- | --- | --- | --- |
| **Example Input** | **Choice: 2** | **Choice: 3** | **Choice: 9** |
| **Display** | Menu:  1. Enter a Bid  2. Load Bids  3. Display All Bids  9. Exit  Enter choice: 2 | Menu:  1. Enter a Bid  2. Load Bids  3. Display All Bids  9. Exit  Enter choice: 3 | Menu:  1. Enter a Bid  2. Load Bids  3. Display All Bids  9. Exit  Enter choice: 9 |
| **Output** | 179 bids read  time: 2993 milliseconds  time: 2.993 seconds | Hoover Steam Vac | 27 | Enterprise  Table | 6 | General Fund  ...  ...  5 Chairs | 19 | General Fund  2 Chairs | 20 | General Fund  Chair | 71.88 | General Fund | Good bye. |