**CS 1400 Fundamentals of Programming**

**Programming Project #04**

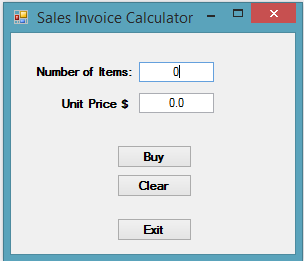
**Sales Invoice Windows Form Application**

**Version 1.0**

**Objective:**

After the completion of this project, you will have created an application that

* contains a simple ***class declaration***,
* creates an ***object*** of the class,
* sends ***messages*** to the object to store and retrieve information, and
* uses a Graphical User Interface (***GUI***).

**Project Description:**

In this Project you will design and create a class that models a simple Sales Invoice. All of the data for a sale and the business logic, required for your program, will be contained in your **SalesInvoice** Domain class.

Your class may only contain variables to hold the number of ***items*** sold, the ***unitPrice*** for these items and the ***salesTicket*** for the items sold.

Figure Form for Sales Invoice Calculator

Your class should also contain the interface methods needed by the **Form** class to create an invoice and present it to the user. Design your user interface so that it offers the functionality described below. Your interface should be intuitive and easy to use, with instructions if needed.

***Program:***

Your program should work as follows:

* The user will enter in the number of ***items*** sold and a ***unitPrice***.
* When the user clicks on the **Buy** Button, your program should

1. Create an object from the ***SalesInvoice*** class.
2. Store the values entered by the user in your **SalesInvoice** object using the appropriate ***Properties***.
3. Using ***Properties*** and Methods of your **SalesInvoice** class, ***calculate*** the ***netSales*** value, the amount of ***stateTax*** and ***localTax*** due, and the ***grossSales*** value.
4. Display an ***invoice*** for the user, similar to the one shown below. You need to use a **MessageBox** to show the invoice as shown in Fig. 2.

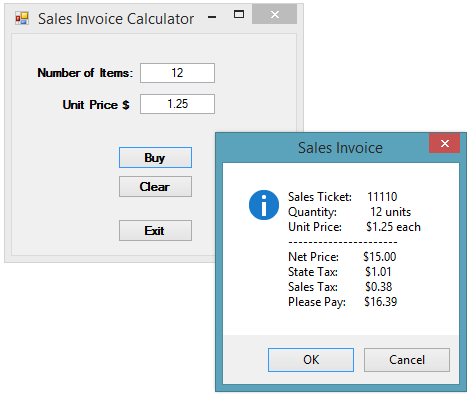


Figure 2 Sales Invoice Output

* The **Clear** Button will clear the **TextBoxes**.
* ***Pseudo-Code*** your design and then implement the code.
* Format and document your code in accordance with the course Style Guide.  Use the ***Grading Checklist*** below to insure you have included everything in your Project files. Submit your completed Project to Canvas.

**File(s) to Submit:**

Zip your entire Project folder and name the zip file  
Proj\_04\_your-initials\_V1.0.zip. For example, I would name my file Proj\_04\_DAF\_V1.0.zip. Submit this assignment as Project #04 to Canvas.

|  |  |  |
| --- | --- | --- |
|  | **Grading Checklist** |  |
| # | **Program** | **C** (correct)  **X** (incorrect) |
| 1 | Meets & works to specifications | 6 points |
| 2 | Error Free, elegant & efficient | 4 points |
| 3 | Pseudo-Code | -3 points |
| 4 | Style Guidelines | -2 points |
| 6 | Source Files(s) & Formatting | -2 points |
| 7 | Project Prolog | -1 points |
| 8 | Function Prologs | -1 points |
| 9 | Zip Filename | -1 points |
| 10 | Lab & Project Names | -1 points |
| 11 | Zip File is invalid or will not unzip | Lab = 0 pts |
|  | Total Points | 10 | 0-9 |

**Sample Program**

You can get an executable that runs correctly on Canvas.