**Artifact One Narrative**

Victor Landi

September 2023

**Artifact Description:**

Artifact Type: Source code from CS 260 Algorithms and Data Structures

Language: Originally written in C++, ported to Python as my enhancement.

Creation Date: The C++ version's copyright notice indicates it was created in 2017. The course CS 260 which the artifact is from was taken in February of 2022. The Python version is my enhancement, ported from the C++ code in September of 2023.

**Justification for Inclusion in ePortfolio:**

This artifact demonstrates proficiency in software development by showcasing my ability to design and implement a bid management program in two different programming languages (C++ and Python).

It highlights my ability to work with user input, data structures, object-oriented programming concepts, and menu-driven user interfaces.

The C++ and Python versions provide a valuable comparison of code in two popular programming languages, demonstrating adaptability and versatility in software development.

For these reasons I chose this artifact for the category of Software Design and Engineering.

**Specific Components Demonstrating Skills:**

In both versions, the creation and usage of a `Bid` data structure/class showcase object-oriented programming skills.

Input handling for user data (title, fund, vehicle, amount) demonstrates skills in user interaction and data validation.

The menu-driven interface showcases control flow and conditional logic skills.

String manipulation for formatting and cleaning the bid amount demonstrates data manipulation abilities.

**Pseudocode for the original program in C++**

// Define a data structure to hold bid information together as a single unit of storage.

struct Bid:

string ItemTitle

string Fund

string VehicleId

double BidAmount

// Display the bid information

function displayBid(bid):

Display "Title: " + bid.ItemTitle

Display "Fund: " + bid.Fund

Display "Vehicle: " + bid.VehicleId

Display "Bid Amount: " + bid.BidAmount as a string

// Prompt user for bid information and return it as a data structure

function getBid():

Create a Bid structure called SomeBid

Display "Enter title: "

Read a line from the input and assign it to SomeBid.ItemTitle

Display "Enter fund: "

Read a line from the input and assign it to SomeBid.Fund

Display "Enter vehicle: "

Read a line from the input and assign it to SomeBid.VehicleId

Display "Enter amount: "

Read a line from the input as a string and assign it to strAmount

Remove any '$' characters from strAmount

Convert strAmount to a double and assign it to SomeBid.BidAmount

Return SomeBid

// Main function

function main():

Create a Bid structure called TheBid

Initialize choice as 0

While choice is not equal to 9:

Display "Menu:"

Display " 1. Enter Bid"

Display " 2. Display Bid"

Display " 9. Exit"

Display "Enter choice: "

Read an integer from the input and assign it to choice

Switch on choice:

Case 1:

TheBid = getBid()

Break

Case 2:

displayBid(TheBid)

Break

Display "Good bye."

**Artifact Improvement:**

The Python version is an adaptation of the C++ version, offering a similar functionality in a different language. It simplifies some aspects, such as input handling.

**Key Differences from C++:**

Class Definition: In Python, a class is defined using the class keyword. The Bid class encapsulates bid information with attributes (fields) and a constructor (\_\_init\_\_ method).

User Input: Python's input() function is used to obtain user input as strings. In C++, cin is used for input.

String Formatting: Python uses string formatting with the format() method to format the bid amount with two decimal places.

Infinite Loop: Python uses a while True loop for the menu, which continues until the user chooses to exit. In C++, a while (choice != 9) loop was used.

Class Instances: In Python, instances of the Bid class are created using Bid().

Attribute Access: Python accesses class attributes using dot notation (e.g., bid.item\_title) instead of bid.ItemTitle in C++.

Type Conversion: Python uses float() for converting the bid amount from a string to a float, and str.strip() to remove leading and trailing whitespace. In C++, atof and erase were used.

Menu Handling: Python uses if-elif-else for menu choice handling instead of a switch-case statement in C++.

**Here is pseudo code for the enhancement, the program written in Python:**

# Define a class to hold bid information

class Bid:

Initialize item\_title attribute

Initialize fund attribute

Initialize vehicle\_id attribute

Initialize bid\_amount attribute

# Function to display bid information

function display\_bid(bid):

Format bid.bid\_amount with two decimal places

Display "Title:", bid.item\_title

Display "Fund:", bid.fund

Display "Vehicle:", bid.vehicle\_id

Display "Bid Amount:", bid\_amount\_str

# Function to get bid information from the user

function get\_bid():

Create an instance of the Bid class and assign it to bid

Get user input for bid.item\_title

Get user input for bid.fund

Get user input for bid.vehicle\_id

Get user input for bid\_amount\_str as a string

If bid\_amount\_str ends with "00", add another "0" to bid\_amount\_str

Convert bid\_amount\_str to a float and assign it to bid.bid\_amount

Return bid

# The main program

function main():

Initialize the\_bid as None

Loop indefinitely:

Display "Menu:"

Display " 1. Enter Bid"

Display " 2. Display Bid"

Display " 9. Exit"

Get user choice as a string

If choice is '1':

Call the get\_bid function to get bid information and assign it to the\_bid

Else if choice is '2':

If the\_bid is not None:

Call the display\_bid function to display bid information

Else:

Display "No bid entered."

Else if choice is '9':

Display "Good bye."

Break out of the loop

Else:

Display "Invalid choice. Please enter a valid option."

If this script is run as the main program:

Call the main function

**Meeting Course Objectives:**

The enhancement of Artifact One meets the course objectives related to demonstrating proficiency in programming languages, data structures, and user interaction. It also highlights adaptability by showing the same program in two languages. By doing so I employed strategies for building collaborative environments that enable diverse audiences to support organizational decision making in the field of computer science.

**Reflection on Enhancement:**

While creating and improving this artifact, I learned:

How to translate code from one programming language to another, which is a valuable skill in a diverse software development environment.

The importance of user-friendly input handling and data validation for a smooth user experience.

Differences and similarities between C++ and Python in terms of syntax and idiomatic coding practices.

Challenges I faced:

Adapting code from one language to another can be tricky, especially when dealing with language-specific nuances.

Ensuring consistent behavior and functionality between the C++ and Python versions could have been challenging; for these examples, I encountered no difficulties.

Managing user input and data validation effectively is crucial but can be complex, especially when dealing with different data types.

Overall, this artifact demonstrates my ability to work with different programming languages and showcases skills in software development, making it a valuable addition to my ePortfolio. From my experience being able to be versatile is vital in any career, especially in computer science. If I worked in a restaurant and my specialty was a burger made exclusively from beef, I would be successful. However, not everyone would choose to eat beef, I would be much more marketable and successful if I also offered alternatives. A menu that offered a turkey burger, a veggie burger in addition to the beef burger would do better overall, appealing to a wide audience and display my culinary skillset was broader. I believe that being versed in more than one computer language also shows this skillset.