CIS-11 Bubble Sort Project Documentation

Team Stackers
Zayden Middleton
Bienvenido Palma Jr
Chris Shey Caponpon Enriquez
LC3 Bubble Sort
May 25, 2025

Advisor: Kasey Nguyen, PhD

Part I – Application Overview

Objectives

This project implements the Bubble Sort algorithm in LC-3 Assembly. The main objective of this project is to apply core LC-3 concepts, such as branching, looping, management of the stack, subroutines, and input handling, into a practical program. This project will test our knowledge and implementation of these fundamental coding practices.

The program will support a number of actions, the main ones being as follows:

- User input of up to 8 integers ranging from 00 - 99
- Memory storage of the input in the form of an array
- Sorting the array using the Bubble Sort algorithm
- Displaying the sorted array in the console.

Business Process

This project simulates a real-world scenario where numerical data needs to be sorted. Users can enter values into the console, and the program takes these inputs and processes them with Bubble Sort, outputting the sorted list. Sorting data is a key part in many programming projects, and higher-level code gets compiled into Assembly, so a direct Assembly implementation mirrors these real-world requirements at a low level.

This project aims to help sort small data sets of integers in a very efficient manner. In a business context, where large amounts of small data sets must be sorted, this program can enhance operations efficiency, thereby reducing time and financial expenditure.

This project is being done now to test and improve our understanding of LC-3 concepts. In a business context, if this program is done later, then there is no problem, as there are other sorting algorithms that can be used instead of bubble sort that provide greater efficiency in sorting a list of integers of this size. In both the business context and this class, if we do not do this project, then the list of eight integers will not be sorted, and the task of sorting the list will not be completed.

Both businesses and the common person can benefit from this type of program. While this specific program may need to be modified to fit into a business, any individual can use this specific program to sort a list of eight integers in ascending order. The people who use this program will not consider this the most important improvement that can possibly be made at the time because there are more efficient alternatives to the Bubble Sort Algorithm.

This specific program will not be used by a business.

User Roles and Responsibilities

Our project team consists of three members, each with crucial roles in contributing to different parts of the development process. Our tasks involve designing, programming, testing, version control, documentation, etc. Everyone worked on the documentation.

- **Zayden Middleton** is responsible for managing the GitHub repository, which allows us to all work on the code together. His tasks involve creating branches, committing changes, and helping solve version control issues. Zayden is also in charge of creating flowcharts to help map out the logic of our project. Zayden is also responsible for a portion of the documentation and the programming.
- **Bienvenido Palma Jr** is responsible for writing a portion of the documentation for the project and will be testing the system by running the program, finding bugs, and suggesting improvements. Bienvenido will also work on a portion of the GitHub repository.

- **Chris Shey Caponpon Enriquez** is in charge of designing the program's layout and working on a portion of the documentation. While also testing to check if the program is working efficiently and meets all the design and functionality requirements. Chris will work on a portion of the GitHub repository.

Production Rollout Considerations

This application will primarily be tested against the provided test dataset specified in the program requirements. Testing will mostly be concerned with the correct outputs of the program, not the speed or complexity of the program. Because the program operates with LC-3 in mind, we can assume that the program will be run locally and there will be no issues with external hardware or integration with other systems. The program will be uploaded to Canvas for the instructor to download and test the provided test case.

Terminology

- **LC-3** - Stands for Little Computer 3. A simple computer architecture, mostly used to teach Assembly programming.
- **Bubble Sort** - An iterative sorting method that steps through a list and compares and swaps items as it goes.
- **Subroutine** - A block of instructions that performs a specific task and can be called upon by other parts of the program.
- **Stack** - Memory area for temporarily storing register values in conjunction with subroutine calls.
- **Assembly** - A low-level classification of programming language with a strong correspondence between language instructions and architectural instructions.
- **Register** - A small amount of fast storage used for loading and storing temporary data.
- **Push** - Method that adds to the top of a stack
- **Pop** - Method that removes from the top of a stack
- **Bit** - The smallest unit of information in a computer, represented by either a 0 (off) or a 1 (on).
- **Word** - In LC-3, a 16-bit long value

- **Array** - In LC-3, a contiguous series of blocks of memory where each location stores a word
- **Conditional Branching** - Allows conditional logical operations in code by taking a condition (in LC-3, either negative, positive, or zero) and jumps to another part of code.

Part II – Functional Requirement

Statement of Functionality

The program will prompt the user to enter eight numeric values, store these values in a predefined memory region, and sort the values into ascending order with Bubble Sort. It will then display the sorted list of numbers in the console.

List of functions:

- Memory management to hold the eight numerical inputs
- Bubble Sort implementation that uses nested loops and conditional branches for comparisons and swaps as required by the algorithm
- Subroutines to handle input reading, sorting, output display, and certain computational, iterative, or stack functions
- Stack operations to save and restore register states during subroutine calls
- ASCII conversion to transform the numerical data into characters outputted to the console and vice versa (inputs will have to be converted from ASCII to numbers)
- Prompt the user to input data, and a clear display of the sorted numbers afterward

Scope

The first phase of our development focuses on planning the basic foundation of our program. Through initial planning, we first create flowcharts and diagrams that illustrate the Bubble Sort process and logic flow. We would also set up a GitHub repository and begin writing the sorting code. While writing basic documentation explaining the purpose of our program and breaking down how our system operates. The next phase is focused on testing and improving our code. Testing our program with different inputs to fix any bugs or issues that may come up with edge cases. This process will be working towards improving the program's usability and ability to handle different inputs without problems. Then finalizing our project, making sure it reaches all project requirements, then submitting.

Performance

The program should respond quickly to user input. If multiple prompts are required, each prompt should be within a couple of seconds at MOST (although realistically it should be much faster). The sorting and outputting of the data should also be relatively quick since the data set is so small. Regardless of the efficiency of the code, performance should not be a bottleneck in this program. The largest amount of time when executing the program should be from the user in inputting the numbers.

Usability

The program should clearly prompt the user for a numerical input and provide immediate visual feedback after each entry. The final output of the sorted values will be in a neat and concise manner that allows the user to immediately know that the data has been sorted.

Documenting Requests for Enhancements

Date	Enhancement	Requested by	Notes	Priority	Release No/ Status
2025-05-29	Implement bubble sort algorithm	Bienvenido Palma Jr	Sort the array of 8 integers	High	Done
2025-05-29	Output array onto console	Zayden Middleton	Outputs the sorted array onto console	High	Done

Part III – Appendices

Flow chart: