# CIS11 Course Project Part 1: Documenting the Project

**Team Name: 96 Bull’s**

**Introduction**

* 1. **Purpose**

We chose to work on a Bubble Sorting program that would be able to get 8 number from the user ranging from 0 – 100 and display the sorted values in ascending order in console.

**The program must fulfill the following criteria:**

1. Contain appropriate addresses: origination, fill, array, input and output.
2. Display sorted values in console.
3. Use appropriate labels and comments.
4. Contain appropriate instructions for arithmetic, data movement and conditional operations.
5. Comprise of 2 or more subroutines and implement subroutine calls.
6. Use branching for control: conditional and iterative.
7. Manage overflow and storage allocation.
8. Manage stack: include PUSH-POP operation on stack.
9. Include save-restore operations.
10. Include pointer
11. Implement ASCII conversion operations
12. Use appropriate system call directives.
13. Testing.
    1. **Intended Audience and Users**

Any person in need of a sorting numbers by values that is also limited to an LC-3 system.

* 1. **Product Scope**

This program is used to sort an array in ascending order, and takes 8 different numbers from the user ranging from (0 – 100)

* 1. **Reference**

**Source Documents for the Program Requirements and Specification**

1. Contain appropriate addresses: origination, fill, array, input and output. (20 points)
2. Display sorted values in console. (20 points)
3. Use appropriate labels and comments. (20 points)
4. Contain appropriate instructions for arithmetic, data movement and conditional operations. (40 points)
5. Comprise of 2 or more subroutines and implement subroutine calls. (20 points)
6. Use branching for control: conditional and iterative. (30 points)
7. Manage overflow and storage allocation. (20 points)
8. Manage stack: include PUSH-POP operation on stack. (20 points)
9. Include save-restore operations. (30 points)
10. Include pointer (20 points)
11. Implement ASCII conversion operations (30 points)
12. Use appropriate system call directives. (10 point)
13. Testing (20 points): Test the program using the below values (green).

**2. Overall Description**

**2.1 Product Perspective**

This program is able to take values that are inputted by the user and sort them in ascending order.

* 1. **Product Functions**

**The overall description of functionality:**

1. The program takes values that a user inputs when prompted. During user input these values are stored in an array for later use.
2. The program utilizes 2 different loops to complete its task. The first loop counts for the number of objects in the array. The second loop does the comparison by subtracting the values and seeing if the total is less than or equal to 0.
3. The program makes sure to compare every object in the array.
4. The end result provides an accurate list of the items in ascending order.
   1. **User Classes and Characteristics**

Legacy Software Users: in desperate need to organize values of 8 numbers.

Developers: Elheas Arroyo-Desanti, Yassin Zaki, Josh Louie Dalisay.

* 1. **Operating Environment**

A system capable of running LC-3 and only LC-3.

* 1. **Design and Implementation Constraints**

If current hardware does not contain an LC-3 environment, access to web to download an LC-3 IDE may be necessary.

* 1. **Assumptions and Dependencies**

Users are required to be able to input values that want to be sorted. Keyboard is necessary to provide inputs. Users must be aware of desired inputs before accessing the program. The program can only operate in an LC-3 environment.

The user must be aware of how to assemble the LC-3 in order to create the object file necessary to run the program in the simulator.

***3*. External Interface Requirements**

* 1. **User Interfaces**

The user will interact with the program by opening the LC-3 simulate app and loading the (.obj) file. The program will then prompt a message asking the user to input 8 numbers ranging from (0-100) using their mouse and keyboard on the console. After the user is finished inputting the 8 numbers the numbers will display on the console in ascending order.

* 1. **Hardware Interfaces**

Windows PC

* 1. **Software Interfaces**

LC-3 Edit and LC-3 Simulate is required to run this program.

* 1. **Communications Interface**

Nothing

**4. Detailed Description of Functional requirements**

**4.1     Type of Requirement**

Purpose: Sort values entered by a user.

Inputs: Inputs are through the keyboard and mouse clicks.

Processing: The input is verified by checking if the value fits under the specified limits of the program.

Outputs: The correct input will result in the console displaying the user-inputted values in ascending order.

Data: User input stored into the program via pointer/array.

LOOP: Used to load the user’s values into the program and help the program understand if the correct amount of values were entered.

SORT: This function is the primary sorting function that compares the values of the numbers and places the values in ascending order from least to greatest.

**4.2 Performance requirements**

4.2.1 The application is anchored to systems capable of running LC-3

4.2.2 Since the application will be displaying decimal values in ascending order, the performance time should be no longer than a single second.

4.2.3 The amount of values able to be stored should be scalable, meaning that the amount of user inputs should be able to be increased depending on user needs.

4.2.4 Error handling should be implemented, and the application should be able to handle all run time errors.

**4.3 Flow Chart and Pseudocode.**

|  |
| --- |
|  |
|  |  |

|  |
| --- |
|  |
|  |  |

START

.ORIG x3000

PUSH/POP

Perform Swap of values in Bubble sort

Load array/Display bubble sort result on console

Jump to sorting algorithm

Load MESG to User/Display User input to console

Initialize Pointer

Initialize I/O Counter

Start of LOOP