PSEUDOCODE FOR BUBBLE SORT ALGORITHM PROGRAM

```
//Prompt user to input 8 numbers ranging from 0 - 100
if          ValueX > 100
            End
Else if ValueX < 0
            AGAIN           //Prompt user again
else if
ValueX >= 0 && ValueX <= 100  then
                                              //obtain all 8 values
Value1
Value2
Value3
Value4
Value5
Value6
Value7
Value8

if (Value4)  > (Value5)
            //swap Value4 and Value5 stay in same position
            JMP
Else
            //Value4 and Value5 switch in position
If (Value4) > (Value3)
            //swap Value4 's position with Value3's position
            JMP
Else
            //Value4 and Value3 stay in same position
If (Value4) >(Value2)
            //swap Value4 's position with Value2's position
            JMP
Else
            //Value4 and Value2 stay in same position
If (Value4) >(Value1)
            //swap Value4 's position with Value1's position
            JMP
Else
            //Value4 and Value1 stay in same position
If (Value4) >(Value6)
            //swap Value4 's position with Value6's position
            JMP
Else
```

```
    //Value4 and Value6 switch in position
If (Value4) >(Value7)
    //Value4 stays in same position as well as Value7
    JMP
Else
    //Value4 and Value7 switch in position
If (Value4) >(Value8)
    //Value4 and Value8 stay in same position
    JMP
Else
    //Value4 and Value8 switch in position
//Continue to do so for all 8 numbers to ensure they are placed in ascending order
    //Fill labels to addresses
//Load and store numbers to register
    //Display ending prompt that inputted numbers are in ascending order
//Output 8 values entered in ascending order
```

ALGORITHM/PSEUDOCODE FOR BUBBLE SORT ALGORITHM PROGRAM

1. Prompt user 8 times, to “input a number ranging from 0 - 100”
- Set up a string to prompt user using a label for ASCEND
- The label ASCEND holds an array of numbers indicating which will be first and last in order to be in ascending order
- Load the string label to prompt ascending numbers in console

2. Record user input

   -Use ASCII offset with decimal value #48 = 0. In other words, Add negative HEX30(HEXN30) to get integer and place in a subroutine

   -If inputted value is greater than 100 or less than 0, exit program

   -If value is in appropriate range, move the integer to designated register, and get stack pointer

   -Continue having user input all 8 values, then convert each to an integer

   -using subroutines for PUSH-POP, store the value to stack, save stack pointer and restore registers from memory

   -The stored values should be placed in a number array and using binary search, organize the 8 numbers into ascending order.

3. Input Validation
   - For any number greater than 100 or less than 0, program will exit
   - Branch for any number larger than 100, program will exit
   - To check if values are larger, add #-100(decimal negative 100) to register

4. Have user input 8 values ranging from 0 -100, display numbers in ascending order
   - Using iterative branching to access ASCEND array
   - Initialize start point value = 0
   - Add #-1 to register to create offset 1(decrement counter), then update value for incrementing pointer

- Add #48 or HEX30 to integer, in order to obtain integer character
- Load the address of the MESSAGE string
- USe PUTS (TRAP 22) to output string: “The 8 numbers in ascending order is: ”
- Present output of the 8 inputted numbers in ascending order

5. End program