CH2120

Class 11

# Main Program

**program** mainProgram

**implicit** **none**

**call** mainBubbleSort

**end** **program** mainProgram

# Bubble Sort: Main

**subroutine** mainBubbleSort

**implicit** **none**

**integer** :: numberOfElements

**real**, **dimension**(:), **allocatable** :: inputArray

**real**, **dimension**(:), **allocatable** :: sortedArray

**write**(\*,\*) "Enter the number of elements (real numbers) you wish to sort:"

**read**(\*,\*) numberOfElements

**allocate**(inputArray(numberOfElements))

**allocate**(sortedArray(numberOfElements))

**call** getArrayFromUser(numberOfElements, inputArray)

**write**(\*,\*) "Original list:"

**call** displayArray(inputArray, numberOfElements)

**call** performBubbleSortBasic(inputArray, sortedArray, numberOfElements)

**write**(\*,\*) "Sorted list using basic bubble sort."

**call** displayArray(sortedArray, numberOfElements)

**call** performBubbleSortEfficient1(inputArray, sortedArray, numberOfElements)

**write**(\*,\*) "Sorted list using efficient bubble sort (version 1)"

**call** displayArray(sortedArray, numberOfElements)

**call** performBubbleSortEfficient2(inputArray, sortedArray, numberOfElements)

**write**(\*,\*) "Sorted list using efficient bubble sort (version 2)"

**call** displayArray(sortedArray, numberOfElements)

**end** **subroutine** mainBubbleSort

# Bubble Sort: Basic

**subroutine** performBubbleSortBasic(inputArray, sortedArray, sizeOfArray)

**implicit** **none**

**integer**, **intent**(in) :: sizeOfArray

**real**, **dimension**(sizeOfArray), **intent**(in) :: inputArray

**real**, **dimension**(sizeOfArray), **intent**(out) :: sortedArray

**integer** passCounter, stepCounter

**integer** numberOfPasses, numberOfComparisons, numberOfSwaps

numberOfPasses = 0

numberOfComparisons = 0

numberOfSwaps = 0

sortedArray = inputArray

**do** passCounter = 1, (sizeOfArray - 1)

numberOfPasses = numberOfPasses + 1

**do** stepCounter = 1, (sizeOfArray - 1)

numberOfComparisons = numberOfComparisons + 1

**if**(sortedArray(stepCounter) > sortedArray(stepCounter + 1)) **then**

numberOfSwaps = numberOfSwaps + 1

**call** swapTwoElements(sortedArray, sizeOfArray, stepCounter, (stepCounter + 1))

**end** **if**

**end** **do**

**end** **do**

**write**(\*,\*) "Bubble Sort Basic--"

**write**(\*,10) "Number of passes = ", numberOfPasses

**write**(\*,10) "Number of comparisons = ", numberOfComparisons

**write**(\*,10) "Number of swaps = ", numberOfSwaps

**write**(\*,\*)

10 **format**(a24, i2)

**end** **subroutine** performBubbleSortBasic

# Bubble Sort: Efficient 1

Eliminate redundant comparisons.

**subroutine** performBubbleSortEfficient1(inputArray, sortedArray, sizeOfArray)

**implicit** **none**

**integer**, **intent**(in) :: sizeOfArray

**real**, **dimension**(sizeOfArray), **intent**(in) :: inputArray

**real**, **dimension**(sizeOfArray), **intent**(out) :: sortedArray

**integer** passCounter, stepCounter

**integer** numberOfPasses, numberOfComparisons, numberOfSwaps

numberOfPasses = 0

numberOfComparisons = 0

numberOfSwaps = 0

sortedArray = inputArray

**do** passCounter = 1, (sizeOfArray - 1)

numberOfPasses = numberOfPasses + 1

**do** stepCounter = 1, (sizeOfArray - passCounter)

numberOfComparisons = numberOfComparisons + 1

**if**(sortedArray(stepCounter) > sortedArray(stepCounter + 1)) **then**

numberOfSwaps = numberOfSwaps + 1

**call** swapTwoElements(sortedArray, sizeOfArray, stepCounter, (stepCounter + 1))

**end** **if**

**end** **do**

**end** **do**

**write**(\*,\*) "Bubble Sort Efficient (Version 1)--"

**write**(\*,10) "Number of passes = ", numberOfPasses

**write**(\*,10) "Number of comparisons = ", numberOfComparisons

**write**(\*,10) "Number of swaps = ", numberOfSwaps

**write**(\*,\*)

10 **format**(a24, i2)

**end** **subroutine** performBubbleSortEfficient1

# Bubble Sort: Efficient 2

Check if list is already sorted.

**subroutine** performBubbleSortEfficient2(inputArray, sortedArray, sizeOfArray)

**implicit** **none**

**integer**, **intent**(in) :: sizeOfArray

**real**, **dimension**(sizeOfArray), **intent**(in) :: inputArray

**real**, **dimension**(sizeOfArray), **intent**(out) :: sortedArray

**integer** passCounter, stepCounter

**integer** numberOfPasses, numberOfComparisons, numberOfSwaps

**logical** :: isSorted

numberOfPasses = 0

numberOfComparisons = 0

numberOfSwaps = 0

sortedArray = inputArray

**do** passCounter = 1, (sizeOfArray - 1)

numberOfPasses = numberOfPasses + 1

isSorted = .**true**.

**do** stepCounter = 1, (sizeOfArray - passCounter)

numberOfComparisons = numberOfComparisons + 1

**if**(sortedArray(stepCounter) > sortedArray(stepCounter + 1)) **then**

numberOfSwaps = numberOfSwaps + 1

**call** swapTwoElements(sortedArray, sizeOfArray, stepCounter, (stepCounter + 1))

isSorted = .**false**.

**end** **if**

**end** **do**

**if** (isSorted .**eqv**. .**true**.) **then**

**exit**

**end** **if**

**end** **do**

**write**(\*,\*) "Bubble Sort Efficient (Version 2)--"

**write**(\*,10) "Number of passes = ", numberOfPasses

**write**(\*,10) "Number of comparisons = ", numberOfComparisons

**write**(\*,10) "Number of swaps = ", numberOfSwaps

**write**(\*,\*)

10 **format**(a24, i2)

**end** **subroutine** performBubbleSortEfficient2

# Common Subroutines

## displayArray.f08

**subroutine** displayArray(inputArray, sizeOfArray)

**implicit** **none**

**integer**, **intent**(in) :: sizeOfArray

**real**, **dimension**(sizeOfArray), **intent**(in) :: inputArray

**integer** elementCounter

**do** elementCounter = 1, sizeOfArray

**write**(\*,10) inputArray(elementCounter)

**end** **do**

**write**(\*,\*) "------------------------"

10 **format**(f8.2)

**end** **subroutine** displayArray

## getArrayFromUser.f08

**subroutine** getArrayFromUser(sizeOfArray, userArray)

**implicit** **none**

**integer**, **intent**(in) :: sizeOfArray

**real**, **dimension**(sizeOfArray), **intent**(out) :: userArray

**integer** elementCounter

**do** elementCounter = 1, sizeOfArray

**write**(\*,10) "Enter element #", elementCounter

**read**(\*,\*) userArray(elementCounter)

**end** **do**

10 **format**(a15, i2)

**end** **subroutine** getArrayFromUser

## swapTwoElements.f08

**subroutine** swapTwoElements(inputArray, sizeOfArray, index1, index2)

**implicit** **none**

**integer**, **intent**(in) :: sizeOfArray, index1, index2

**real**, **dimension**(sizeOfArray), **intent**(inout) :: inputArray

**real** :: swapper

swapper = inputArray(index1)

inputArray(index1) = inputArray(index2)

inputArray(index2) = swapper

**end** **subroutine** swapTwoElements

## Output

Enter the number of elements (real numbers) you wish to sort:

6

Enter element # 1

-21.94

Enter element # 2

0.05

Enter element # 3

2.65

Enter element # 4

-4.18

Enter element # 5

3.19

Enter element # 6

19.63

Original list:

-21.94

0.05

2.65

-4.18

3.19

19.63

------------------------

Bubble Sort Basic--

Number of passes = 5

Number of comparisons = 25

Number of swaps = 2

Sorted list using basic bubble sort.

-21.94

-4.18

0.05

2.65

3.19

19.63

------------------------

Bubble Sort Efficient (Version 1)--

Number of passes = 5

Number of comparisons = 15

Number of swaps = 2

Sorted list using efficient bubble sort (version 1)

-21.94

-4.18

0.05

2.65

3.19

19.63

------------------------

Bubble Sort Efficient (Version 2)--

Number of passes = 3

Number of comparisons = 12

Number of swaps = 2

Sorted list using efficient bubble sort (version 2)

-21.94

-4.18

0.05

2.65

3.19

19.63

------------------------