Module main():

//Declare variables

//array size

Declare int size = 20

//array for list of names

Declare str names[size]

//the name to be searched

Declare str searchTerm

//the number of array lookups during binary search

Declare int lookUpnum

//function calls

//ask user for names

getName(names)

//sort the name array

bubbleSort(names, size)

//show sorted names

showNames(names)

//search the name array

search(names, searchTerm, lookUpnum, size)

End Module

//Module for getting list of names

Module getName(str ref names)

//reset counter

Set counter = 0

//ask user for names

While counter < 20

Display “Enter name:”

Input name[counter]

//increment counter

Set counter = counter + 1

End While

End Module

//show names after bubble sort

Module showNames(str names)

//reset counter

Set counter = 0

//show names

While counter < 20

Display name[counter]

//increment counter

Set counter = counter + 1

End while

End module

//Module for sorting the name array

Module bubblesort(str names, int size)

//this variable contains the subscript of the last element to

//compare

Declare int maxElement

//counter for the inner loop

Declare int index

//variable for local storage

Declare str temp

//the outer loop makes maxElement the last element to compare

//for each pass through the array. maxElement is initially the

//index of the last element in the array

For maxElement = size - 1 to 0 step – 1

//the inner loop steps through the array comparing each

//element with its neighbor

For index = 0 to maxElement – 1

If names[index] > array[index + 1] then

//if two elements are out of order, they swap

Set temp = name[index]

Set name[index] = name[index + 1]

Set name[index + 1] = temp

End If

End For

End For

End Module

Module search(str names, str ref searchTerm int ref lookUpnum, int size)

//variable to hold subscript of name

Declare int index

//variable to control the loop

Declare str again = “Y”

While (again == “Y” or again == “y”)

//ask name to search for

Display “Enter a name to search: “

Input searchTerm

//search for name and get number of array lookups

Index, lookUpnum = binSearch(names, searchTerm, size, lookUpnum)

If index != - 1 then

//show name if found

Display “Name Found”, names[index]

//show name position in array

Display “Position: “, index

//show number of array lookups

Display “Array lookups: “, lookUpnum

Else

//show that name was not found in the array

Display searchTerm, “ was not found”

//show number of array lookups

Display “Array lookups: “, lookUpnum

End If

//ask if user wants to search again

Display “Do you want to search again? (Y=Yes, N=No)”

Input again

End While

End module

Function int binSearch(str names, str name, int size,int ref lookUpnum)

//subscript of first element

Declare int first = 0

//subscript of last element

Declare int last = size – 1

//search value position

Declare int position = - 1

//flag

Declare boo found = False

//subscript of midpoint

Declare int middle

While (NOT found) AND (first <= last)

//calculate midpoint

Set middle = (first + last) / 2

//value at midpoint?

If names[middle] == searchTerm then

Set found = True

Set position = middle

//increment array lookup counter

lookUpnum = lookUPnum + 1

End if

//increment array lookup counter

lookUpnum = lookUPnum + 1

//value in lower half?

If names[middle] > searchTerm then

Set last = middle – 1

//increment array lookup counter

lookUpnum = lookUPnum + 1

End if

//increment array lookup counter

lookUpnum = lookUPnum + 1

// value in upper half?

if not (names[middle] == searchTerm or names[middle] > searchTerm) then

Set first = middle + 1

//increment array lookup counter

lookUpnum = lookUpnum + 1

End if

End While

//return number of array lookups and position of item or -1 if

//item was not found

Return position

End Function