DEBUG08-01

// This application reads 10 numbers and sorts them,

// then displays the three highest numbers

start

Declarations

num SIZE = 10

num number

num numbers[SIZE]

num count = 0

num temp // undeclared variable

getReady()

while count < SIZE // <= would allow for 11 numbers entered

detailLoop()

endwhile

finish()

stop

getReady()

output "Enter a number "

input number

return

detailLoop()

numbers[count] = number

output "Enter a number "

count = count + 1 // need to increment

return

finish()

sort()

output "The highest three are ", numbers[SIZE-1], numbers[SIZE-2], numbers[SIZE-3] //highest numbers are at the end of the array

return

sort()

num y = 0

num COMPS = count - 1

while y < COMPS

num x = 0 // initialize x as part of outer loop

while x < COMPS

if numbers[x+1] < numbers[x] then // compare to next element in array

swap()

endif

x = x + 1

endwhile

y = y + 1

endwhile

return

swap()

num temp

temp = numbers[x + 1]

numbers[x + 1] = numbers[x]

numbers[x] = temp

return

DEBUG08-02

// This application reads student typing test data

// including number of errors on the test, and the number

// of words typed per minute. Grades are assigned based

// on the following table:

// Errors

// Speed 0 1 2 or more

// 0-30 C D F

// 31-50 C C F

// 51-80 B C D

// 81-100 A B C

// 101 and up A A B

start

Declarations

num MAX\_ERRORS = 2

num errors

num wordsPerMinute

String grades[5][3] = {"C", "D", "F"},

{"C", "C", "F"},

{"B", "C", "D"},

{"A", "B", "C"},

{"A", "A", "B"} //string not num

num LIMITS = 5

num speedLimits[LIMITS] = 0, 31, 51, 81, 101

num row

output "Enter number of errors on the test "

input errors

if errors > MAX\_ERRORS then

errors = 2 // 2 is the most errors allowed

endif

output "Enter the speed in words per minute "

input wordsPerMinute // speed undeclared

row = 0

while row < LIMITS AND wordsPerMinute >= speedLimits[row] // evaluate based on position row

row = row + 1

endwhile

row = row - 1

output "Your grade is ", grades[wordsPerMinute][errors] // columns should be errors

stop

DEBUG08-03

// This application reads sales data for an automobile dealership.

// Up to 100 sale amounts can be entered. The entered sale amounts

// are sorted so that the median sale can be displayed.

start

Declarations

num SIZE = 100

num QUIT = 99999

num saleAmount

num sales[SIZE]

num count = 0

num middlePosition

num x

num y

num temp

num comps

getReady()

while saleAmount <> QUIT and count < SIZE

detailLoop()

endwhile

finish()

stop

getReady()

output "Enter sale amount "

input saleAmount

return

detailLoop()

sales[count] = saleAmount // need to populate the array

output "Enter sale amount "

input saleAmount

count = count + 1 // increment after so position 0 is filled

return

finish()

sort()

middlePosition = count / 2 // may not have entered 100 sales

output "The median sale amount is ", sales[middlePosition]

return

sort()

comps = count - 1

while y < comps

x = 0

while x < comps

if sales[x] > sales[x + 1] then

swap()

endif

endwhile

endwhile

return

swap() // function called swap

temp = sales[x + 1]

sales[x + 1] = sales[x]

sales[x] = temp // need to assign a value to sales[x]

return