After spending hours trying to get quicksort to work properly and asking my group to look at it to no avail, I decided to write this instead.

I tried to implement a recursive quicksort the same way I have for the last two languages (Java and FORTRAN). In both of those languages, it went off without a hitch. I believe a different method than what I implemented would be required to make it work in COBOL.

```
HEY
ENTER NUM 01: 3
ENTER NUM 02: 2
ENTER NUM 03: 1
ENTER NUM 04: 6
ENTER NUM 05: 5
ENTER NUM 06: 4
ENTER NUM 07: 9
ENTER NUM 08: 8
ENTER NUM 09: 7
ENTER NUM 10: 0

Presorted Array:
003
002
001
006
005
004
009
008
007
000
PARTITION CALL
PARTITION RETURN
QUICKSORTFUNC CALL 1
PARTITION CALL
PARTITION RETURN
QUICKSORTFUNC CALL 1
PARTITION CALL

attempt to reference unallocated memory (signal SIGSEGV)
abnormal termination - file contents may be incorrect
```

Quicksort output

Here is the final program's output. As you can see, the Array input works, it is then printed back out, and it moves on to the **quicksortfunc** subroutine where **partition** is called and returned. After this, **quicksortfunc** is recursively called two more times, but the program exits out on the third time.

I believed this to be the way that COBOL passed values by reference, but even after playing around with the BY CONTENT and BY VALUE keywords to send data to the subroutine more as Java would, I had no success. I believe the answer to fixing the program to lie somewhere in here, but I can't find it.

```
25  PROCEDURE DIVISION USING LIST, SORTVARS, PARTITIONVARS, SWAPVARS, RETURNVARS.
26          IF LAST1 > FIRST1 THEN
27          *> Get PIVOTINDEX for current iteration.
28              CALL 'partition' USING LIST, BY VALUE SORTVARS, BY VALUE PARTITIONVARS, BY VALUE SWAPVARS, RETURNVARS
29
30              SUBTRACT 1 FROM PIVOTINDEX
31              CALL 'quicksortfunc' USING LIST, BY VALUE SORTVARS, BY VALUE SWAPVARS, RETURNVARS
32
33              ADD 2 TO PIVOTINDEX
34              CALL 'quicksortfunc' USING LIST, BY VALUE SORTVARS, BY VALUE SWAPVARS, RETURNVARS
35
36              SUBTRACT 1 FROM PIVOTINDEX
37          END-IF.
38  EXIT PROGRAM.
```

*One of my many attempted remedies for this section of **quicksortfunc.cob***

In the end, I truly wasn't too bummed about not finished the program. I felt like I had learned the language enough to write simple programs and read other programs.

Proper Quicksort Implementation.

I also found this implementation of a COBOL quicksort that would work, but I didn't want to just copy straight from the author.