Computing Machinery I
Assignment 3

Sorting One-Dimensional Arrays

Create an ARMv8 assembly language program that implements the following algorithm:

```
#define  SIZE 50

int main()
{
  int v[SIZE], i, j, min, temp;

  /*  Initialize array to random positive integers, mod 256  */
  for (i = 0; i < SIZE; i++) {
    v[i] = rand() & 0xFF;
    printf("v[%d]: %d\n", i, v[i]);
  }

  /*  Sort the array using a selection sort  */
  for (i = 0; i < SIZE - 1; i++) {
    /*  Find the least element in the right subarray  */
    min = i;
    for (j = i + 1; j < SIZE; j++) {
      /*  Compare elements  */
      if (v[j] < v[min])
        min = j;
    }

    /*  Swap elements  */
    temp = v[min];
    v[min] = v[i];
    v[i] = temp;
  }

  /*  Print out the sorted array  */
  printf("\nSorted array:\n");
  for (i = 0; i < SIZE; i++)
    printf("v[%d]: %d\n", i, v[i]);

  return 0;
}
```

Create space on the stack to store all local variables, using the techniques described in lectures. Use *m4* macros or assembler equates for all stack variable offsets. Optimize your code so that it uses as few instructions as possible. Be sure, however, that you always read or write memory when using or assigning to the local variables. Name the program *assign3.asm*.

Also run the program in *gdb*, first displaying the contents of the array before sorting, and then displaying it again once the sort is complete (use the *x* command to examine memory). You should show that the algorithm is working as expected. Capture the *gdb* session using the *script* UNIX command, and name the output file *script.txt*.

Other Requirements

Make sure your code is readable and fully documented, including identifying information at the top of each file. You must comment each line of assembly code. Your code should also be well designed: make sure it is well organized, clear, and concise.

New Skills Needed for this Assignment:

- Use of the stack to store local variables
- Addressing stack variables
- Use of *m4* macros or assembler equates for stack variable offsets
- Storing and addressing one-dimensional arrays on the stack

Submit the following:

1. Your assembly source code files for the program, and your script output file. Use the *Assignment 3* Dropbox Folder in D2L to submit electronically. The TA will assemble and run your program to test it.

Be sure to name your program and script file as described above.

Computing Machinery I
Assignment 3 Grading

Student:_____

Functionality			
Loop to initialize array	2	_____	
Display of unsorted array	2	_____	
Outer loop of sort	2	_____	
Inner loop of sort	2	_____	
Comparison of array elements	2	_____	
Exchange of array elements	2	_____	
Display of sorted array	2	_____	
Use of macros/equates for stack variable offsets	4	_____	
Optimization	4	_____	
Script showing *gdb* session	2	_____	
Complete documentation and commenting	4	_____	
Formatting (use of columns and white space)	4	_____	
Design quality	2	_____	
Total	**34**	**_____**	**_____%**