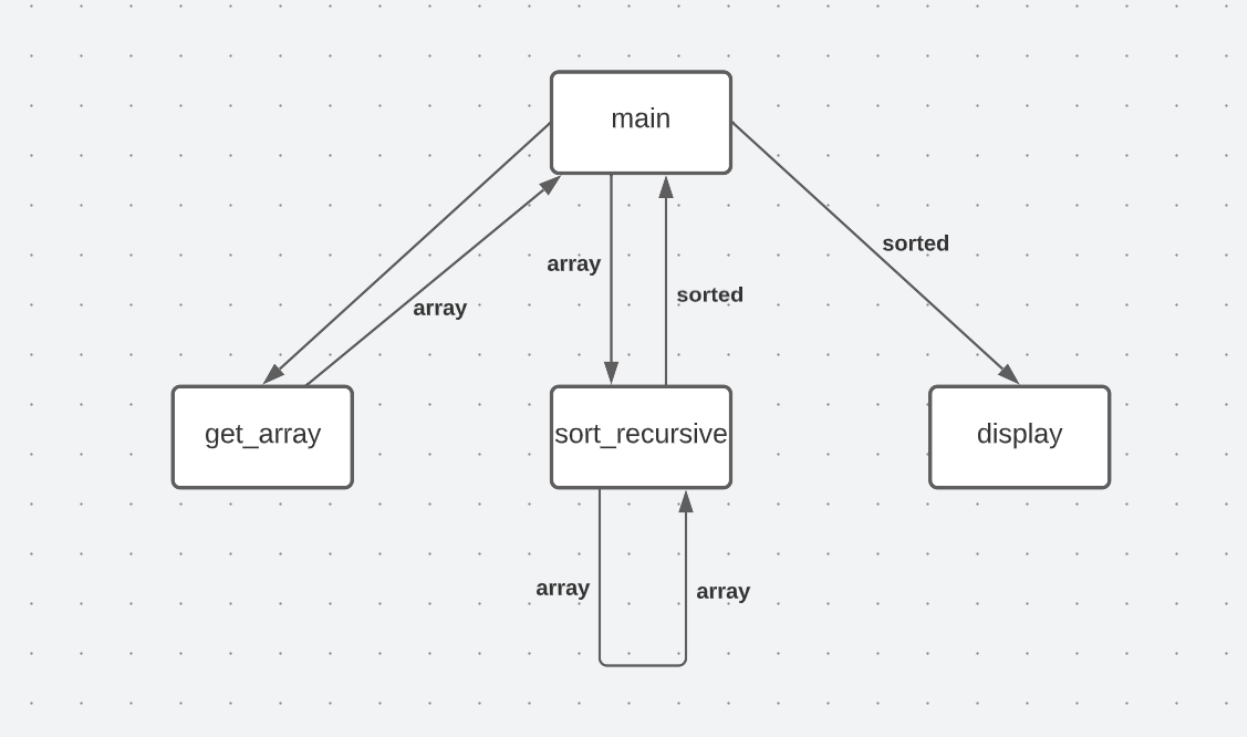
Braeden Pope

Modularization Design

Segregation Sort Design

Structure Chart



Segregation Sort Psuedocode

```
sort_recursive(array, i_start, i_end):
        i_pivot = i_end // 2
        i_up = i_start
        i_down = i_end
        start_found = false
        end_found = false
        IF (i_start + 1 == i_end - 1):
                IF (array[i_start] > array[i_end]):
                        array[i_start], array[i_end] = array[i_end], array[i_start]
                        RETURN array
        IF (i_end – i_start == 1):
```

```
        array[i_start], array[i_end] = array[i_end], array[i_start]
        RETURN array
IF (i_pivot == 0 or i_end == i_start):
        RETURN array
WHILE (NOT end_found and not start_found):
        WHILE (i_up <= i_pivot AND NOT start_found):
                IF (array[i_up] > array[i_pivot]):
                        start_found = true
                ELIF (i_up + 1 == i_pivot):
                        i_up += 1
                        start_found = TRUE
                ELSE:
                        i_up += 1



        WHILE (i_down > i_pivot AND NOT end_found)
                IF (array[i_down] < array[i_pivot]):
                        end_found = true
                ELSE:
                        i_down -= 1
        IF (start_found and end_found or i_down == i_pivot):
                IF (i_up == i_pivot):
                        i_pivot = i_down
                        start_found = TRUE
                        end_found = TRUE
                ELSE:
                        start_found = FALSE
                        end_found = FALSE
                array[i_up], array[i_down] = array[i_down], array[i_up]
front = sort_recursive(array, i_start, i_up)
back = sort_recursive(front, i_pivot + 1, i_end)
RETURN back
```