

Lab 8 Report ARM

Bubble sort is a simple sorting algorithm that repeatedly steps through the list of elements to be sorted, compares each adjacent pair of elements, and swaps them if they are in the wrong order. The algorithm gets its name from the way smaller elements "bubble" to the top of the list as it is being sorted. This is implemented using the following steps:

1. Start at the beginning of the list of elements to be sorted
2. Compare the first two elements. If the first element is greater than the second element, swap them
3. Move to the next pair of elements, and compare them in the same way. Continue this process until the end of the list is reached
4. At this point, the largest element will be at the end of the list
5. Repeat steps 1-4 for the remaining unsorted portion of the list (i.e., everything except the last element)
6. Continue this process until the entire list is sorted

Bubble sort has a time complexity of

$$O(n^2)$$

ARM implementation

The assembly program used the following data section:

```
.data
    array:          .word -123, 548, 923, 431, 560, -348
    endarr:
```

Before

We observe the array values in their initial order

The debugger window is titled "Stopped". The "Registers" pane on the left shows the following values:

Register	Value
r0	0
r1	0
r2	0
r3	0
r4	0
r5	0
r6	0
r7	0
r8	0
r9	0
r10	0
r11	0
r12	0
sp	0
lr	0
pc	100
cpsr	467 NZCVI SVC
spsr	0 NZCVI ?

The "Memory (Ctrl-M)" pane on the right shows the contents of memory starting at address 0. The "array" variable is located at address 0x100, and its initial values are shown in the memory dump.

Address	Memory contents and ASCII
00000000	fffffb0 -1431655766 -1431655766 -1431655766 -1431655766
00000001	fffffcd0 -1431655766 -1431655766 -1431655766 -1431655766
00000002	fffffd00 -1431655766 -1431655766 -1431655766 -1431655766
00000003	fffffe00 -1431655766 -1431655766 -1431655766 -1431655766
00000004	fffff000 -1431655766 -1431655766 -1431655766 -1431655766
00000005	00000000 -382926352 -476045312 -476041216 -499040255
00000006	00000010 -369098737 -532393982 -476041216 -369098743
00000007	00000020 -509574909 -409968632 -494370812 -409964536
00000008	00000030 -514457593 -637534206 -411017208 -498565116
00000009	00000040 -411013112 -494718975 -514654203 -1157627917
0000000a	00000050 -494788607 -514719740 -1157627923 -390266384
0000000b	00000060 -516948194 -442564592 -442560496 -532606976
0000000c	00000070 -509603519 -335544351 -352321538 136
0000000d	00000080 160 0 -123 548
0000000e	00000090 923 431 560 -348
0000000f	000000a0 -1431655766 -1431655766 -1431655766 -1431655766
00000010	000000b0 -1431655766 -1431655766 -1431655766 -1431655766
00000011	000000c0 -1431655766 -1431655766 -1431655766 -1431655766
00000012	000000d0 -1431655766 -1431655766 -1431655766 -1431655766
00000013	000000e0 -1431655766 -1431655766 -1431655766 -1431655766
00000014	000000f0 -1431655766 -1431655766 -1431655766 -1431655766
00000015	00000100 -1431655766 -1431655766 -1431655766 -1431655766
00000016	00000110 -1431655766 -1431655766 -1431655766 -1431655766
00000017	00000120 -1431655766 -1431655766 -1431655766 -1431655766
00000018	00000130 -1431655766 -1431655766 -1431655766 -1431655766
00000019	00000140 -1431655766 -1431655766 -1431655766 -1431655766
00000150	00000150 -1431655766 -1431655766 -1431655766 -1431655766
00000160	00000160 -1431655766 -1431655766 -1431655766 -1431655766
00000170	00000170 -1431655766 -1431655766 -1431655766 -1431655766
00000180	00000180 -1431655766 -1431655766 -1431655766 -1431655766
00000190	00000190 -1431655766 -1431655766 -1431655766 -1431655766

After

We observe the array values in their sorted order

The debugger window is titled "Running". The "Registers" pane on the left shows the following values:

Register	Value
r0	136
r1	6
r2	5
r3	1
r4	0
r5	0
r6	0
r7	0
r8	0
r9	0
r10	0
r11	0
r12	0
sp	0
lr	120
pc	120
cpsr	1610613203 NZCVI SVC
spsr	0 NZCVI ?

The "Memory (Ctrl-M)" pane on the right shows the contents of memory starting at address 0. The "array" variable is located at address 0x100, and its sorted values are shown in the memory dump.

Address	Memory contents and ASCII
00000000	fffffb0 -1431655766 -1431655766 -1431655766 -1431655766
00000001	fffffcd0 -1431655766 -1431655766 -1431655766 -1431655766
00000002	fffffd00 -1431655766 -1431655766 -1431655766 -1431655766
00000003	fffffe00 -1431655766 -1431655766 -1431655766 -1431655766
00000004	fffff000 0 0 0 0
00000005	00000000 -382926352 -476045312 -476041216 -499040255
00000006	00000010 -369098737 -532393982 -476041216 -369098743
00000007	00000020 -509574909 -409968632 -494370812 -409964536
00000008	00000030 -514457593 -637534206 -411017208 -498565116
00000009	00000040 -411013112 -494718975 -514654203 -1157627917
0000000a	00000050 -494788607 -514719740 -1157627923 -390266384
0000000b	00000060 -516948194 -442564592 -442560496 -532606976
0000000c	00000070 -509603519 -335544351 -352321538 136
0000000d	00000080 160 0 -348 -123
0000000e	00000090 431 548 560 923
0000000f	000000a0 -1431655766 -1431655766 -1431655766 -1431655766
00000010	000000b0 -1431655766 -1431655766 -1431655766 -1431655766
00000011	000000c0 -1431655766 -1431655766 -1431655766 -1431655766
00000012	000000d0 -1431655766 -1431655766 -1431655766 -1431655766
00000013	000000e0 -1431655766 -1431655766 -1431655766 -1431655766
00000014	000000f0 -1431655766 -1431655766 -1431655766 -1431655766
00000015	00000100 -1431655766 -1431655766 -1431655766 -1431655766
00000016	00000110 -1431655766 -1431655766 -1431655766 -1431655766
00000017	00000120 -1431655766 -1431655766 -1431655766 -1431655766
00000018	00000130 -1431655766 -1431655766 -1431655766 -1431655766
00000019	00000140 -1431655766 -1431655766 -1431655766 -1431655766
00000150	00000150 -1431655766 -1431655766 -1431655766 -1431655766
00000160	00000160 -1431655766 -1431655766 -1431655766 -1431655766
00000170	00000170 -1431655766 -1431655766 -1431655766 -1431655766
00000180	00000180 -1431655766 -1431655766 -1431655766 -1431655766
00000190	00000190 -1431655766 -1431655766 -1431655766 -1431655766