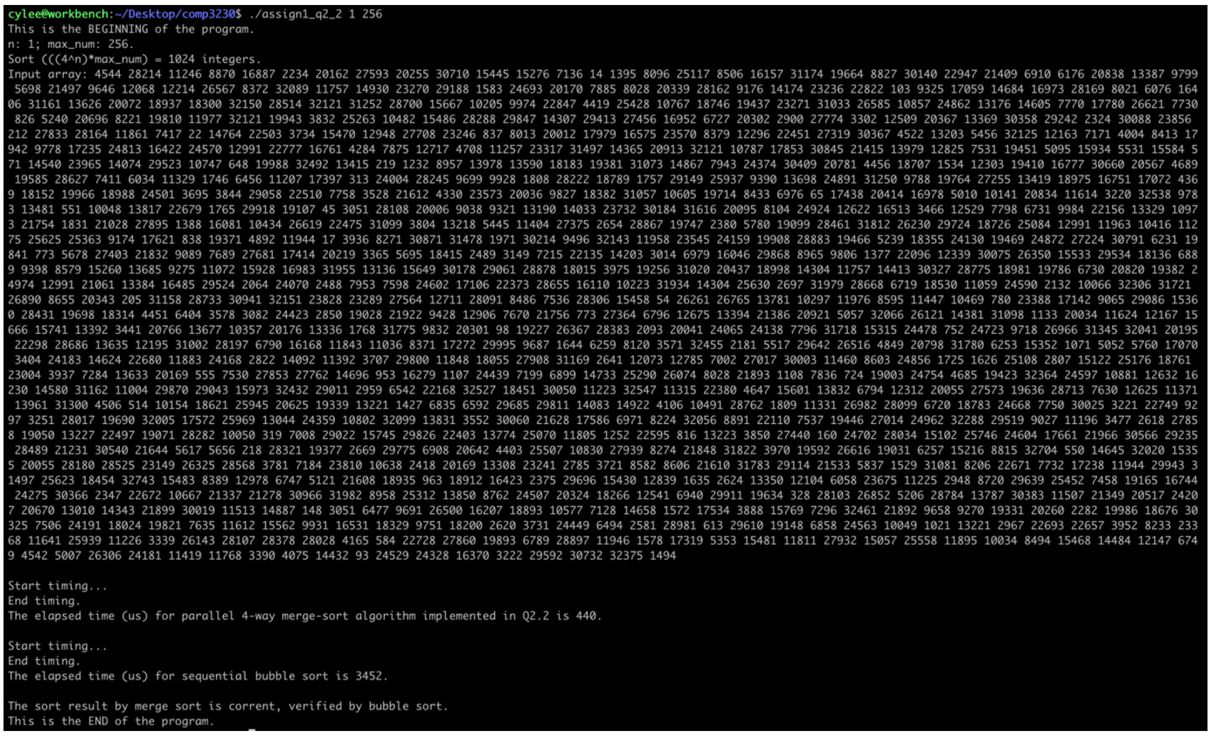
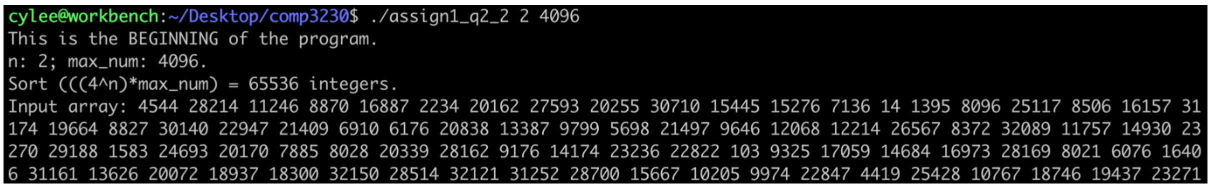


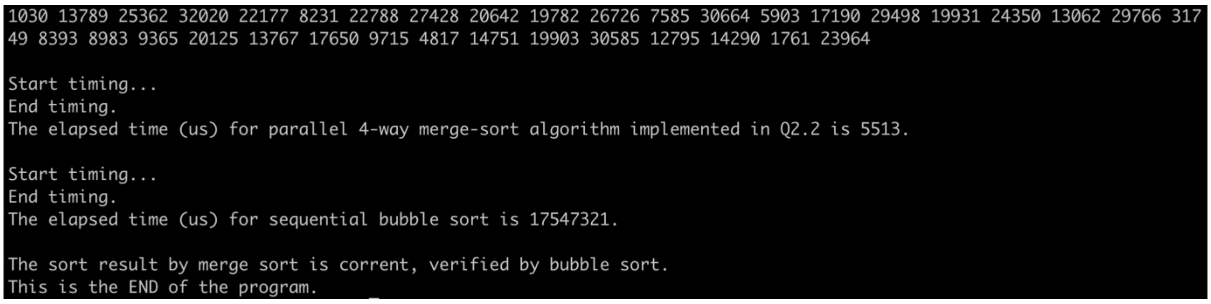
```
cylee@workbench:~/Desktop/comp3230$ ./assign1_q2_2 1 256
This is the BEGINNING of the program.
n: 1; max_num: 256.
Sort (((4^n)*max_num) = 1024 integers.
Input array: 4544 28214 11246 8870 16887 2234 20162 27593 20255 30710 15445 15276 7136 14 1395 8096 25117 8506 16157 31174 19664 8827 30140 22947 21409 6910 6176 20838 13387 9799
 5698 21497 9646 12068 12214 26567 8372 32089 11757 14930 23270 29188 1583 24693 20170 7885 8028 20339 28162 9176 14174 23236 22822 103 9325 17059 14684 16973 28169 8021 6076 164
06 31161 13626 20072 18937 18300 32150 28514 32121 31252 28700 15667 10205 9974 22847 4419 25428 10767 18746 19437 23271 31033 26585 10857 24862 13176 14605 7770 17780 26621 7730
 826 5240 20696 8221 19810 11977 32121 19943 3832 25263 10482 15486 28288 29847 14307 29413 27456 16952 6727 20302 2900 27774 3302 12509 20367 13369 30358 29242 2324 30088 23856
212 27833 28164 11861 7417 22 14764 22503 3734 15470 12948 27708 23246 837 8013 20012 17979 16575 23570 8379 12296 22451 27319 30367 4522 13203 5456 32125 12163 7171 4004 8413 17
942 9778 17235 24813 16422 24570 12991 22777 16761 4284 7875 12717 4708 11257 23317 31497 14365 20913 32121 10787 17853 30845 21415 13979 12825 7531 19451 5095 15934 5531 15584 5
71 14540 23965 14074 29523 10747 648 19988 32492 13415 219 1232 8957 13978 13590 18183 19381 31073 14867 7943 24374 30409 20781 4456 18707 1534 12303 19410 16777 30660 20567 4689
 19585 28627 7411 6034 11329 1746 6456 11207 17397 313 24004 28245 9699 9928 1808 28222 18789 1757 29149 25937 9390 13698 24891 31250 9788 19764 27255 13419 18975 16751 17072 436
9 18152 19966 18988 24501 3695 3844 29058 22510 7758 3528 21612 4330 23573 20036 9827 18382 31057 10605 19714 8433 6976 65 17438 20414 16978 5010 10141 20834 11614 3220 32538 978
3 13481 551 10048 13817 22679 1765 29918 19107 45 3051 28108 20006 9038 9321 13190 14033 23732 30184 31616 20095 8104 24924 12622 16513 3466 12529 7798 6731 9984 22156 13329 1097
3 21754 1831 21028 27895 1388 16081 10434 26619 22475 31099 3804 13218 5445 11404 27375 2654 28867 19747 2380 5780 19099 28461 31812 26230 29724 18726 25084 12991 11963 10416 112
75 25625 25363 9174 17621 838 19371 4892 11944 17 3936 8271 30871 31478 1971 30214 9496 32143 11958 23545 24159 19908 28883 19466 5239 18355 24130 19469 24872 27224 30791 6231 19
841 773 5678 27403 21832 9089 7689 27681 17414 20219 3365 5695 18415 2489 3149 7215 22135 14203 3014 6979 16046 29868 8965 9806 1377 22096 12339 30075 26350 15533 29534 18136 688
9 9398 8579 15260 13685 9275 11072 15928 16983 31955 13136 15649 30178 29061 28878 18015 3975 19256 31020 20437 18998 14304 11757 14413 30327 28775 18981 19786 6730 20820 19382 2
4974 12991 21061 13384 16485 29524 2064 24070 2488 7953 7598 24602 17106 22373 28655 16110 10223 31934 14304 25630 2697 31979 28668 6719 18530 11059 24590 2132 10066 32306 31721
26890 8655 20343 205 31158 28733 30941 32151 23828 23289 27564 12711 28091 8486 7536 28306 15458 54 26261 26765 13781 10297 11976 8595 11447 10469 780 23388 17142 9065 29086 1536
0 28431 19698 18314 4451 6404 3578 3082 24423 2850 19028 21922 9428 12906 7670 21756 773 27364 6796 12675 13394 21386 20921 5057 32066 26121 14381 31098 1133 20034 11624 12167 15
666 15741 13392 3441 20766 13677 10357 20176 13336 1768 31775 9832 20301 98 19227 26367 28383 2093 20041 24065 24138 7796 31718 15315 24478 752 24723 9718 26966 31345 32041 20195
 22298 28686 13635 12195 31002 28197 6790 16168 11843 11036 8371 17272 29995 9687 1644 6259 8120 3571 32455 2181 5517 29642 26516 4849 20798 31780 6253 15352 1071 5052 5760 17070
 3404 24183 14624 22680 11883 24168 2822 14092 11392 3707 29800 11848 18055 27908 31169 2641 12073 12785 7002 27017 30003 11460 8603 24856 1725 1626 25108 2807 15122 25176 18761
23004 3937 7284 13633 20169 555 7530 27853 27762 14696 953 16279 1107 24439 7199 6899 14733 25290 26074 8028 21893 1108 7836 724 19003 24754 4685 19423 32364 24597 10881 12632 16
230 14580 31162 11004 29870 29043 15973 32432 29011 2959 6542 22168 32527 18451 30050 11223 32547 11315 22380 4647 15601 13832 6794 12312 20055 27573 19636 28713 7630 12625 11371
 13961 31300 4506 514 10154 18621 25945 20625 19339 13221 1427 6835 6592 29685 29811 14083 14922 4106 10491 28762 1809 11331 26982 28099 6720 18783 24668 7750 30025 3221 22749 92
97 3251 28017 19690 32005 17572 25969 13044 24359 10802 32099 13831 3552 30060 21628 17586 6971 8224 32056 8891 22110 7537 19446 27014 24962 32288 29519 9027 11196 3477 2618 2785
8 19050 13227 22497 19071 28282 10050 319 7008 29022 15745 29826 22403 13774 25070 11805 1252 22595 816 13223 3850 27440 160 24702 28034 15102 25746 24604 17661 21966 30566 29235
 28489 21231 30540 21644 5617 5656 218 28321 19377 2669 29775 6908 20642 4403 25507 10830 27939 8274 21848 31822 3970 19592 26616 19031 6257 15216 8815 32704 550 14645 32020 1535
5 20055 28180 28525 23149 26325 28568 3781 7184 23810 10638 2418 20169 13308 23241 2785 3721 8582 8606 21610 31783 29114 21533 5837 1529 31081 8206 22671 7732 17238 11944 29943 3
1497 25623 18454 32743 15483 8389 12978 6747 5121 21608 18935 963 18912 16423 2375 29696 15430 12839 1635 2624 13350 12104 6058 23675 11225 2948 8720 29639 25452 7458 19165 16744
 24275 30366 2347 22672 10667 21337 21278 30966 31982 8958 25312 13850 8762 24507 20324 18266 12541 6940 29911 19634 328 28103 26852 5206 28784 13787 30383 11507 21349 20517 2420
7 20670 13010 14343 21899 30019 11513 14887 148 3051 6477 9691 26500 16207 18893 10577 7128 14658 1572 17534 3888 15769 7296 32461 21892 9658 9270 19331 20260 2282 19986 18676 30
325 7506 24191 18024 19821 7635 11612 15562 9931 16531 18329 9751 18200 2620 3731 24449 6494 2581 28981 613 29610 19148 6858 24563 10049 1021 13221 2967 22693 22657 3952 8233 233
68 11641 25939 11226 3339 26143 28107 28378 28028 4165 584 22728 27860 19893 6789 28897 11946 1578 17319 5353 15481 11811 27932 15057 25558 11895 10034 8494 15468 14484 12147 674
9 4542 5007 26306 24181 11419 11768 3390 4075 14432 93 24529 24328 16370 3222 29592 30732 32375 1494

Start timing...
End timing.
The elapsed time (us) for parallel 4-way merge-sort algorithm implemented in Q2.2 is 440.

Start timing...
End timing.
The elapsed time (us) for sequential bubble sort is 3452.

The sort result by merge sort is corrent, verified by bubble sort.
This is the END of the program.
```

T(sequential bubblesort) = 3452 us, T(parallel 4-way merge-sort) = 440 us
In this case, parallel 4-way merge-sort can run faster than the sequential bubble sort.
There exist a proper size n and max_num such that parallel 4-way merge-sort can run faster than the sequential bubble sort.
n = 1, max_num = 256
speedup = T(sequential bubblesort)/ T(parallel 4-way merge-sort) = 7.85



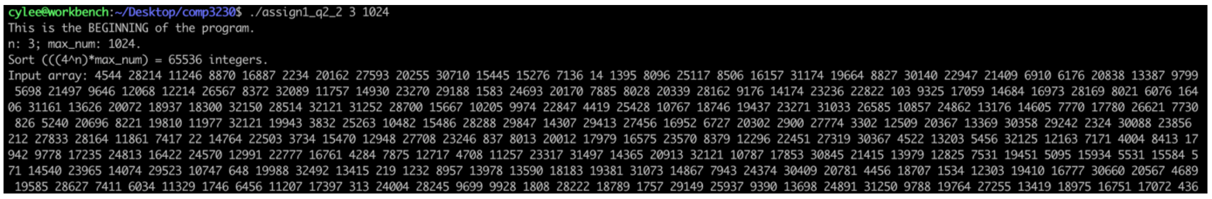
```
cylee@workbench:~/Desktop/comp3230$ ./assign1_q2_2 2 4096
This is the BEGINNING of the program.
n: 2; max_num: 4096.
Sort (((4^n)*max_num) = 65536 integers.
Input array: 4544 28214 11246 8870 16887 2234 20162 27593 20255 30710 15445 15276 7136 14 1395 8096 25117 8506 16157 31
174 19664 8827 30140 22947 21409 6910 6176 20838 13387 9799 5698 21497 9646 12068 12214 26567 8372 32089 11757 14930 23
270 29188 1583 24693 20170 7885 8028 20339 28162 9176 14174 23236 22822 103 9325 17059 14684 16973 28169 8021 6076 1640
6 31161 13626 20072 18937 18300 32150 28514 32121 31252 28700 15667 10205 9974 22847 4419 25428 10767 18746 19437 23271
```

SKIP INTERMEDIATE OUTPUT



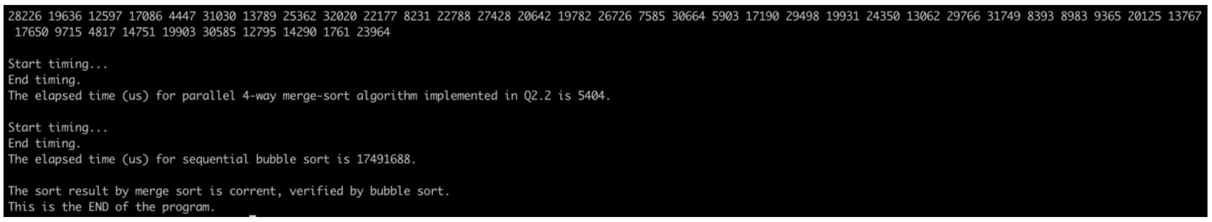
```
1030 13789 25362 32020 22177 8231 22788 27428 20642 19782 26726 7585 30664 5903 17190 29498 19931 24350 13062 29766 317
49 8393 8983 9365 20125 13767 17650 9715 4817 14751 19903 30585 12795 14290 1761 23964

Start timing...
End timing.
The elapsed time (us) for parallel 4-way merge-sort algorithm implemented in Q2.2 is 5513.

Start timing...
End timing.
The elapsed time (us) for sequential bubble sort is 17547321.

The sort result by merge sort is corrent, verified by bubble sort.
This is the END of the program.
```

T(sequential bubblesort) = 17547321 us, T(parallel 4-way merge-sort) = 5513 us
In this case, parallel 4-way merge-sort can run faster than the sequential bubble sort.
There exist a proper size n and max_num such that parallel 4-way merge-sort can run faster than the sequential bubble sort.
n = 2, max_num = 4096
speedup = T(sequential bubblesort)/ T(your parallel 4-way merge-sort) = 3182.9



```
cylee@workbench:~/Desktop/comp3230$ ./assign1_q2_2 3 1024
This is the BEGINNING of the program.
n: 3; max_num: 1024.
Sort (((4^n)*max_num) = 65536 integers.
Input array: 4544 28214 11246 8870 16887 2234 20162 27593 20255 30710 15445 15276 7136 14 1395 8096 25117 8506 16157 31174 19664 8827 30140 22947 21409 6910 6176 20838 13387 9799
 5698 21497 9646 12068 12214 26567 8372 32089 11757 14930 23270 29188 1583 24693 20170 7885 8028 20339 28162 9176 14174 23236 22822 103 9325 17059 14684 16973 28169 8021 6076 164
06 31161 13626 20072 18937 18300 32150 28514 32121 31252 28700 15667 10205 9974 22847 4419 25428 10767 18746 19437 23271 31033 26585 10857 24862 13176 14605 7770 17780 26621 7730
 826 5240 20696 8221 19810 11977 32121 19943 3832 25263 10482 15486 28288 29847 14307 29413 27456 16952 6727 20302 2900 27774 3302 12509 20367 13369 30358 29242 2324 30088 23856
212 27833 28164 11861 7417 22 14764 22503 3734 15470 12948 27708 23246 837 8013 20012 17979 16575 23570 8379 12296 22451 27319 30367 4522 13203 5456 32125 12163 7171 4004 8413 17
942 9778 17235 24813 16422 24570 12991 22777 16761 4284 7875 12717 4708 11257 23317 31497 14365 20913 32121 10787 17853 30845 21415 13979 12825 7531 19451 5095 15934 5531 15584 5
71 14540 23965 14074 29523 10747 648 19988 32492 13415 219 1232 8957 13978 13590 18183 19381 31073 14867 7943 24374 30409 20781 4456 18707 1534 12303 19410 16777 30660 20567 4689
 19585 28627 7411 6034 11329 1746 6456 11207 17397 313 24004 28245 9699 9928 1808 28222 18789 1757 29149 25937 9390 13698 24891 31250 9788 19764 27255 13419 18975 16751 17072 436
```

SKIP INTERMEDIATE OUTPUT

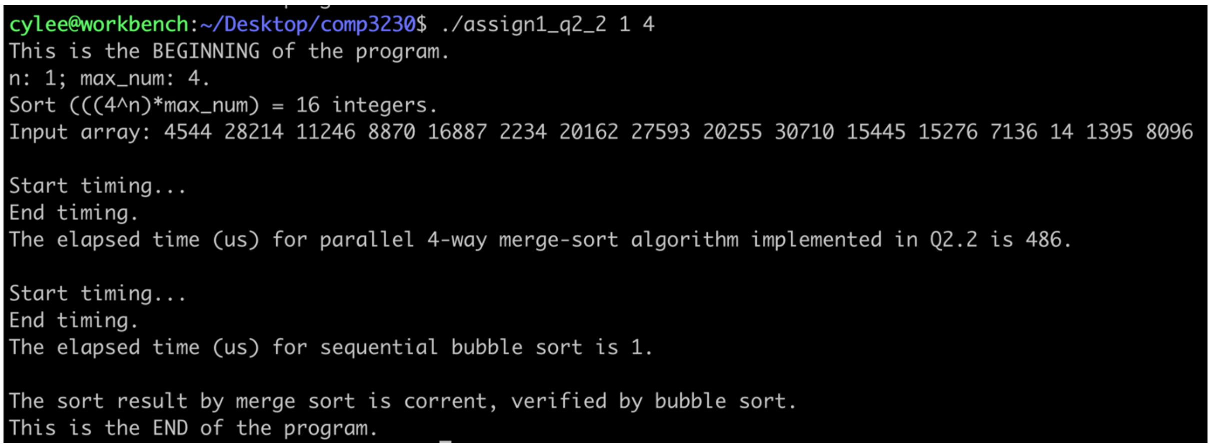


```
28226 19636 12597 17086 4447 31030 13789 25362 32020 22177 8231 22788 27428 20642 19782 26726 7585 30664 5903 17190 29498 19931 24350 13062 29766 31749 8393 8983 9365 20125 13767
 17650 9715 4817 14751 19903 30585 12795 14290 1761 23964

Start timing...
End timing.
The elapsed time (us) for parallel 4-way merge-sort algorithm implemented in Q2.2 is 5404.

Start timing...
End timing.
The elapsed time (us) for sequential bubble sort is 17491688.

The sort result by merge sort is corrent, verified by bubble sort.
This is the END of the program.
```

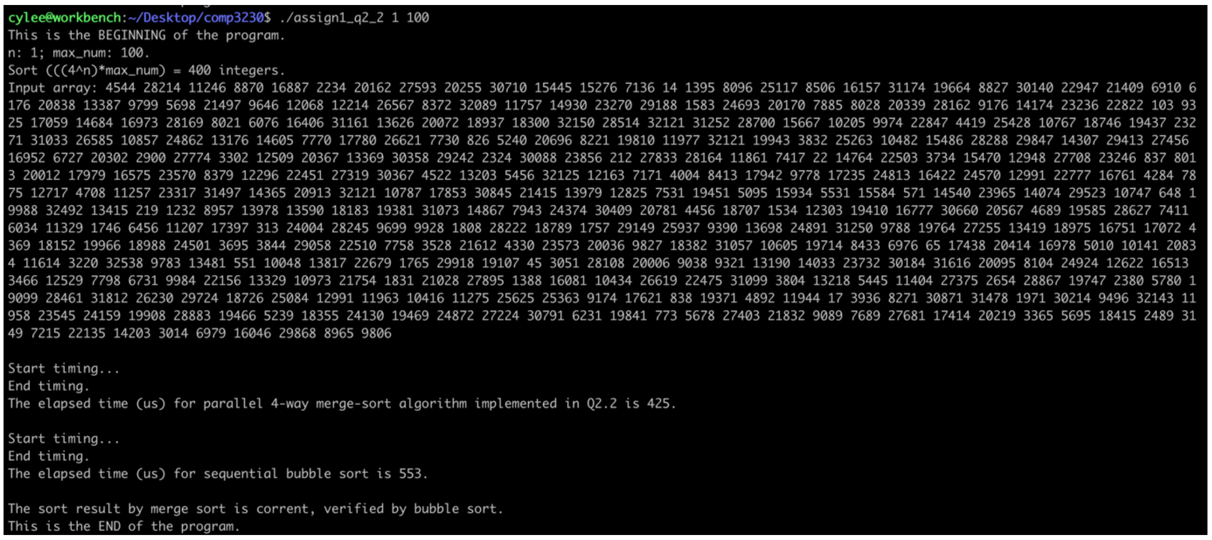
T(sequential bubblesort) = 17491688 us, T(parallel 4-way merge-sort) = 5404 us
In this case, parallel 4-way merge-sort can run faster than the sequential bubble sort.
There exist a proper size n and max_num such that parallel 4-way merge-sort can run faster than the sequential bubble sort.
n = 3, max_num = 1024
speedup = T(sequential bubblesort)/ T(your parallel 4-way merge-sort) = 3236.8

Analysis:
Time complexity of four way merge-sort in worst case = O(n $\log_4$ n)
Time complexity of bubble sort in worst case = O(n^2)
However,
Time complexity of four way merge-sort in best case = O(n $\log_4$ n)
Time complexity of bubble sort in best case = O(n)

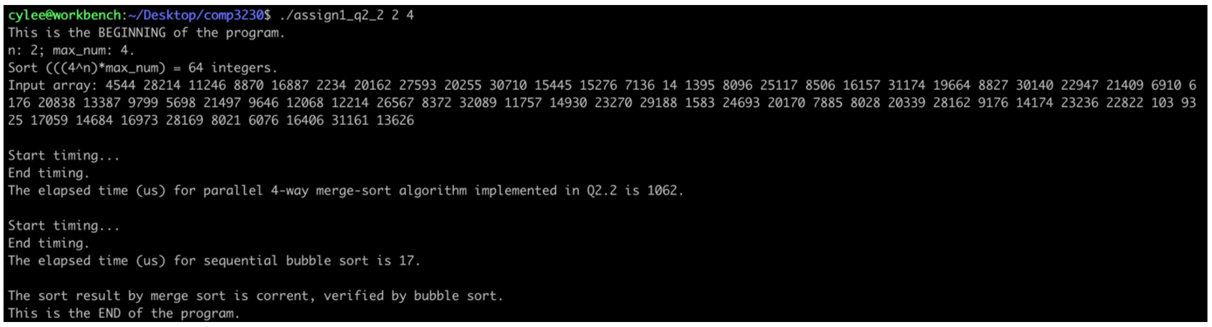
Therefore, for small n and max_num, the bubble sort is faster than the four way merge-sort. If we increase either n and max_num gradually, the four way merge-sort will be faster than the bubble sort gradually.



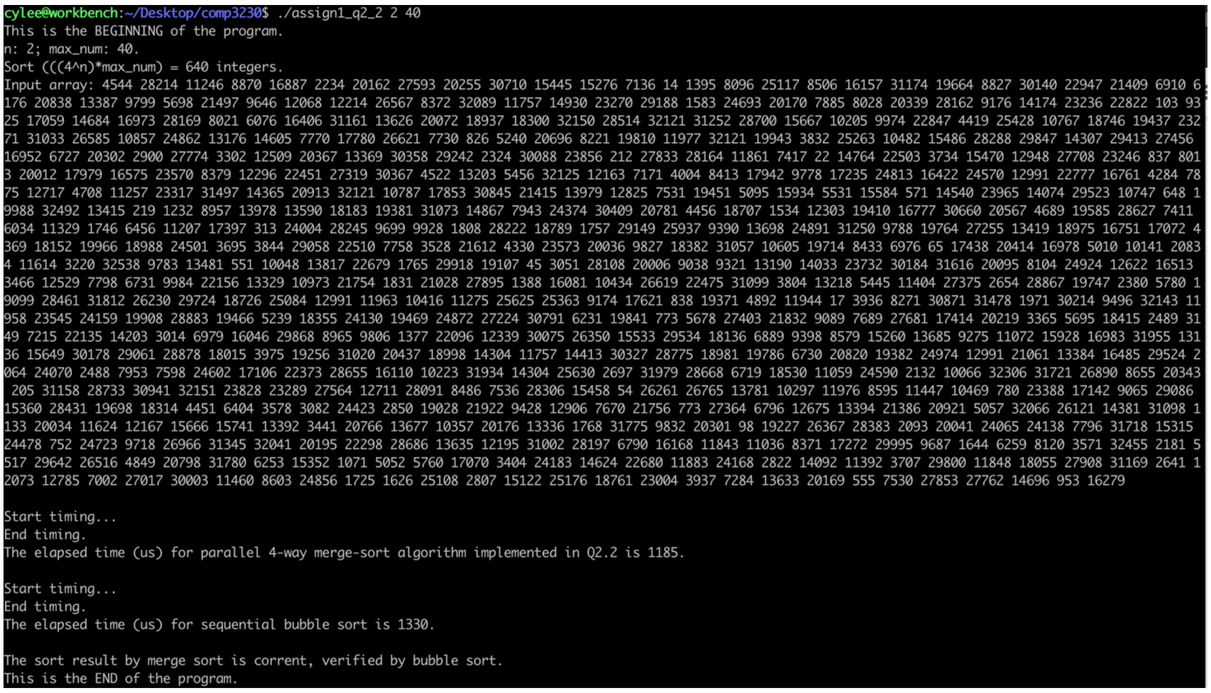
```
cylee@workbench:~/Desktop/comp3230$ ./assign1_q2_2 1 4
This is the BEGINNING of the program.
n: 1; max_num: 4.
Sort (((4^n)*max_num) = 16 integers.
Input array: 4544 28214 11246 8870 16887 2234 20162 27593 20255 30710 15445 15276 7136 14 1395 8096

Start timing...
End timing.
The elapsed time (us) for parallel 4-way merge-sort algorithm implemented in Q2.2 is 486.

Start timing...
End timing.
The elapsed time (us) for sequential bubble sort is 1.

The sort result by merge sort is corrent, verified by bubble sort.
This is the END of the program.
```



```
cylee@workbench:~/Desktop/comp3230$ ./assign1_q2_2 1 100
This is the BEGINNING of the program.
n: 1; max_num: 100.
Sort (((4^n)*max_num) = 400 integers.
Input array: 4544 28214 11246 8870 16887 2234 20162 27593 20255 30710 15445 15276 7136 14 1395 8096 25117 8506 16157 31174 19664 8827 30140 22947 21409 6910 6
176 20838 13387 9799 5698 21497 9646 12068 12214 26567 8372 32089 11757 14930 23270 29188 1583 24693 20170 7885 8028 20339 28162 9176 14174 23236 22822 103 93
25 17059 14684 16973 28169 8021 6076 16406 31161 13626 20072 18937 18300 32150 28514 32121 31252 28700 15667 10205 9974 22847 4419 25428 10767 18746 19437 232
71 31033 26585 10857 24862 13176 14605 7770 17780 26621 7730 826 5240 20696 8221 19810 11977 32121 19943 3832 25263 10482 15486 28288 29847 14307 29413 27456
16952 6727 20302 2900 27774 3302 12509 20367 13369 30358 29242 2324 30088 23856 212 27833 28164 11861 7417 22 14764 22503 3734 15470 12948 27708 23246 837 801
3 20012 17979 16575 23570 8379 12296 22451 27319 30367 4522 13203 5456 32125 12163 7171 4004 8413 17942 9778 17235 24813 16422 24570 12991 22777 16761 4284 78
75 12717 4708 11257 23317 31497 14365 20913 32121 10787 17853 30845 21415 13979 12825 7531 19451 5095 15934 5531 15584 571 14540 23965 14074 29523 10747 648 1
9988 32492 13415 219 1232 8957 13978 13590 18183 19381 31073 14867 7943 24374 30409 20781 4456 18707 1534 12303 19410 16777 30660 20567 4689 19585 28627 7411
6034 11329 1746 6456 11207 17397 313 24004 28245 9699 9928 1808 28222 18789 1757 29149 25937 9390 13698 24891 31250 9788 19764 27255 13419 18975 16751 17072 4
369 18152 19966 18988 24501 3695 3844 29058 22510 7758 3528 21612 4330 23573 20036 9827 18382 31057 10605 19714 8433 6976 65 17438 20414 16978 5010 10141 2083
4 11614 3220 32538 9783 13481 551 10048 13817 22679 1765 29918 19107 45 3051 28108 20006 9038 9321 13190 14033 23732 30184 31616 20095 8104 24924 12622 16513
3466 12529 7798 6731 9984 22156 13329 10973 21754 1831 21028 27895 1388 16081 10434 26619 22475 31099 3804 13218 5445 11404 27375 2654 28867 19747 2380 5780 1
9099 28461 31812 26230 29724 18726 25084 12991 11963 10416 11275 25625 25363 9174 17621 838 19371 4892 11944 17 3936 8271 30871 31478 1971 30214 9496 32143 11
958 23545 24159 19908 28883 19466 5239 18355 24130 19469 24872 27224 30791 6231 19841 773 5678 27403 21832 9089 7689 27681 17414 20219 3365 5695 18415 2489 31
49 7215 22135 14203 3014 6979 16046 29868 8965 9806

Start timing...
End timing.
The elapsed time (us) for parallel 4-way merge-sort algorithm implemented in Q2.2 is 425.

Start timing...
End timing.
The elapsed time (us) for sequential bubble sort is 553.

The sort result by merge sort is corrent, verified by bubble sort.
This is the END of the program.
```



```
cylee@workbench:~/Desktop/comp3230$ ./assign1_q2_2 2 4
This is the BEGINNING of the program.
n: 2; max_num: 4.
Sort (((4^n)*max_num) = 64 integers.
Input array: 4544 28214 11246 8870 16887 2234 20162 27593 20255 30710 15445 15276 7136 14 1395 8096 25117 8506 16157 31174 19664 8827 30140 22947 21409 6910 6
176 20838 13387 9799 5698 21497 9646 12068 12214 26567 8372 32089 11757 14930 23270 29188 1583 24693 20170 7885 8028 20339 28162 9176 14174 23236 22822 103 93
25 17059 14684 16973 28169 8021 6076 16406 31161 13626

Start timing...
End timing.
The elapsed time (us) for parallel 4-way merge-sort algorithm implemented in Q2.2 is 1062.

Start timing...
End timing.
The elapsed time (us) for sequential bubble sort is 17.

The sort result by merge sort is corrent, verified by bubble sort.
This is the END of the program.
```



```
cylee@workbench:~/Desktop/comp3230$ ./assign1_q2_2 2 40
This is the BEGINNING of the program.
n: 2; max_num: 40.
Sort (((4^n)*max_num) = 640 integers.
Input array: 4544 28214 11246 8870 16887 2234 20162 27593 20255 30710 15445 15276 7136 14 1395 8096 25117 8506 16157 31174 19664 8827 30140 22947 21409 6910 6
176 20838 13387 9799 5698 21497 9646 12068 12214 26567 8372 32089 11757 14930 23270 29188 1583 24693 20170 7885 8028 20339 28162 9176 14174 23236 22822 103 93
25 17059 14684 16973 28169 8021 6076 16406 31161 13626 20072 18937 18300 32150 28514 32121 31252 28700 15667 10205 9974 22847 4419 25428 10767 18746 19437 232
71 31033 26585 10857 24862 13176 14605 7770 17780 26621 7730 826 5240 20696 8221 19810 11977 32121 19943 3832 25263 10482 15486 28288 29847 14307 29413 27456
16952 6727 20302 2900 27774 3302 12509 20367 13369 30358 29242 2324 30088 23856 212 27833 28164 11861 7417 22 14764 22503 3734 15470 12948 27708 23246 837 801
3 20012 17979 16575 23570 8379 12296 22451 27319 30367 4522 13203 5456 32125 12163 7171 4004 8413 17942 9778 17235 24813 16422 24570 12991 22777 16761 4284 78
75 12717 4708 11257 23317 31497 14365 20913 32121 10787 17853 30845 21415 13979 12825 7531 19451 5095 15934 5531 15584 571 14540 23965 14074 29523 10747 648 1
9988 32492 13415 219 1232 8957 13978 13590 18183 19381 31073 14867 7943 24374 30409 20781 4456 18707 1534 12303 19410 16777 30660 20567 4689 19585 28627 7411
6034 11329 1746 6456 11207 17397 313 24004 28245 9699 9928 1808 28222 18789 1757 29149 25937 9390 13698 24891 31250 9788 19764 27255 13419 18975 16751 17072 4
369 18152 19966 18988 24501 3695 3844 29058 22510 7758 3528 21612 4330 23573 20036 9827 18382 31057 10605 19714 8433 6976 65 17438 20414 16978 5010 10141 2083
4 11614 3220 32538 9783 13481 551 10048 13817 22679 1765 29918 19107 45 3051 28108 20006 9038 9321 13190 14033 23732 30184 31616 20095 8104 24924 12622 16513
3466 12529 7798 6731 9984 22156 13329 10973 21754 1831 21028 27895 1388 16081 10434 26619 22475 31099 3804 13218 5445 11404 27375 2654 28867 19747 2380 5780 1
9099 28461 31812 26230 29724 18726 25084 12991 11963 10416 11275 25625 25363 9174 17621 838 19371 4892 11944 17 3936 8271 30871 31478 1971 30214 9496 32143 11
958 23545 24159 19908 28883 19466 5239 18355 24130 19469 24872 27224 30791 6231 19841 773 5678 27403 21832 9089 7689 27681 17414 20219 3365 5695 18415 2489 31
49 7215 22135 14203 3014 6979 16046 29868 8965 9806 1377 22096 12339 30075 26350 15533 29534 18136 6889 9398 8579 15260 13685 9275 11072 15928 16983 31955 131
36 15649 30178 29061 28878 18015 3975 19256 31020 20437 18998 14304 11757 14413 30327 28775 18981 19786 6730 20820 19382 24974 12991 21061 13384 16485 29524 2
064 24070 2488 7953 7598 24602 17106 22373 28655 16110 10223 31934 14304 25630 2697 31979 28668 6719 18530 11059 24590 2132 10066 32306 31721 26890 8655 20343
 205 31158 28733 30941 32151 23828 23289 27564 12711 28091 8486 7536 28306 15458 54 26261 26765 13781 10297 11976 8595 11447 10469 780 23388 17142 9065 29086
15360 28431 19698 18314 4451 6404 3578 3082 24423 2850 19028 21922 9428 12906 7670 21756 773 27364 6796 12675 13394 21386 20921 5057 32066 26121 14381 31098 1
133 20034 11624 12167 15666 15741 13392 3441 20766 13677 10357 20176 13336 1768 31775 9832 20301 98 19227 26367 28383 2093 20041 24065 24138 7796 31718 15315
24478 752 24723 9718 26966 31345 32041 20195 22298 28686 13635 12195 31002 28197 6790 16168 11843 11036 8371 17272 29995 9687 1644 6259 8120 3571 32455 2181 5
517 29642 26516 4849 20798 31780 6253 15352 1071 5052 5760 17070 3404 24183 14624 22680 11883 24168 2822 14092 11392 3707 29800 11848 18055 27908 31169 2641 1
2073 12785 7002 27017 30003 11460 8603 24856 1725 1626 25108 2807 15122 25176 18761 23004 3937 7284 13633 20169 555 7530 27853 27762 14696 953 16279

Start timing...
End timing.
The elapsed time (us) for parallel 4-way merge-sort algorithm implemented in Q2.2 is 1185.

Start timing...
End timing.
The elapsed time (us) for sequential bubble sort is 1330.

The sort result by merge sort is corrent, verified by bubble sort.
This is the END of the program.
```