

Q2.3:

(1) $n=1$, $\text{max_num}=1024$, total 4K integers, 4 processes

```
jpwang@workbench: ~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_e...
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$ gcc assign1_q2_funcs.c main.c -o main -lm
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$ ./main 1 1024
This is the BEGINNING of the program.
n: 1; max_num: 1024.
Sort (((4^n)*max_num) = 4096 integers.

Start timing...
End timing.
[Merge sort] The elapsed time (us) is 681.

[Bubble sort] The elapsed time (us) is 50181.

The sort result by merge sort is correct, verified by bubble sort.
This is the END of the program.
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$
```

$$T(\text{sequential bubble sort}) / T(\text{parallel 4-way merge-sort}) = 50181/681=74$$

(2) $n=2$, $\text{max_num}=4096$, total 64K integers, 16 processes

```
jpwang@workbench: ~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_e...
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$ ./main 2 4096
This is the BEGINNING of the program.
n: 2; max_num: 4096.
Sort (((4^n)*max_num) = 65536 integers.

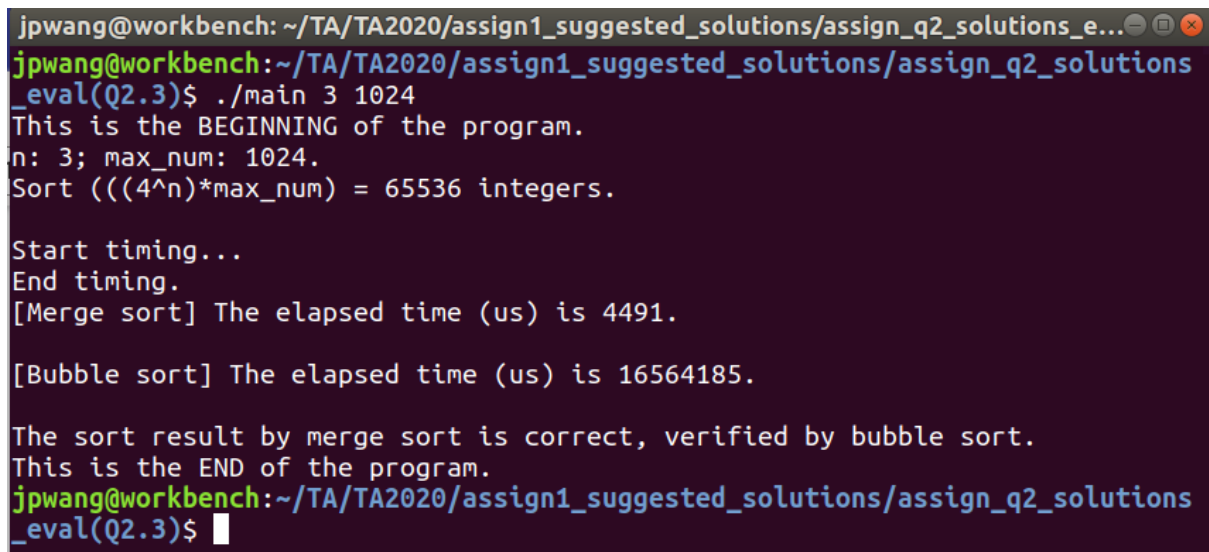
Start timing...
End timing.
[Merge sort] The elapsed time (us) is 4206.

[Bubble sort] The elapsed time (us) is 16784123.

The sort result by merge sort is correct, verified by bubble sort.
This is the END of the program.
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$
```

$$T(\text{sequential bubble sort}) / T(\text{parallel 4-way merge-sort}) = 16784123/4206 = 3991$$

(3) $n=3$, $\text{max_num}=1024$, total 64K integers. 64 processes



```
jpwang@workbench: ~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_e...
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions
_eval(Q2.3)$ ./main 3 1024
This is the BEGINNING of the program.
n: 3; max_num: 1024.
Sort (((4^n)*max_num) = 65536 integers.

Start timing...
End timing.
[Merge sort] The elapsed time (us) is 4491.

[Bubble sort] The elapsed time (us) is 16564185.

The sort result by merge sort is correct, verified by bubble sort.
This is the END of the program.
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions
_eval(Q2.3)$
```

$T(\text{sequential bubble sort}) / T(\text{parallel 4-way merge-sort}) = 16564185 / 4491 = 3688$

(4) Can you find a proper size of n and max_num to show your parallel 4-way merge-sort can run faster than the sequential bubble sort?

Any solution which reports a faster speed with the speedup and screenshots is correct.

Please note that the parallel merge sort here sometimes runs slower than the sequential bubble sort as shown in the screenshot below.

```
jpwang@workbench: ~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_e...
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$ ./main 1 4
This is the BEGINNING of the program.
n: 1; max_num: 4.
Sort (((4^n)*max_num) = 16 integers.

Start timing...
End timing.
[Merge sort] The elapsed time (us) is 522.

[Bubble sort] The elapsed time (us) is 1.

The sort result by merge sort is correct, verified by bubble sort.
This is the END of the program.
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$ ./main 2 4
This is the BEGINNING of the program.
n: 2; max_num: 4.
Sort (((4^n)*max_num) = 64 integers.

Start timing...
End timing.
[Merge sort] The elapsed time (us) is 807.

[Bubble sort] The elapsed time (us) is 15.

The sort result by merge sort is correct, verified by bubble sort.
This is the END of the program.
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$ ./main 2 10
This is the BEGINNING of the program.
n: 2; max_num: 10.
Sort (((4^n)*max_num) = 160 integers.

Start timing...
End timing.
[Merge sort] The elapsed time (us) is 944.

[Bubble sort] The elapsed time (us) is 109.

The sort result by merge sort is correct, verified by bubble sort.
This is the END of the program.
jpwang@workbench:~/TA/TA2020/assign1_suggested_solutions/assign_q2_solutions_eval(Q2.3)$
```