Chapter 1. The Role of Algorithms in Computing

1.1-2.
Memory usage is a significant factor of consideration in a real-world setting.

1.1-4.
Shortest path problem finds a shortest path between two nodes. Traveling salesman problem finds a shortest path that visits each city and returns to the origin city.

1.1-5.
The longest increasing subsequence problem is the problem in which only the best solution will do. The A* algorithm is the problem in which the heuristic solution will be enough.

1.2-2.
For $n < 44$, insertion sort beat merge sort.

1.2-3.
$n = 15$ is the smallest value such that $100n^2 < 2^n$.

1-1.
For $n = 10^{10000}$, algorithm with $\log n$ complexity solves in 0.033 seconds.
For $n = 10^{10000}$, algorithm with $\sqrt{n}$ complexity solves in 10^{4985} centuries.
For $n = 10^{30}$, algorithm with $\log n$ complexity solves in 0.0001 seconds.
For $n = 10^{30}$, algorithm with $\sqrt{n}$ complexity solves in 0.315 centuries.
For $n = 10^{30}$, algorithm with n complexity solves in $3.170 \cdot 10^{14}$ centuries.
For $n = 10^{15}$, algorithm with $\sqrt{n}$ complexity solves in 31.62 seconds.
For $n = 10^{15}$, algorithm with n complexity solves in 0.317 centuries.

For $n = 10^{15}$, algorithm with $n \log n$ complexity solves in 15.798 centuries.
For $n = 10^{13}$, algorithm with $\sqrt{n}$ complexity solves in 3.162 seconds.
For $n = 10^{13}$, algorithm with n complexity solves in 4 months.
For $n = 10^{13}$, algorithm with $n \log n$ complexity solves in 0.1371 centuries.
For $n = 10^{10}$, algorithm with $\sqrt{n}$ complexity solves in 0.1 seconds.
For $n = 10^{10}$, algorithm with n complexity solves in 3 hours.
For $n = 10^{10}$, algorithm with $n \log n$ complexity solves in 4 days.
For $n = 10^8$, algorithm with n complexity solves in 2 minutes.
For $n = 10^8$, algorithm with $n \log n$ complexity solves in 1 hours.
For $n = 10^8$, algorithm with n^2 complexity solves in 3.2 centuries.
For $n = 10^7$, algorithm with n complexity solves in 10 seconds.
For $n = 10^7$, algorithm with $n \log n$ complexity solves in 3.9 minutes.
For $n = 10^7$, algorithm with n^2 complexity solves in 3.2 years.
For $n = 10^6$, algorithm with n complexity solves in 1 seconds.
For $n = 10^6$, algorithm with $n \log n$ complexity solves in 0.33 minutes.
For $n = 10^6$, algorithm with n^2 complexity solves in 11.6 days.
For $n = 10^5$, algorithm with $n \log n$ complexity solves in 1.66 seconds.
For $n = 10^5$, algorithm with n^2 complexity solves in 3 hours.
For $n = 10^5$, algorithm with n^3 complexity solves in 0.317 centuries.
For $n = 10^4$, algorithm with $n \log n$ complexity solves in 0.13 seconds.
For $n = 10^4$, algorithm with n^2 complexity solves in 2 minutes.
For $n = 10^4$, algorithm with n^3 complexity solves in 11.6 days.
For $n = 10^3$, algorithm with n^2 complexity solves in 1 seconds.
For $n = 10^3$, algorithm with n^3 complexity solves in 17 minutes.
For $n = 10^2$, algorithm with n^3 complexity solves in 1 seconds.
For $n = 10^2$, algorithm with 2^n complexity solves in 420000000000000 centuries.
For $n = 50$, algorithm with 2^n complexity solves in 36 years.
For $n = 40$, algorithm with 2^n complexity solves in 13 days.
For $n = 30$, algorithm with 2^n complexity solves in 18 minutes.
For $n = 20$, algorithm with 2^n complexity solves in 1.1 seconds.
For $n = 20$, algorithm with $n!$ complexity solves in 771 centuries.
For $n = 18$, algorithm with $n!$ complexity solves in 2.03 centuries.
For $n = 17$, algorithm with $n!$ complexity solves in 11.3 years.
For $n = 16$, algorithm with $n!$ complexity solves in 243 days.
For $n = 15$, algorithm with $n!$ complexity solves in 16 days.
For $n = 14$, algorithm with $n!$ complexity solves in 25 hours.
For $n = 13$, algorithm with $n!$ complexity solves in 2 hours.

For $n = 12$, algorithm with $n!$ complexity solves in 8 minutes.
For $n = 11$, algorithm with $n!$ complexity solves in 40 seconds.
For $n = 10$, algorithm with $n!$ complexity solves in 4 seconds.