1. Suppose a machine on average takes 10^{-8} seconds to execute a single algorithm step. What is the largest input size for which the machine will execute the algorithm in 2 seconds assuming the number of steps is $T(n) =$

1. $log_2(n)$

$$\begin{aligned} log_2(n) \cdot 10^{-8} &= 2 \\ &= \frac{2}{10^{-8}} \\ &= 2 \cdot 10^8 \\ n &= 2^{2 \cdot 10^8} \end{aligned}$$

2. $\sqrt{n}$

$$\begin{aligned} \sqrt{n} \cdot 10^{-8} &= 2 \\ &= 2 \cdot 10^8 \\ n &= (2 \cdot 10^8)^2 \\ &= 4 \cdot 10^{16} \end{aligned}$$

3. n

$$\begin{aligned} n \cdot 10^{-8} &= 2 \\ n &= 2 \cdot 10^8 \end{aligned}$$

4. n^2

$$\begin{aligned} n^2 \cdot 10^{-8} &= 2 \\ &= 2 \cdot 10^8 \\ n &= \sqrt{2} \cdot 10^4 \end{aligned}$$

5. n^3

$$\begin{aligned} n^3 \cdot 10^{-8} &= 2 \\ &= 2 \cdot 10^8 \\ n &= \sqrt[3]{2 \cdot 10^8} \end{aligned}$$

6. 2^n

$$\begin{aligned} 2^n \cdot 10^{-8} &= 2 \\ &= 2 \cdot 10^8 \\ n &= log_2(2 \cdot 10^8) \\ &= 1 + 8 \cdot log_2(10) \end{aligned}$$

2. For the machine in the previous example, how long will it take to run the algorithm for an input of size $1,000$ assuming the time complexities from the same example?

1. $log_2(n)$

$$log_2(10^3) \cdot 10^{-8} = 3 \cdot log_2(10) \cdot 10^{-8}$$

2. $\sqrt{n}$

$$\sqrt{10^3} \cdot 10^{-8} \approx 31 \cdot 10^{-8}$$

3. n

$$10^3 \cdot 10^{-8} = 10^{-5}$$

4. n^2

$$\begin{aligned} (10^3)^2 \cdot 10^{-8} &= 10^6 \cdot 10^{-8} \\ &= 10^{-2} \end{aligned}$$

5. n^3

$$\begin{aligned} (10^3)^3 \cdot 10^{-8} &= 10^9 \cdot 10^{-8} \\ &= 10 \end{aligned}$$

6. 2^n

$$2^{10^3} \cdot 10^{-8}$$

3. An algorithm takes 0.5 seconds to run on an input of size 100. How long will it take to run on an input size of 1000 if the algorithm has a running time that is linear? quadratic? log-linear? cubic?

1. n

$$\begin{aligned} 100 \cdot c &= 0.5 \\ c &= \frac{0.5}{100} \end{aligned}$$

$$1000 \cdot \frac{0.5}{100} = 0.5 \cdot 10$$

2. n^2

$$\begin{aligned} 10^4 \cdot c &= 0.5 \\ c &= 0.5 \cdot 10^{-4} \end{aligned}$$

$$\begin{aligned} (10^3)^2 \cdot 0.5 \cdot 10^{-4} &= 10^6 \cdot 0.5 \cdot 10^{-4} \\ &= 10^2 \cdot 0.5 \end{aligned}$$

3. $n \cdot log_2(n)$

$$\begin{aligned} 100 \cdot log_2(100) \cdot c &= 0.5 \\ c &= \frac{0.5}{100 \cdot log_2(100)} \end{aligned}$$

$$\begin{aligned} 10^3 \cdot log_2(10^3) \cdot \frac{0.5}{100 \cdot log_2(100)} &= 10 \cdot 3 \cdot log_2(10) \cdot \frac{0.5}{log_2(10) \cdot 2} \\ &= \frac{10 \cdot 3 \cdot 0.5}{2} \end{aligned}$$

4. n^3

$$\begin{aligned} (10^2)^3 \cdot c &= 0.5 \\ c &= 0.5 \cdot 10^{-6} \end{aligned}$$

$$\begin{aligned} (10^3)^3 \cdot c &= 10^9 \cdot 0.5 \cdot 10^{-6} \\ &= 0.5 \cdot 10^3 \end{aligned}$$

4. An algorithm is to be implemented and run on a processor that can execute a single instruction in an average of 10^{-9} seconds. What is the largest problem size that can be solved in one hour by the algorithm on this processor if the number of steps needed to execute the algorithm is $n, n^2, n^3, log_2(n)$? Assume n is the input size.

1. n

$$\begin{aligned} n \cdot 10^{-9} &= 3600 \\ n &= 3600 \cdot 10^9 \end{aligned}$$

2. n^2

$$\begin{aligned} n^2 \cdot 10^{-9} &= 3600 \\ n^2 &= 3600 \cdot 10^9 \\ n &= \sqrt{3600 \cdot 10^9} \end{aligned}$$

3. n^3

$$\begin{aligned} n^3 \cdot 10^{-9} &= 3600 \\ n^3 &= 3600 \cdot 10^9 \\ n &= \sqrt[3]{3600 \cdot 10^9} \end{aligned}$$

4. $log_2(n)$

$$\begin{aligned} log_2(n) \cdot 10^{-9} &= 3600 \\ log_2(n) &= 3600 \cdot 10^9 \\ n &= 2^{3600 \cdot 10^9} \end{aligned}$$

5. Determine the asymptotic running time for the following piece of code, assuming that n represents the input size.

1.

$$\begin{array}{rcl} sum & = & 0 \\ for(i = 0; & i < n; & i++) \\ & sum++; & \end{array}$$

$$\sum_{i=1}^{n} 1 = \Theta(n)$$

2.

$$\begin{array}{rccc} sum = 0; & & & \\ for(i = 0; & i < n; & i++) & \\ for(j = 0; & j < n^2; & j++) & \\ & sum++ & & \end{array}$$

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{n^2} 1 &= \sum_{i=1}^{n} n^2 \\ &= \Theta(n^3) \end{aligned}$$

3.

$$\begin{array}{rccc} sum = 0 & & & \\ for(i = 0; & i < n; & i++) & \\ for(j = 0; & j < i; & j++) & \\ & sum++; & & \end{array}$$

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{i} 1 &= \sum_{i=1}^{n} i \\ &= \frac{n \cdot (n+1)}{2} \\ &= \Theta(n^2) \end{aligned}$$

4.

$$\begin{array}{rccc} sum = 0 & & & \\ for(i = 0; & i < n; & i++) & \\ for(j = 0; & j < i^2; & j++) & \\ for(k = 0; & k < j; & k++) & \\ & sum++; & & \end{array}$$

$$
\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{i^2}\sum_{k=1}^{j}1 &= \sum_{i=1}^{n}\sum_{j=1}^{i^2}j \\
&= \sum_{i=1}^{n}\frac{(i^2+1)\cdot i^2}{2} \\
&= \sum_{i=1}^{n}\Theta(i^4) \\
&= \Theta(\int_1^n x^4 dx) \\
&= \Theta(n^5)
\end{aligned}
$$

5.

$$
\begin{aligned}
sum = 0 \\
for(i = 0; \quad i < \tfrac{n}{2}; \quad i++) \\
for(j = 0; \quad j < \tfrac{i^2}{2}; \quad j++) \\
sum++;
\end{aligned}
$$

$$
\begin{aligned}
\sum_{i=1}^{\frac{n}{2}}\sum_{j=1}^{\frac{i^2}{2}}1 &= \sum_{i=1}^{\frac{n}{2}}\frac{i^2}{2} \\
&= \sum_{i=1}^{\frac{n}{2}}\Theta(i^2) \\
&= \Theta(\int_1^{\frac{n}{2}} x^2\cdot dx) \\
&= \Theta((\frac{n}{2})^3) \\
&= \Theta(n^3)
\end{aligned}
$$

6.

$$
\begin{aligned}
for(i = 0; \quad i < len(a) \quad ; i++) \\
binary\_search(a, a[i])
\end{aligned}
$$

$$
n\cdot\sum_{i=1}^{n}O(log_2(n) \quad = \quad O(n^2\cdot log_2(n))
$$

7.

$$\begin{array}{lll} for(i = 0; & i < n; & i++) \\ for(j = 0; & j < n; & j++) \\ & linear\_search(a, key) & \end{array}$$

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{n} O(n) &= \sum_{i=1}^{n} O(n^2) \\ &= O(n^3) \end{aligned}$$

6. What is the largest value of n such that an algorithm whose running time is $10 \cdot n^2$ runs faster than an algorithm whose running time is $50 \cdot n$ on the same machine?

$$\begin{aligned} 10 \cdot n^2 &< 50 \cdot n \\ n &< 5 \end{aligned}$$

$n = 4$ is the largest value such that the quadratic time algorithm runs faster than the linear time algorithm.