Data Structures and Algorithms
Fall 2025
Assignment 1
Maximum Marks: 100
Due: Friday, Oct 10, 2025 (**5pm**)

Assignment:

Note: The assignment must be submitted in hand written form. Electronic copies or printouts are not acceptable.

Q1. Order the following functions by growth rate: $n, \sqrt{n}, n^{3/2}, n^2, n \log n, n \log(\log n), n \log^2 n, n \log(n^2), n \log^2 (n^2), 2/n, 2^n, 2^{n/2}, 100, n^2 \log n, n^3, n/e^n$., Indicate which functions grow at the same rate. **(10 marks)**

Q2. Suppose $T_1(n) = O(T_2(n))$ and $T_2(n) = O(T_1(n))$. Which of the following is always true? **(10 marks)**

a. $T_1(n) = T_2(n))$
b. $T_1(n) = \Omega\ (T_2(n))$
c. $T_2(n) = \Theta\ (T_1(n))$
d. $T_1(n) + T_2(n) = O(T_1(n))$
e. $T_1(n) - T_2(n) = o(T_1\ (n))$
f. $T_1(n) * T_2(n) = O(T_1(n))$
g. $T_1(n) / T_2(n) = O(1)$
h. $T_1(n) / T_2(n) = O(T_1(n))$
i. $T_1(n) * T_2(n) = O(T_1(n)^2)$
j. *If* $T_1(n) = \Theta(T_2(n))$ and $T_2(n) = \Theta(T_1(n))$, **then** $f(n) = g(n))$

Q3. *If* $f(n) = O(g(n))$, $g(n) = O(h(n))$, **and** $h(n)/ f(n) = n^2$, then which of the following are true? **(15 marks)**

a. $f(n) = O(h(n))$
b. $h(n) = \Omega(f(n))$
c. $h(n) = \Theta(f(n))$
d. $f(n) + g(n) = O(h(n))$
e. $f(n) + g(n) = \Theta(h(n))$
f. $h(n) + g(n) = \Omega(f(n))$
g. $f(n) + h(n) = \Theta(g(n))$
h. $g(n) / f(n) = O(1)$
i. $g(n) / f(n) = O(f(n))$
j. $g(n) / f(n) = O(f(n^2))$
k. $h(n) / f(n) = O(f(n^2))$
l. $f(n) * g(n) = O(f(n))$, for any $h(n)$
m. $f(n) * g(n) = O(f(n^2))$, for any $h(n)$
n. $f(n) * g(n) = O(f(n))$, only if $(h(n) = O(n^2)$
o. $f(n) * g(n) = O(f(n^2))$ only if $(h(n) = O(n^2)$

Q4. Which function grows faster? (the two functions may have similar growth rate as well) (5 + 2 + 2 + 2 + 2 + 2 = **15 marks**)

(a)	$\boldsymbol{n}\log\boldsymbol{n}$,	or,	$\boldsymbol{n}^{[1 + 1/\log \boldsymbol{n}\,]}$	
(b)	$\boldsymbol{n^k}$	or	$\boldsymbol{c^n}$,	where $\boldsymbol{k} \geq \mathbf{1}$ and $\mathbf{c} \geq \mathbf{1}$, are constants
(c)	$\mathbf{Log_2}\,\boldsymbol{n}$	or	$\mathbf{log_{10}}\,\boldsymbol{n}$,	
(d)	$\boldsymbol{n^2}\,\mathbf{log\ n}$	or	$\boldsymbol{n}\,\mathbf{log^2\ n}$,	
(e)	$\mathbf{8}^{\boldsymbol{n}}$	or	$\mathbf{4}^{\boldsymbol{n}}$	
(f)	$\mathbf{Log}\ \boldsymbol{n}^{\mathbf{log17}}$	or	$\mathbf{Log\ 17}^{\mathbf{log}\,\boldsymbol{n}}$	

Q5. Prove that for any constant, $\boldsymbol{k}$, $\mathbf{log}^{\boldsymbol{k}}\boldsymbol{n} = \boldsymbol{o}(\boldsymbol{n})$. (**5 marks**)

Q6. Find two functions $\boldsymbol{f}(\boldsymbol{n})$ and $\boldsymbol{g}(\boldsymbol{n})$ such that (**5 marks**)

neither $(n) = O(g(n))$

nor $g(n) = O(f(n))$.

Q7. For each of the following six program fragments: (**30 marks**)

a) Give an analysis of the running time (**Big-Oh** will do).
b) Implement the code in the language of your choice, and give the running time for several values of $\boldsymbol{n}$ (e.g., n = 10, 20, 30 ,40,...,100).
c) Compare your analysis with the actual running times.

```
(1)    sum = 0;
for( i=0; i<n; i++ )
      sum++;

(2)    sum = 0;
for( i=0; i<n; i++ )
      for( j=0; j<n; j++ )
            sum++;

(3)    sum = 0;
for( i=0; i<n; i++ )
      for( j=0; j<n*n; j++ )
            sum++;
```

```
(4)    sum = 0;
for( i=0; i<n; i++ )
      for( j=0; j<i; j++ )
            sum++;
```

```
(5)    sum = 0;
for( i=0; i<n; i++ )
      for( j=0; j<i*i; j++ )
            for( k=0; k<j; k++)
                  sum++;
```

```
(6)    sum = 0;
for( i=1; i<n; i++ )
      for( j=1; j<i*i; j++ )
            if( j%1 == 0 )
                  for( k=0; k<j; k++ )
                        sum++;
```

Q8. An algorithm takes **1 ms** for input size **1000**. How large a problem (in terms of data size) can be solved in **1 min** if the running time is the following (neglecting the running times of lower order terms)? **(10 marks)**

- **a) Linear**
- **b) Logarithmic**
- **c) O(*N* log *N*)**
- **d) Quadratic**
- **e) Cubic**