2022 Summer Mid Set A	**1a** Calculate the time complexity of the following function: worstCase(n): int i, j, k, a, b, sum for (i = 0; i < n; i = i + 3) for (j = n; j >= 1; j = j / 5) for (k = 1; k <= n; k = k * 5) sum = a + b **Ans : $O(n\log_5^2 n)$**
2022 Summer Mid Set B	**1a** Calculate the time complexity of the following function: worstCase(n): int i, j, k, a, b, sum for (i = 1; i < n; i = i + 3) for (j = n; j >= 1; j = j - 2) for (k = n; k >= 1; k = k / 4) sum = a + b **Ans : $O(n^2\log_4 n)$**
2022 Fall Mid Set A	**1a** Finding_Worst_Case(n): ``` int i,j,k,m,multi,a,b,c for(i = n; i >= 1; i = i / 7){ for(j = 1; j <= n; j = j + 3) { for(k=1; k<=40 ; k=k+1){ multi=a*b } for(m=n ; m>=1 ; m=m-5){ multi=multi*c } } } ``` First for loop : log7n Second for loop : n/3 Third for loop : 40 Fourth for loop : n/5

	Total = (log7 n * n/3 * (40+n/5)) **Ans : $O(n^2 \log_7 n)$**
2022 Fall Mid Set B	**1a** Finding_Worst_Case(n): `int i,j,k,m,multi,a,b,c` `for( i = n; i >= 1; i = i - 4 ){` `for( j = 1; j <= n; j = j * 3 ) {` `for( k=1; k<=20 ; k=k+1){` `multi=a*b` `}` `for( m=n ; m>=1 ; m=m/5 ){` `multi=multi*c` `}` `}` `}` First for loop : n/4 Second for loop : $\log_3 n$ Third for loop : 20 Fourth for loop : $\log_5 n$ Total = (n/4 * $\log_3$ n * (20 + $\log_5$ n)) **Ans: $O(n(\log n)^2)$**
2023 Spring Mid Set A	**1a.** In the primary scholarship exam in Bangladesh, four lakh (n=4,00,000) students take part but only the top **50** students are given an award. **Write** the asymptotic time complexity to give the awards. Assume that each award is given in a constant time. **Ans: O(1) / Constant** **1b.** Write the asymptotic time complexity of the following function. `1. def contains_duplicates(elements):`

```
2.   for outer in range(len(elements)):
3.       for inner in range(len(elements)):
4.           if outer == inner:
5.               continue
6.
7.           if elements[outer] == elements[inner]:
8.               return True
9.
10.   return False
```

Ans: O(n^2)

2023 Spring Mid Set B

1a. In the primary scholarship exam in Bangladesh, two lakh (n=2,00,000) students take part but only the top **25** students are given an award.
Write the asymptotic time complexity to give the awards. Assume that each award is given in a constant time.

Ans : O(1) / Constant.

1b. Write the asymptotic time complexity of the following function.

```
1. def cumulative_sum(elem):
2.     for outer in range(len(elem)):
3.         for inner in range(outer+1,len(elem)):
4.             elem[outer]= elem[outer] + elem[inner]
5.
6.     return elem
```

Ans: O(n^2).

2023 Summer Mid Set A

1a. Find the time-complexity of the following task in terms of number of students.

You are given a student attendance sheet. Each student has a unique integer ID. You have to count the number of students having an even number as ID. The list is sorted but the IDs are not necessarily consecutive. So you check each ID one by one.

Ans: If number of students is N, then **O(N)**

1b. Write the asymptotic time complexity of the following code snippet. Show your works/reasoning.

```
1.  for i in range (1,n)
```

	2. **j= 1** 3. **while j < i*i** 4. **j= j+1** **Ans:** $T(N) = 1^2 + 2^2 + 3^2 + \dots + n^2 = (n(n+1)(2n+1))/6$ $= \mathbf{O(n^3)}$
2023 Summer Mid Set B	1a. **Find** the time-complexity of the following task in terms of number of students. You are given a student attendance sheet. Each student has a unique integer ID. You have to count the number of students having an ID which is divisible by 3. The list is sorted but the IDs are not necessarily consecutive. So you check each ID one by one. **Ans:** If number of students is N, then **O(N)** **1b. Write** the asymptotic time complexity of the following code snippet. Show your works/reasoning. 1. **for i in range (1,n)** 2. **j= 1** 3. **while j*j < i** 4. **j= j+1** **Ans:** First for loop : n-1 ≈ n Second while loop : Runs till j*j >= i. Means j equals to √i. Number of iterations for each value of i is √i. So, the series becomes, √1 + √2 + √3 + √4 + √5 + ... + √n. $\sqrt{1} + \sqrt{2} + \sqrt{3} + \sqrt{4} + \sqrt{5} + \dots + \sqrt{n} = \sum_1^n \sqrt{i} = \int_1^n \sqrt{n} dn = \frac{2}{3} * n^{3/2} \approx n^{3/2} \approx n^{1/2} * n^1 = \sqrt{n} * n$ **So, time complexity = O(n√n)**
2023 Fall Mid Set A	**1a.** Explain the time complexity of the following code snippet in regards of the Big-O notation: 1. **for (i=0; i<n; i+=4) {** 2. **for (j=1; j<n; j*=2) {** 3. **for (k=0; k<30; k++) {**

```
4.                  print("Am I still not 30?!!");
5.              }
6.              print("Why, God, why? We had a Deal!");
7.              for (m=n; m>0; m-=2) {
8.                  print("Could you BE more dramatic?");
9.              }
10.         }
11. }
```

Ans:
First Loop : n/4
2nd Loop : log2n
3rd Loop : 30
4th Loop : n/2

$n/4 * \log_2 n * (30 + n/2)$
$\approx n * \log_2 n * n$
$\approx n^2 * \log_2 n$

1b. Consider the following functions.

$f_1(n) = (log\ n)^{2023}$

$f_2(n) = n^2\ log_n(n^n)$

$f_3(n) = n^3 + 7n^2$

$f_4(n) = 2.023^n$

$f_5(n) = n\ log\ n$

$f_6(n) = n * \sqrt[3]{n^2}$

Now do the followings:

a. Write a correct asymptotic upper bound for each of the above.
b. Sort the functions in ascending order of their growth rate, assuming n is significantly large. Just write the sorted order, no need to show any simulation.

Ans 1(b)(a):Yet to be covered!
And 1(b)(b): $f_1 < f_5 < f_6 < f_3 = f_2 < f_4$

2023 Fall Mid Set B

1a. Explain the time complexity of the following code snippet in regards of the Big-O notation:

```
1. for (i=0; i<n; i+=4) {
2.         for (j=1; j<n; j*=2) {
3.             for (k=0; k<20; k++) {
```

```
4.                  print("Am I still not 30?!!");
5.              }
6.              print("Why, God, why? We had a Deal!");
7.              for (m=n; m>0; m-=4) {
8.                  print("Could you BE more dramatic?");
9.              }
10.         }
11. }
```

Ans:

First Loop : n/4
2nd Loop : log2n
3rd Loop : 20
4th Loop : n/4

n/4 * $\log_2 n$ * (20 + n/4)
≈ n * $\log_2 n$ * n
≈ n^2 * $\log_2 n$

1b. Consider the following functions.

$f_1(n) = (\log n)^{2000}$

$f_2(n) = n^3 \log_n(n^n)$

$f_3(n) = n^3 + 7n^2$

$f_4(n) = 4^n$

$f_5(n) = n \log n$

$f_6(n) = n * \sqrt{n}$

Now do the followings:

a. Write a correct asymptotic upper bound for each of the above.
b. Sort the functions in ascending order of their growth rate, assuming n is significantly large. Just write the sorted order, no need to show any simulation.

Ans 1(b)(a):Yet to be covered!
And 1(b)(b): $f_1 < f_5 < f_6 < f_3 < f_2 < f_4$