Week 1 Quiz

**Due date: 2024-02-07, 23:59 IST.**

**Your last recorded submission was on 2024-01-31, 08:51 IST**

All questions carry equal weightage. You may submit as many times as you like within the deadline. Your final submission will be graded.

***2 points***

***Q1***

An image processing application begins with two n×n matrices A and B. The first phase of preprocessing the inputs takes $O(n^2)$ steps for each of A and B. The second step involves a convolution of A and B to yield a new matrix C in time $O(n^3)$. This is followed by an edge detection phase that takes times $O(n^2)$ for matrix C. What is the most accurate and concise description of the complexity of the overall algorithm?

- $O(n^2)$
- $O(n^3)$
- $O(n^2+n^3)$
- $O(n^5)$

**EXPLAINATION:**

The bottleneck of the program is in second step where the time complexity is $O(n^3)$. Hence the time complexity of program is **$O(n^3)$.**

***Q2)***

***2 points***

2)We are trying to determine the worst case time complexity of a library function that is provided to us, whose code we cannot read. We test the function by feeding large numbers of random inputs of different sizes. We find that for inputs of size 400 and 4,000, the function always returns well within one second, but for inputs of size 40,000 it sometimes takes a couple of seconds and for inputs of size 400,000 it sometimes takes a few minutes. What is a reasonable conclusion we can draw about the worst case time complexity of the library function? (You can assume, as usual, that a typical desktop PC performs $10^9$ basic operations per second.)

- O(n log n)
- O($n^2$)
- O($n^3$)
- O($n^3$ log n)

**EXPLAINATION:**

After calculating by taking the provided input we can easily get that for **O($n^2$)**:

a) 400*400 / $10^9$ ~ 0.00016 sec

b) 4000*4000 / $10^9$ ~ 0.016 sec

c) 40000*40000 / $10^9$ ~ 1.6 sec

d)400000*400000 / $10^9$ ~ 160 sec ~ 2.67 min

***Q3)***

***2 points***

Suppose f(n) is $252n^3+164n^2+507$ and g(n) is $n^4 + 5n + 12$. Let h(n) be a third, unknown function. Which of the following is **not** possible.

- (•) h(n) is O(f(n)) and h(n) is also O(g(n))
- ( ) h(n) is O(f(n)) but h(n) is not O(g(n))
- ( ) h(n) is O(g(n)) but h(n) is not O(f(n))
- ( ) h(n) is not O(f(n)) and h(n) is also not O(g(n))

**EXPLAINATION:**

**The statement h(n) is O(f(n)) and h(n) is also O(g(n)) is not possible.** Here's why:

Big O notation:

- O(f(n)) represents the set of functions that grow at most as fast as f(n) for sufficiently large values of n.
- In other words, there exists a positive constant c and a value n0 such that $|h(n)| \leq c * |f(n)|$ for all $n \geq n0$.

Analyzing the functions:

- f(n) has the highest order term of n^3, signifying cubic growth.
- g(n) has the highest order term of n^4, signifying quartic growth.

Since n^4 grows faster than n^3 for all values of n greater than 1, it is impossible for a function to belong to O(f(n)) and O(g(n)) simultaneously. No constant "c" can limit both cubic and quartic growth with the same effectiveness.

Therefore, the correct answer is h(n) is O(f(n)) and h(n) is also O(g(n)).

The other statements are all possible:

- h(n) is O(f(n)) but h(n) is not O(g(n)): Consider h(n) = n^3, which grows as fast as f(n) but slower than g(n).
- h(n) is O(g(n)) but h(n) is not O(f(n)): Consider h(n) = n^4 + 1, which grows slower than f(n) but as fast as g(n).
- h(n) is not O(f(n)) and h(n) is also not O(g(n)): Consider h(n) = 2^n, which grows exponentially and dominates both f(n) and g(n).

Remember, Big O notation only considers the dominant term of a function's growth for sufficiently large values of n.

**<SRC : GOOGLE BARD>**

***Q4)***

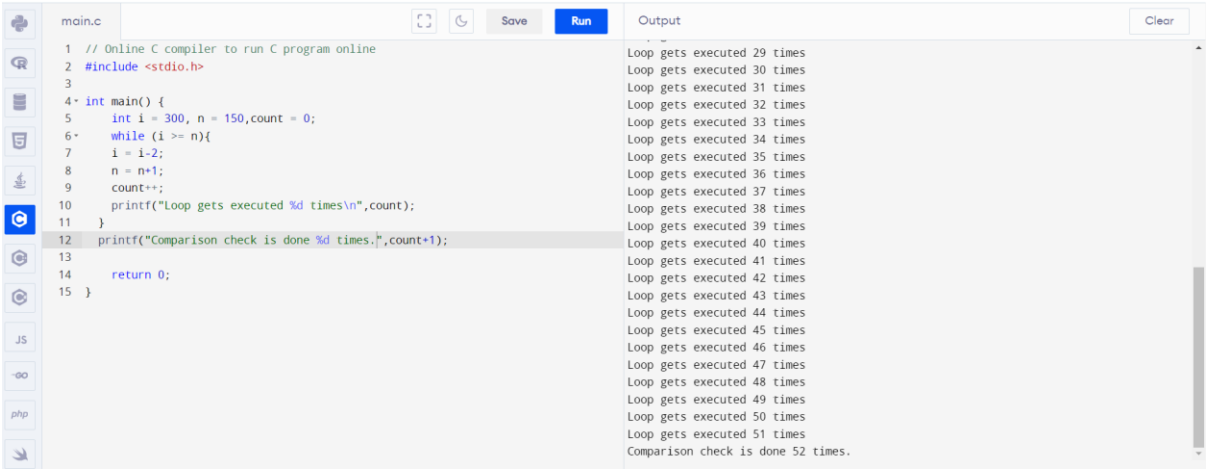
***2 points***

How many times is the comparison i >= n performed in the following program?

```
int i = 300, n = 150;
main(){
  while (i >= n){
    i = i-2;
    n = n+1;
  }
}
```

- 50
- 51
- 52
- 53

**EXPLAINATION:**




Don’t get confused with while loop , Loop gets executed 51 times and after that one more comparison occurs where i ! >= n . Answer is **52**.

**Q5)**

***2 points***

If T(n) is $O(n^2 \sqrt{n})$ which of the following is **false**?

- T(n) is $O(n^2 \log n)$
- T(n) is $O(n^3)$
- T(n) is $O(n^3 \log n)$
- T(n) is $O(n^4)$

You may submit any number of times before the due date. The final submission will be considered for grading.

**EXPLAINATION:**

All others are time complexities greater than $O(n^2 \sqrt{n})$. **$O(n^2 \log n)$** is having less time complexity and is false statement. **ANS: T(n) is $O(n^2 \log n)$**

IF YOU HAVE ANY OTHER (CORRECT ANSWER && CORRECT EXPLAINATION) THEN YOU CAN MESSAGE ME ON LINKEDIN: https://www.linkedin.com/in/sujallimje/.

THIS FILE WAS UPLOADED ON GITHUB: https://github.com/sujallimje/NPTEL-DAA-ASSIGNMENT-2024