

Question 1

(a) (5 points) Show that $f(x) = x^2 + 4x$ is $O(x^2)$.

- ☐ $x^2 + 4x \leq c |x^2|$
- ☐ $x^2 + 4x \leq x^2 + 4x^2 \quad \forall x > 1$
- ☐ $x^2 + 4x \leq 5x^2 \quad \forall x > 1$
- ☐ $x_0 = 1$ and $c = 5$

Therefore, $f(x) = O(x^2)$.

(b) (5 points) Show that $f(x) = x^2$ is NOT $O(\sqrt{x})$.

If $x^2 \leq c\sqrt{x}$;

$x^{3/2} \leq c$;

As $x \rightarrow \infty$, $x^{3/2} \rightarrow \infty$, but c is a constant.

Therefore, $f(x) = x^2$ is NOT $O(\sqrt{x})$.

Note:

$x^2 \leq x^2 \quad \forall x > 1$

$x^2 \leq x^2 \quad \forall x > 1$

$x_0 = 1$ and $c = 1$

(c) (5 points) Show that $f(x) = x$ is $\Omega(\log x)$.

$x \Rightarrow c \log x$

$2^x \Rightarrow cx$

$x_0 = 1, c = 2$

Therefore, $f(x) = x$ is $\Omega(\log x)$.

(d) (10 points) Show that $f(x) = (2x^2 - 3)/((3x^4 + x^3 - 2x^2 - 1))$ is $\Theta(x^{-2})$.

$$f(x) = (2x^2 - 3)/((3x^4 + x^3 - 2x^2 - 1))$$

$2x^2 - 3$ is nearly equal to $2x^2$;

$3x^4 + x^3 - 2x^2 - 1$ is nearly equal to $3x^4$;

$$f(x) \approx 2/3x^2;$$

Big-O:

$$f(x) \leq c g(x), x > x_0$$

$$2/3x^2 \leq c x^{-2}$$

Therefore, $c = 1$ and $x_0 = 1$, $O(x^{-2})$;

Big-Ω:

$$f(x) \geq c g(x), x > x_0$$

$$2/3x^2 \geq c x^{-2}$$

Therefore, $c = 1/3$ and $x_0 = 1$, $\Omega(x^{-2})$;

Thus, that $f(x) = (2x^2 - 3)/((3x^4 + x^3 - 2x^2 - 1))$ is $\Theta(x^{-2})$.

Question 2 (15 points). Please rank the following functions based on their $O(\cdot)$ complexity of running time. The function that has the least complexity should be ranked 1. Please explain your answer to get full credit.

$$f_1(x) = x \log_2 x$$

$$f_2(x) = 3^x$$

$$f_3(x) = \sqrt{x}$$

$$f_4(x) = x!$$

$$f_5(x) = 2^x$$

Rank:

$$f_3(x) \rightarrow f_1(x) \rightarrow f_5(x) \rightarrow f_2(x) \rightarrow f_4(x)$$

Explain my answer(Big-O):

$$f_3(x) = \sqrt{x} \leq c\sqrt{x} \quad \forall x > 1 \Rightarrow c = 2; x_0 = 1$$

Big-O Complexity is $O(\sqrt{x})$, so it is slower than x .

$$f_1(x) = x \log_2 x \leq cx^2 \quad \forall x > 1 \Rightarrow c = 1; x_0 = 1$$

Big-O Complexity is $O(x \log_2 x)$, so it is faster than x .

$f_5(x)$ and $f_2(x)$ belong to n^x type.

$$f_5(x) = 2^x \Rightarrow c x^2 \quad \forall x > 4 \Rightarrow c = 1; x_0 = 4$$

Therefore, Big- Ω for 2^x is x^2 .

Big-O Complexity is $O(n^x)$, so 3^x is faster than 2^x

$$f_4(x) = x! \Rightarrow c 3^x \quad \forall x > 4 \Rightarrow c = 1; x_0 = 7$$

Therefore, Big- Ω for $x!$ is 3^x .

And then,

Big-O

$$f_4(x) = x! \leq cx! \quad \forall x > 1 \Rightarrow c = 2; x_0 = 1$$

Result of Rank:

$$f_3(x) \rightarrow f_1(x) \rightarrow f_5(x) \rightarrow f_2(x) \rightarrow f_4(x)$$

Question 3 :

(a) (15 points) Please describe an efficient algorithm in English using a data structure such as array / linked list / stack / queue to solve this problem.

1. Create a new Stack:

- It can store opening parenthesis ({, (, [) when they are encountered.

2. Traverse the String

- Use “for” to loop through each character into string.
- If it’s an opening parenthesis, “push” it into “stack”.
- If it’s a closing parenthesis:

First, check if the stack is empty.

Secondly, “pop” the top element and check if it matches the closing parenthesis.

Finally, check if the stack is empty.

(b) (5 points) What is the asymptotic upper bound of complexity of running time for your algorithm?

Stack<Character> queue = new Stack<>();	time	
	1	
for(char character : string.toCharArray()){	n	
if (character == '{' character == '(' character == '[') {	k	Number of opening brackets
queue.push(character);	k	Number of opening brackets
}		
else if(character == '}' character == ')' character == ']'){	k	Number of closing brackets
if (queue.isEmpty()) {	k	Number of closing brackets
return false;	1	
}		
char left = queue.pop();	k	Number of closing brackets
if (!isMatch(left, character)) {	k	Number of closing brackets
return false;	1	
}		
}		
return queue.isEmpty();	1	

Big-O:

$$f(x) = n + 6k + 4 \quad (k < n) \approx n + 6n + 4 \leq 7n + 4 \quad \forall n > 1 \Rightarrow c = 11; x_0 = 1$$

f(n) grows as O(n)

(c) (40 points) Please write a program in either Python / Java / C++ that realizes your algorithm in (a). To receive full credit, please structure your code, write comments and show the output for the above two examples.

Example1: I { love [the { rains } ()] }

```
xinggchen@Xinggs-MacBook-Pro HW1 % cd /Users/xi  
ualMachines/jdk-21.jdk/Contents/Home/bin/java -X  
orkspaceStorage/5b186955a9a9c5c41e227983c8196a36  
The output is true
```

Example2: "I { love [the { rains] () }";

```
xinggchen@Xinggs-MacBook-Pro HW1 %  
ualMachines/jdk-21.jdk/Contents/  
orkspaceStorage/5b186955a9a9c5c4  
The output is false
```

