Practice Quiz - Big O Complexity

Started: Apr 8 at 2:13pm

Quiz Instructions

This quiz is not an actual quiz and will not be counted toward the final grade. It is provided to test whether you have grasped the topic.

Be sure to perform the required readings assigned on this topic.

Question 1 — 1 pts

Arrange the following expressions from slowest to fastest growth rate.

$4n\ log\ n + 6n$
2^n
$15.5n^2 + 100n$
$40(log\ n)^2$
$3n!$
$12n^7$
$log(log\ n)$
$log\ n$
$sqrt(n)$
$100n$
546

Please use the table provided here:
https://en.wikipedia.org/wiki/Time_complexity#Table_of_common_time_comple

(https://en.wikipedia.org/wiki/Time_complexity#Table_of_common_time_complexities)

Edit View Insert Format Tools Table

12pt Paragraph

p 0 words </>

Question 2

1 pts

Determine the big O complexity for the following function, which represents the number of steps required for some algorithm.

$F(n) = 2\sqrt{n} + 5n * log\ n + 100n^2$

- $O(100n^2)$
- $O(n)$
- $O(n\ log\ n)$
- $O(n^2)$

Question 3

1 pts

Determine the big O complexity for the following function, which represents the number of steps required for some algorithm.

$T(n) = 3(2^n) + n^8 + 1024n$

- ○ $O(3(2^n))$
- ○ $O(2^n)$
- ○ $O(n^8)$
- ○ $O(n)$

Question 4

1 pts

Determine the big O complexity for the following function, which represents the number of steps required for some algorithm.

$G(n) = 3log\ n! + 54log(log\ n) + 20(log\ n)^2$

- ○ $O(log(log\ n))$
- ○ $O(log\ n)$
- ○ $O(n\ log\ n)$
- ○ $O((log\ n)^2)$

Question 5

1 pts

Evaluate the following code segment and determine the big O complexity:

```
def sum(n):
    sum = 0
    i = n
    while i > 0:
        sum += i
        i = i // 2
    return sum
```

- ○ $O(1)$

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

Question 6

1 pts

Suppose an algorithm is $O(logn)$, where n is the input size. If the size of the input is doubled, how will the execution time change?

Edit View Insert Format Tools Table

12pt Paragraph B I U A T²

p 0 words </>

Question 7

1 pts

Evaluate the following function and determine the big O complexity:

```
def disjoint1(A, B, C):
    for a in A:
        for b in B:
            for c in C:
                if a == b == c:
                    return False # a common value is found
    return True # the sets are disjoint
```

- ○ $O(n^3)$
- ○ $O(n^2)$
- ○ $O(n)$
- ○ $O(1)$

Question 8

1 pts

Evaluate the following code segment and determine the big O complexity:

```
test = 0
for i in range(n):
   test = test + 1

for j in range(n):
   test = test - 1
```

- ○ $O(n^2)$
- ○ $O(1)$
- ○ $O(log(n))$
- ○ $O(n)$

Question 9

1 pts

Evaluate the following code segment and determine the big O complexity:

```
for i in range( n ) :
    if i % 3 == 0 :
        sum = 0
        j = n
        while j > 0 :
            sum += j
            j = j // 2
    elif i % 2 == 0 :
        for j in range( 5 ) :
            sum += j
    else :
        for j in range( n ) :
            sum += j
```

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

Question 10

1 pts

Determine the big O complexity of the following function:

```
def example( n ):
  count = 0
  for i in range( n ) :
    for j in range( 125 ) :
      count += 1
  return count
```

- ○ $O(n^2)$
- ○ $O(n)$
- ○ $O(1)$

- ○ $O(125 * n)$

Question 11

1 pts

A program that uses an $O(n^2)$ algorithm will always take longer to run than a program that uses an $O(n\ log\ n)$ algorithm.

- ○ True
- ○ False

Question 12

2 pts

Show that $\sum_{i=1}^{n} i^2$ is $O\left(n^3\right)$.

Edit View Insert Format Tools Table

12pt Paragraph

p 0 words

Not saved Submit Quiz