Newton School

Refer a friend

Back

Number of questions:
21 Questions

Test Time:
40 minutes

Test Date:
Oct 1, 2021

Your Score:
21/21

Help

Question 1/21

What is the running time complexity of the following code snippet?

```
int value = 0;
for(int i=0; i < n; i++)
for(int j=0; j < i; j++)
value += 1;
```

- n
- (n+1)
- n(n-1)/2 ✓
- n(n+1)/2

Question 2/21

Consider the following functions:

$f(n) = 2^n$
$g(n) = n!$
$h(n) = n^{\log n}$

Which of the following statements about the asymptotic behavior of f(n), g(n), and h(n) is true?

- f(n) = O(g(n)); g(n) = O(h(n))
- f(n) = Ω (g(n)); g(n) = O(h(n))
- g(n) = O(f(n)); h(n) = O(f(n))
- h(n) = O(f(n)); g(n) = Ω (f(n)) ✓

Question 3/21

In a competition, four different functions are observed. All the functions use a single for
loop and within the for loop, the same set of statements are executed. Consider the following for
loops:

(A) for(i = 0; i < n; i++)
(B) for(i = 0; i < n; i += 2)
(C) for(i = 1; i < n; i *= 2)
(D) for(i = n; i > -1; i /= 2)

If n is the size of the input (positive), which function is most efficient(if the task to be performed is
not an issue)?

- A
- B
- C ✓
- D

Question 4/21

Which of the following does NOT belong to the family of notations?

- Big (O)
- Big (Ω)
- Big (θ)
- Big (⋈) ✓

Question 5/21

Two main measures for the efficiency of an algorithm are:

- Processor and memory
- Complexity and capacity
- Time and space ✓
- Data and space

Question 6/21

Consider the following function:

```
static int fun1(int n)
{
int i, j, k, p, q = 0;

for ( i = 1; i < n; ++i)
{
p = 0;
for ( j=n; j>1; j=j/2)
++p;
for ( k=1; k < p; k =k*2)
++q;
}
return q;
}
```

- O(n)
- n (logn)^2
- n^3
- nlogn
- nlog(logn) ✓

Question 7/21

Time Complexity

$\sum_{i=0}^{n}$

and the following choices for X:

I. $\Theta(n^4)$
II. $\Theta(n^5)$
III. $O(n^5)$
IV. $\Omega(n^3)$

The equality above remains correct if X is replaced by:

- Only I
- Only II
- I or III or IV but not II ✓
- II or III or IV but not I

Question 8/21

What is the time complexity of fun( )?

```
int fun(int n)
{
int count=0;
for(int i=0; i < n; i++)
for(int j=I=i; j > 0; j--)
count= count+1;
return count;
}
```

- Theta(n)
- Theta(n^2) ✓
- Theta(nlogn)
- Theta(nlog(nlogn))

Question 9/21

Let A[1, ..., n] be an array storing a bit (1 or 0) at each location, and f(m) is a function whose time complexity is θ(m). Consider the following program fragment written in a Java like language:

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

```
{
if (A[i] == 1)
counter++;
else {
f(counter);
counter = 0;
}
}
```

The complexity of this program fragment is:

- $\Omega(n^2)$
- $\Omega(n \log n)$ and $O(n^2)$
- $\theta(n)$ ✓
- $O(n)$

Question 10/21

Algorithm A and B have a worst-case running time of O(n) and O(logn), respectively. Therefore, algorithm B always runs faster than the algorithm A.

- True
- False ✓

Question 11/21

Which of the following is not $O(n^2)$?

- $(15^{10}) * n + 12099$
- $n^{1.98}$
- $n^3 / (\text{sqrt}(n))$ ✓
- $(2^{20}) * n$

Question 12/21

There are 4 different algorithms. A1, A2, A3, A4 to solve a given problem with the order
log(n), log(log(n)), nlog(n), n/log(n) respectively. Which is the best algorithm?

- A1
- A2 ✓
- A3
- A4

Question 13/21

The minimum number of comparisons required to find the minimum and the maximum of
100 numbers is _____.

- 148 ✓
- 147
- 146
- 140

Question 14/21

Which of the given options provides the increasing order of asymptotic complexity of functions f1, f2, f3 and f4?

$f1(n) = 2^n$
$f2(n) = n^{\wedge(3/2)}$
$f3(n) = nLogn$
$f4(n) = n^{\wedge(Logn)}$

- f3, f2, f4, f1 ✓
- f3, f2, f1, f4
- f2, f3, f1, f4
- f2, f3, f4, f1

Question 15/21

How is time complexity measured?

- By counting the number of statements in an algorithm
- By counting the number of primitive operations performed by the algorithm ✓

performed by the algorithm on a given input size

By counting the size of data input to the algorithm

None of the above

Question 16/21

What is the Big O time complexity of the following?

```
for(int i=0; i < n; i++)
{
   // body of the first loop
   for(int j=0; j < m; j++)
   {
    // body of the second loop
   }
}
```

O(n)

O(m)

O(nm) ✓

O(n+m)

Question 17/21

Which of the following case does not exist in complexity theory:

Best Case

Worst case

Average case

Null Case ✓

Question 18/21

What is the time complexity of the below function?

```
void fun(int n, int arr[ ])
{
int i = 0, j = 0;
for(; i < n; ++i)
while(j < n && arr[i] < arr[j])
j++;
}
```

O(n) ✓

$O(n^2)$

O(nlogn)

$O(n(logn)^2)$

Question 19/21

What is time complexity of fun( )?

```
int fun(int n)
{
int count=0;
for (int i= n; i> 0; i/=2)
for(int j=0; j< i; j++)
count+= 1;
return count;
}
```

$O(n^2)$

O(nlogn)

O(n) ✓

O(nlog(nlogn))

Question 20/21

What will be the time complexity for the following fragment of code?

```
for(int i=0; i < n; i++)
i*=k;
```

O(n)

O(k)

O($\log_n k$)

O($\log_k n$) ✓

Question 21/21

The increasing order of following functions in terms of asymptotic complexity is:

f1(n) = $n^{0.999999}$ log n
f2(n) = 10000000n
f3(n) = 10000000^n
f4(n) = n^2

f1(n); f4(n); f2(n); f3(n)

f1(n); f2(n); f3(n); f4(n)

f2(n); f1(n); f4(n); f3(n)

f1(n); f2(n); f4(n); f3(n) ✓

For any queries, you can reach us at
support@newtonschool.co