Back

Time and Space Complexity - Revision

Number of questions:
10 Questions

Test Date:
Dec 30, 2022

Your Score:
21/26

Question 1/10
Consider the following two functions. What are the time complexities of the functions?

```
int fun1(int n)
{
if (n <= 1) return n;
return 2*fun1(n-1);
}
int fun2(int n)
{
if (n <= 1) return n;
return fun2(n-1) + fun2(n-1);
}
```

- O(2^n) for both fun1() and fun2()
- O(n) for fun1() and O(2^n) for fun2() ✓
- O(2^n) for fun1() and O(n) for fun2()
- O(n) for both fun1() and fun2()

Question 2/10
The minimum number of comparisons required to find the minimum and the maximum of
100 numbers is _____.

- 147.1 to 148.1 ✓
- 145.1 to 146.1
- 140 to 146
- 140 to 147

Question 3/10
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) ✓

Question 4/10
Consider equality:

$$\sum_{i=0}^{n} i^3 = X$$

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 5/10

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^(3/2)

f3(n) = nLogn

f4(n) = n^(Logn)

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

Question 6/10

Consider the following functions:

f(n) = 3n√x

g(n) = 2√x log2n

h(n) = n!

Which of the following is true?

- h(n) is O(f(n))
- h(n) is O(g(n))
- g(n) is not O(f(n))
- f(n) is O(g(n)) ✓

Question 7/10

Consider the following program fragment for reversing the digits in a given integer to obtain
a new integer. Let n = $D_1D_2...Dm$

```
int n, rev;
rev = 0;
while (n > 0)
{
rev = rev*10 + n%10;
n = n/10;
}
```

The loop invariant condition at the end of the ith iteration is:

- n = $D_1D_2....Dm-i$ and rev = $DmDm-1...Dm-i+1$ ✓
- n = $Dm-i+1...D_{m-1}Dm$ and rev = $D_{m-1}....D_2D_1$
- n!= rev
- n = $D_1D_2....Dm$ and rev = $DmD_{m-1}...D_2D_1$

Question 8/10

What is the time complexity of the following code:-

```
void fun(){
Int cnt=0;
for(int i=0; i < n ;i++){
n=n/2;
cnt++;
}
}
```

- O(log(n)) ✓
- O(log(log(n))) ✗

O(n)

O(1)

Question 9/10

The running time of the algorithm is represented by following recurrence relation:

$T(n) = n$, if $n \leq 3$

$= T(n/3) + Cn$, otherwise

Which is the time complexity?

$\Theta(n)$ ✓

$\Theta(n\log n)$

$\Theta(n^2)$

$\Theta(n^2 \log n)$

Question 10/10

What is the time complexity of the following recursive function?

```
int Dosomething (int n) {
if(n≤2)
return 1;
else
return (Dosomething (floor(sqrt(n))) + n);
}
```

$\Theta(n^2)$

$\Theta(n\log_2 n)$

$\Theta(\log_2 n)$

$\Theta(\log_2 \log_2 n)$ ✓