



Please read the instructions carefully:

- \* Mark your answers on the provided OMR answer sheet only.
- \* (14 Questions x 1 mark = 14) + (3 Questions x 2 marks = 6) = 20 marks
- \* Negative marking = 25% of total marks for incorrect answer per question.
- \* Use of calculators or electronic devices, mobile phones etc. is not permitted.

Student's Name:

Roll No:

Time: 1 hr

1. What is the correct arrangement of the following functions in increasing order of asymptotic complexity?. Start with the lower to higher.

- a.  $n, \log n, n \log n, n^2$    b.  $\log n, n, n \log n, n^2$    c.  $n^2, n \log n, n, \log n$    d.  $\log n, n \log n, n, n^2$

2. Let  $f(n) = 3n^2 + 2n + 1$ . Which of the following is true?

- a.  $f(n) = \Omega(n^3)$    b.  $f(n) = o(n^2)$    c.  $f(n) = O(n)$    d.  $f(n) = \Theta(n^2)$

3. Which of the following statements is true?

- a. If  $f(n) = o(g(n))$ , then  $f(n) = \Theta(g(n))$
- b. If  $f(n) = O(g(n))$ , then  $g(n) = \Omega(f(n))$
- c. If  $f(n) = \omega(g(n))$ , then  $f(n) = O(g(n))$
- d. If  $f(n) = \Theta(g(n))$ , then  $f(n) = o(g(n))$

4. Let  $f(n) = n \log n$  and  $g(n) = n^{1.1}$ . Which of the following is the most appropriate?

- a.  $f(n) = O(g(n))$    b.  $f(n) = \Omega(g(n))$    c.  $f(n) = \Theta(g(n))$    d.  $f(n) = o(g(n))$

5. What is the time complexity of the following loop?

```
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j += 2)
        print(i, j);
}
```

- a.  $O(n^3)$    b.  $O(n)$    c.  $O(n^2)$    d.  $O(n \log n)$

6. What is the time complexity of the following loop?

```
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < i; j++)
        print(i, j);
}
```

- a.  $O(n)$    b.  $O(n^3)$    c.  $f(n) = O(n^2)$    d.  $O(n \log n)$

7. What is the time complexity of the following loop?

```
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j = j * 2)
        x = x+1;
}
```

- a.  $O(\log n)$    b.  $O(n)$    c.  $O(n \log n)$    d.  $O(n^3)$

8. What is the time complexity of the following loop?

```
for (int i = n; i >= 2; i =  $\sqrt{i}$ )
    // some constant-time operation
```

- a.  $O(\log n)$    b.  $O(n^3 \log n)$    c.  $O(n^2 \log n)$    d.  $O(\log \log n)$

9. What is the time complexity of the following C function? [2 marks]

```
int fun(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;
}
```

- a.  $O(n^2)$    b.  $O(n^2 \log n)$    c.  $O(n \log \log n)$    d.  $O(n \log n)$

10. In the standard 3-peg Tower of Hanoi with  $n$  disks, the minimum number of moves required to move from source to destination via auxiliary is:

- a.  $n!$    b.  $n^2$    c.  $2^n - 1$    d.  $2^n$

11. Solve  $T(n) = T(n/2) + \frac{n}{\log n}$ ,  $T(1) = \Theta(1)$  [2 marks]

- a.  $\Theta(\log n)$    b.  $\Theta\left(\frac{n}{\log n}\right)$    c.  $\Theta(n)$    d.  $\Theta(n \log n)$

12. Consider the recurrence relation  $T(n) = 2T(n/2) + n$ . What is the asymptotic complexity of  $T(n)$

- a.  $\Theta(n)$    b.  $O(n \log n)$    c.  $O(n)$    d.  $O(n^2)$

13. Which of the following pseudo code will print the digits of a given number in reverse?

**Note:**

- The modulo operator % produces the remainder. For example, the value of  $x \% n$  is the remainder when  $x$  is divided by  $n$ .
- $a // b$  gives the quotient when  $a$  is divided by  $b$ . For example,  $5 // 2$  gives 2.

```
a. def reverse_num(num):
    while num > 10:
        print(num % 10, end=" ")
        num = num / 10
```

- b. `def reverse_num(num):`  
     `while(num >= 10):`  
         `print(num % 10, end="")`  
         `num = num // 10`
- c. `def reverse_num(num):`  
     `while(num > 0):`  
         `print(num % 10, end="")`  
         `num = num // 10`
- d. `def reverse_num(num):`  
     `while(num >= 10):`  
         `print(num % 10, end="")`  
         `num = num // 10`

14. What type of additional memory is required for each recursive call?

- a. Heap   b. Stack   c. Queue   d. None

15. **Consider the following two functions:** [2 marks]

```
def fib(n):
    if n <= 1:
        return n
    else:
        return fib(n-1) + fib(n-2)

def fib_memo(n, memo={}):
    if n in memo:
        return memo[n]
    if n <= 1:
        memo[n] = n
    else:
        memo[n] = fib_memo(n-1, memo) + fib_memo(n-2, memo)
    return memo[n]
```

- `fib(5)` — How many recursive calls will occur?
- `fib_memo(5)` — How many recursive calls and memo hits will occur?

Tips - Draw recursion tree to understand this better

- |   |  |
|---|--|
| a. <code>fib(5)</code> : (recursion calls = 16) | <code>fib_memo(5)</code> : (recursion calls = 5) + (memo hits = 8) |
| b. <code>fib(5)</code> : (recursion calls = 10) | <code>fib_memo(5)</code> : (recursion calls = 8) + (memo hits = 5) |
| c. <code>fib(5)</code> : (recursion calls = 5)  | <code>fib_memo(5)</code> : (recursion calls = 5) + (memo hits = 9) |
| d. <code>fib(5)</code> : (recursion calls = 15) | <code>fib_memo(5)</code> : (recursion calls = 6) + (memo hits = 3) |

16. **Which of the following statements is true regarding asymptotic notations?**

- a. Little o notation denotes an upper bound that is not asymptotically tight.
- b. Big  $\Omega$  notation represents the worst-case complexity of an algorithm.
- c. Big O notation always provides a tight upper bound.
- d.  $\Theta$  notation represents only the lower bound of an algorithm's running time.

17. **Consider an unordered list of N distinct integers. What is the minimum number of element comparisons required to find an integer in the list that is NOT the largest in the list?**

- a. N   b. 2N-1   c. N-1   d. 1