UNIVERSIDAD DE MÁLAGA

**Analysis and Design of Algorithms**
Complexity and Sorting
($2^{o}$ – Comput Eng, Softw Eng, Comput Sci & Math)
E.T.S.I. INFORMÁTICA

# Essential Problems

1. Indicate which of the following statements are true:

   *a)* $n^2 \in O(n^3)$
   *b)* $n^3 \in O(n^2)$
   *c)* $2^{n+1} \in O(2^n)$
   *d)* $(n+1)! \in O(n!)$
   *e)* $f(n) \in O(n) \Rightarrow 2^{f(n)} \in O(2^n)$
   *f)* $3^n \in O(2^n)$
   *g)* $\log n \in O(\sqrt{n})$
   *h)* $\sqrt{n} \in O(\log n)$

   Repeat the above changing $O(\cdot)$ by $\Omega(\cdot)$.

2. Arrange the following functions in increasing order of growth.

$$(n-2)!,\ 4\ln(n+100)^{10},\ 2^{2n},\ 10^{-6}n^5+9n^3,\ (\ln n)^2,\ \sqrt[3]{n},\ 5^n,\ n+10^{10}$$

3. Solve the following recurrences and indicate their order of growth:

   *a)* $t(n) = 3t(n-1) + 4t(n-2)$ with $t(0) = 0$ and $t(1) = 1$
   *b)* $t(n) = 4t(n-1) - (n+5)3^n$ with $t(0) = 0$. Feb 2013
   *c)* $t(n) = t(n-1) + 2t(n-2) - 2t(n-3)$ with $t(0) = 106$, $t(1) = 100$, $t(2) = 100$.
   *d)* $t(n) = 2t(n/4) + \sqrt{n}$ with $t(1) = 1$. Sep 2018
   *e)* $t(n) = 4t(n/3) + n^2$ with $t(1) = 1$.

4. For each of the following recurrences, determine their order of growth using the Master Theorem. Then, find the exact solution for each of them (no need to compute values for the coefficients that depend upon initial conditions).

   *a)* $t(n) = 4t(n/2) + n^2$
   *b)* $t(n) = 2t(n/2) + n\log_2 n$
   *c)* $t(n) = 3t(n/2) + 5n + 3$
   *d)* $t(n) = 2t(n/2) + \log_2 n$
   *e)* $t(n) = 2t(\sqrt{n}) + \log_2 n$
   *f)* $t(n) = 5t(n/2) + (n\log_2 n)^2$

5. Find a recurrence (and solve it) for the complexity of the following algorithm:

```
public static int recursive (int n) {
    if ( n <= 1 )
        return 1;
    else
        return (recursive (n-1) + recursive (n-1));
}
```

6. Given an integer $x$ and an ordered integer array $A$ of length $n$ whose elements are all different:

*a*) Design an algorithm to determine whether there are two elements in $A$ whose sum is exactly $x$.

*b*) Determine the complexity of this algorithm.

*c*) Is your solution linear (i.e. $O(n)$)? If not, try to find such a solution.

## Additional Problems

7. Assuming $T_1 \in O(f)$ and $T_2 \in O(f)$, indicate which of the following statements are true:

    *a*) $T_1 + T_2 \in O(f)$.

    *b*) $T_1 - T_2 \in O(f)$.

    *c*) $T_1/T_2 \in O(1)$.

    *d*) $T_1 \in O(T_2)$

8. Obtain an expression of the computational cost $t(n)$ of the following piece of code, assuming multiplication is the basic operation.

    **for** $i \leftarrow 1$ **to** $n$ **do**
        **for** $j \leftarrow 1$ **to** $i$ **do**
            **for** $k \leftarrow j$ **to** $n$ **do**
                $r \leftarrow r \times k$
            **endfor**
        **endfor**
    **endfor**

9. Indicate for each of the following functions the class $\Theta(g(n))$ they belong to.

    *a*) $(n^2+1)^{10}$

    *b*) $\sqrt{10n^2+7n+3}$

    *c*) $2n\ln(n+2)^2 + (n+2)^2\ln(n/2)$

    *d*) $2^{n+1} + 3^{n-1}$

    *e*) $\lfloor \log_2 n \rfloor$

10. Indicate for each of the following pairs of functions whether they have the same order of growth or which of the functions grows faster.

    *a*) $n(n+1)$ and $2000n^2$

    *b*) $\log_2 n$ and $\ln n$

    *c*) $100n^2$ and $0{,}001n^3$

    *d*) $(\log_2 n)^2$ and $\log_2 n^2$

    *e*) $2^{n-1}$ and $2^n$

    *f*) $(n-1)!$ and $n!$

11. Compute the best and worst case complexity of the following algorithm for determining whether a matrix is symmetric or not.

```
public static boolean symmetric (int [] [] matrix, int n){
int i, j;
    boolean b;

    b = true;
    i = 0;
    while(i<n && b){
```

```
        j = i + 1;
        while(j<n  && b){
            b = (matrix [i][j] == matrix[j][i]);
            j ++;
        };
    i ++;
};
return b;
}
```

12. Given the following recursive algorithm for computing Fibonacci numbers:

    **func** Fib ($\downarrow n$: $\mathbb{N}$):$\mathbb{N}$
    **variables** $f$: $\mathbb{N}$
    **begin**
      **if** $n$<3 **then** $f \leftarrow 1$
      **else** $f \leftarrow$ Fib($n$−1)+Fib($n$−2)
      **endif**
      **return** $f$
    **end**

    *a*) Find a recurrence for the computational cost of the algorithm, assuming the sum is the basic operation.

    *b*) Solve the recurrence to obtain the exact number of operations performed by the algorithm.

13. Solve the following linear recurrence:

$$t(n) = \begin{cases} n & n \leqslant 1 \\ 5t(n-1) - 6t(n-2) & n > 1 \end{cases}$$

14. Let $f(n) \in \Theta(n^k)$, and let $t(n) = at(n/b) + f(n)$. Express the Master Theorem in this particular case.

15. Find a recurrence for the computational cost of the following algorithm (printing is assumed to be the basic operation), and provide a tight bound for this cost using the Master Theorem.

    **proc** A ($\downarrow n$: $\mathbb{N}$)
    **variables** $i$, $j$: $\mathbb{N}$
    **begin**
      **for** $i \leftarrow 1$ **to** $n$ **do**
        **for** $j \leftarrow 1$ **to** $i$ **do**
          print($i$, $j$, $n$)
        **endfor**
      **endfor**
      **if** $n$>0 **then**
        **for** $i \leftarrow 1$ **to** 4 **do**
          A($n$/2)
        **endfor**
      **endif**
    **end**

16. Apply the Master Theorem to determine the exact order of growth of the following recurrence: Jun 2008

$$t(n) = 3t(n/2) + n \log n$$

17. Solve the following recurrence: Jun 2008

$$t(n) = 2^n \left(t(n/2)\right)^2$$

with the initial condition $t(1) = 1$.

18. Given the recurrence: Sep 2009

$$t(n) = 3t(\sqrt[9]{n}) + \log n$$

Find its order of growth using the Master Theorem (**hint:** a change of variable is required first).

Jun 2010 19. Apply the Master Theorem to determine the order of growth of the following recurrence:

$$t(n) = 5t(\sqrt[25]{n}) + \sqrt{\log n}$$

Sep 2012 20. Solve the following recurrences:

*a*) Apply the Master Theorem to obtain the exact order of growth:

$$t(n) = 2t(\sqrt{n}) + \log n$$

*b*) Solve exactly, with the initial condition $t(1) = 1$.

$$t(n) = 2t(n-1) + 2$$

Feb 2014 21. Solve the following recurrence:

$$T(n) = \begin{cases} 0 & n = 0 \\ 2T(n-1) + n + 2^n & n > 0 \end{cases}$$

Feb 2016 22. Solve the following equation using the Master Theorem:

$$T(n) = 3T(n/4) + n \log n$$

Sep 2016 23. Solve the following recurrence:

$$T(n) = 4T(n/3) + n, n > 1$$

where $T(n) = 1$ for $n \leqslant 1$. Use the method you find most appropriate.

Feb 2018 24. Find the exact solution for the following recurrence:

$$T(n) = 7T(n/2) - 14T(n/4) + 8T(n/8) + n, n > 4,$$