

Preguntas ADA Primer Parcial

1. La complejidad temporal en el mejor de los casos de un algoritmo recursivo...

Las demás opciones son falsas.

2.

Pertenece $3n^2 + 3$ a $O(n^3)$?

si

3. Un algoritmo recursivo basado en el esquema divide y vencerás...

... será mas eficiente cuanto mas equitativo sea la división en subproblemas.

4. El coste temporal asintótico del fragmento

`s=0; for(i =0; i<n; i++) for(j=i; j<n; j++) s+= i*j;`

y del fragmento

`s=0; for(i =0; i<n; i++) for(j=i; j<n; j++) s+= i*i*j;` son...

...iguales.

5. Indica cual de estas tres expresiones es falsa:

☐ a. $\Theta(n/2) = \Theta(n)$

☐ b. $\Theta(n) \subseteq O(n)$

☒ c. $\Theta(n) \subseteq \Theta(n^2)$ ✓

6. Un problema de tamaño n puede transformarse en tiempo $O(n)$ en siete de tamaño $n/7$; por otro lado, la solución al problema cuando la talla es 1 requiere un tiempo constante. ¿Cual de estas clases de coste temporal asintotico es la mas ajustada?

$O(n \log n)$

7. Indica cuál es la complejidad en función de n , donde k es una constante(no depende de n) del fragmento siguiente:

```
for( int i = k; i < n - k; i++){  
    A[i] = 0;  
    for( int j = i - k; j < i + k; j++ )  
        A[i] += B[j];  
}
```

$O(n)$

8. Indica cuál es la complejidad en el peor caso de la función replace:

```
unsigned bound( const vector<int> &v ) {  
    for( unsigned i = 0; i < v.size(); i++ )  
        if( v[i] == '0' )  
            return i;  
    return v.size();  
}  
  
void replace( vector<int>& v, int c ) {  
    for( unsigned i = 0; i < bound(v); i++ )  
        v[i] = c;  
}
```

$O(n^2)$

9. Sea $f(n)$ la solución de la relación de recurrencia $f(n) = f(n/2) + 1$; $f(1) = 1$. Indica cuál de estas tres expresiones es cierta:

$f(n) \in O(\log(n))$

10. La versión de Quicksort que utiliza como pivote la mediana del vector...

... el hecho de que el vector estuviera previamente ordenado o no, no influye en la complejidad temporal de este algoritmo.

11. Indica cuál es la complejidad, en función de n , del siguiente fragmento de código:

```
s=0; for(i=0;i<n;i++) for(j=i;j<n;j++) s+=i*j;
```

$O(n^2)$

12. Un problema de tamaño n puede transformarse en tiempo $O(n)$ en siete de tamaño $n/7$; por otro lado, la solución al problema cuando la talla es 1 requiere un tiempo constante. ¿cual de estas clases de coste temporal asintótico es la más ajustada?

$O(n \log n)$

13. Indica cuál es la complejidad, en función de n , del siguiente fragmento de código:

```
int a = 0;  
  
for(int i=0;i<n;i++)  
    for( int j=i;j>0;j/=2)  
        a += A[i][j];
```

$O(n \log n)$

14. ¿Cuál es la complejidad temporal de la siguiente función recursiva?

```
unsigned desperdicio (unsigned n){
    if (n<=1)
        return 0;
    unsigned sum = desperdicio (n/2) + desperdicio (n/2);
    for (unsigned i=1; i<n-1; i++)
        for (unsigned j=1; j<=i; j++)
            sum+=i*j;
    return sum;
}
```

$O(n^2)$

14.¿Cuál es la complejidad temporal de la siguiente función recursiva?

```
unsigned desperdicio (unsigned n){
    if (n<=1)
        return 0;
    unsigned sum = desperdicio (n/2) + desperdicio (n/2);
    for (unsigned i=1; i<n-1; i++)
        for (unsigned j=1; j<=i; j++)
            for (unsigned k=1; k<=j; k++)
                sum+=i*j*k;
    return sum;
}
```

$O(n^3)$

15. Considerad estos dos fragmentos:

$s=0$; for($i=0$; $i<n$; $i++$) $s+=i$; y $s=0$; for($i=0$; $i<n$; $i++$) if($a[i]!=0$) $s+=i$;

y un array $a[i]$ de números enteros. Indicad cuál de estas tres afirmaciones es cierta:

El coste temporal asintótico, tanto en el caso mejor como en el caso peor de los dos programas es el mismo.

16. Los algoritmos de ordenación Quicksort y Mergesort tienen en común...

... que aplican la estrategia de divide y vencerás.

17. Dada la siguiente relación de recurrencia. ¿Que cota es verdadera?

$$f(n) = \begin{cases} 1 & n=1 \\ \sqrt{n} + 3f(n/3) & n>1 \end{cases}$$

$f(n) \subset O(n)$

18. Un problema de tamaño N puede transformarse en tiempo $O(N^2)$ en nueve de tamaño $N/3$; por otro lado, la solución al problema cuando la talla es 1 requiere un tiempo constante.

¿cual de estas clases de coste temporal asintótico es la más ajustada?

$O(n^2 \log n)$

19. ¿Cual de estas tres expresiones es falsa?

$$\begin{aligned} 3n^2 + 1 &\in O(n^3) \\ n + n \log(n) &\in \Theta(n) \quad \checkmark \\ n + n \log(n) &\in \Omega(n) \end{aligned}$$

20. Indica cuál de estas tres expresiones es cierta:

$$\begin{aligned} O(n^2) &\subset O(2^{\log(n)}) \subset O(2^n) \\ O(n^2) &\subset O(2^{\log(n)}) \subseteq O(2^n) \\ O(2^{\log(n)}) &\subset O(n^2) \subset O(2^n) \quad \checkmark \end{aligned}$$

21. Para que la complejidad de un algoritmo presente caso mejor y peor distintos ...

... es condición necesaria que existan instancias distintas del problema con el mismo tamaño.

22. considerar la función siguiente:

```
int M( int i, int f) {
    if ( i == f )
        return i;
    else {
        e = v[ M( i, (i+f)/2 ) ];
        f = v[ M( (i+f)/2+1, f ) ];
        if (e < f)
            return e;
        else
            return f;
    }
}
```

Si la talla del problema viene dada por $n = f - i + 1$. ¿Cual es el coste temporal asintótico en el supuesto de que n sea una potencia de 2?

$O(n)$