TALLER 1

Preguntas:

1) No, para valores pequeños de n, el algoritmo B puede ser una mejor opción.

2) Porque la eficiencia es un elemento vital para el trabajo a realizar

3)
- Habilidad del programador
- Complejidad del problema
- Paradigma utilizado
- Salarios y Recursos
- Capacidad de la máquina
- Complejidad del código
- Sistema Operativo

Ejercicios:

1)
```
PALINDROMO(A)
n = length(A)
for i ∈ {1...⌊n/2⌋}
    if (A[i] != A[n-i+1])
        return false
return true
```

c_1 1

c_2 $a; a \in \{1 \ldots \lfloor n/2 \rfloor + 1\}$

c_3 $b; b \in \{1 \ldots \lfloor n/2 \rfloor\}$ ("a-1")

c_4 $1 \cdot \emptyset \equiv d$

c_5 $1 \cdot \emptyset \equiv e$

Costo: $c_1 + ac_2 + bc_3 + dc_4 + dc_5$

Mejor de los Casos: $a = 1, b = 1, d = 1, e = \emptyset$

$c_1 + c_2 + c_3 + c_4 = O(1)$

Peor de los Casos: $a = \lfloor n/2 \rfloor + 1, b = \lfloor n/2 \rfloor, d = \emptyset, e = 1$

$c_1 + c_2(\lfloor n/2 \rfloor + 1) + c_3(\lfloor n/2 \rfloor) + c_5 \leq c_1 + c_2(n/2 + 1) + c_3(n/2) + c_5$

$= O(n)$

2) DIAGONAL (M)

```
n = length(M)

for i ∈ {1...n}
    For j ∈ {1...n}
        if(i != j and M(i,j) != ∅)
            return false
return true
```

c_1 1

c_2 $a; a \in \{1 \dots n+1\}$

c_3 $\sum_i b; b \in \{1 \dots n+1\}$

c_4 $\sum_i c; c \in \{1 \dots n\}$

c_5 $1 : \emptyset \equiv d$

c_6 $1 : \emptyset \equiv e$

Costo: $c_1 + ac_2 + c_3 \sum_i b + c_4 \sum_i c + c_5 d + c_6 e$

Mejor de los Casos: $a=1; b=1; c=1; d=1; e=\emptyset$

$$c_1 + c_2 + c_3 \sum_{i=1}^{1} 1 + c_4 \sum_{i=1}^{1} 1 + c_5 = O(1)$$

Peor de los Casos: $a=n+1; b=n+1; c=n; d=\emptyset; e=1$

$$c_1 + c_2(n+1) + c_3 \sum_{i=1}^{n+1} (n+1) + c_4 \sum_{i=1}^{n+1} n + c_6 = O(n^2)$$

3) MERGE3(A, P, r)

```
if (p < r)
    q1 = ⌊length(A)/3⌋
    q2 = 2*q1
    MERGE3(A, P, q1)
    MERGE3(A, q1+1, q2)
    MERGE3(A, q2+1, r)
```

$\Theta(1)$

$\Theta(n) \leftarrow$ MERGE(A, P, q1, q2, r)

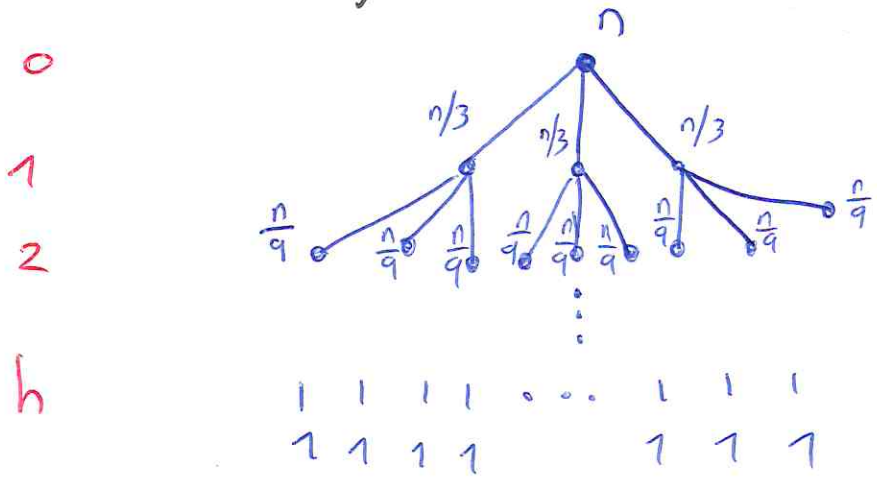
```
MERGE(A, P, q1, q2, r)
L = L[q1 - p + 2]
M = M[q2 - q1 + 2]
R = R[r - q2 + 2]
L[q1 - p + 2] = M[q2 - q1 + 2] = R[r - q2 + 2] = ∞
for x ∈ {P...q1}
    L[i] = A[x]
    i++
i = 1
for x ∈ {q1+1 ... q2}
    M[j] = A[x]
    j+1
k = 1
for x ∈ {q2+1 ... r}
    R[k] = A[x]
```

$\Theta(n)$

```
i=j=k=1
for x ∈ {p ... r}
    if (L[i] ≤ M[j])
        if (L[i] ≤ R[k])
            A[x] = L[i]
            i++
        else
            A[x] = R[k]
            k++
    else
        if (M[j] ≤ R[k])
            A[x] = M[j]
            j++
        else
            A[x] = R[k]
            k++
```

$O(n)$

$$T(n) = 3T(\lfloor n/3 \rfloor) + O(n)$$



Costos:

- dato: c
- nodo: $c\frac{n}{3^k}$
- nivel: $3^k\left(c\frac{n}{3^k}\right) = cn$
- árbol: $\sum_{k=0}^{h} cn$

$$\frac{n}{3^h} = 1 \rightarrow n = 3^h \rightarrow \log_3 n = h$$

$$\sum_{k=0}^{\log_3 n} cn = cn \sum_{k=0}^{\log_3 n} 1 = cn \log_3 n$$

$$= O(n \log_3 n)$$