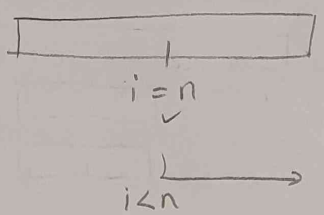
# PREGUNTA 3

JORGE DE GOYCOECHEA
20.538.979-2

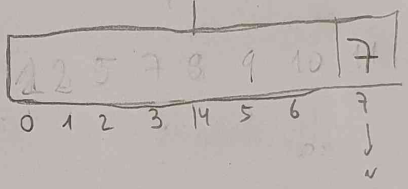
a)

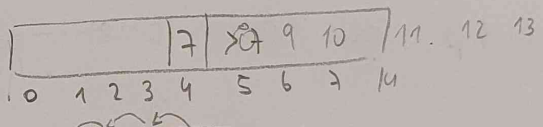
```
numerosMagicos(A, i, f)
    m = ⌊(i+f)/2⌋
    if m = A[m]: return m
    else if m > A[m]:
        numerosMagicos(A, i, m-1)
    else if m < A[m]:
        numerosMagicos(A, m+1, f)
```

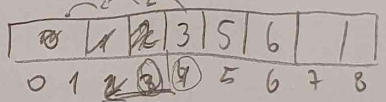
$O(\log(n))$




2 4 15
1 2 3 4 5

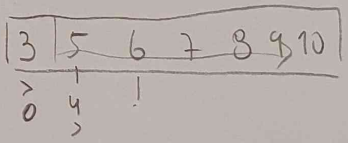


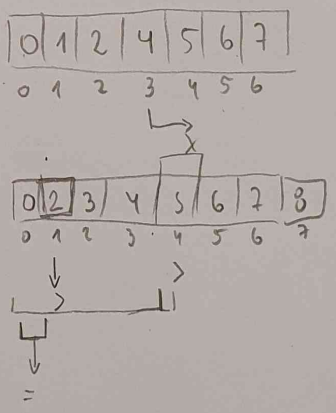




b)

```
numerosMagicos(A, i, f)
    m = ⌊(i+f)/2⌋
    if m = A[m]: return m
    elif m > A[m]:
        return numerosMagicos(A, i, m-1)
    elif m < A[m]:
        return numerosMagicos(A, m+1, f)
    else: return null
```



b) El peor caso de mi algoritmo es cuando no existe ningun numero mágico, o este se encuentra en "i" o "f".

Para cada iteración, se chequea si el valor central del array es un numero magico, si no lo es, se divide el array y se vuelve a chequear. Esto es asi hasta que quede un solo elemento en el sub-array.

Como se ejecuta dividiendo el tamaño hasta quedar en uno la complejidad del peor caso es de $O(\ln(n))$

y en comparacion su mejor caso es $O(1)$, con un numero magico en la mediana del arreglo.