

```

for i=0 to n {
    for j=1 to n {
        for k=1 to 4 {
            }
        }
    }
}

```

 = konstant
 = abhängig von
 instanz

$(n \cdot n) \cdot n \cdot 4 \cdot Tid(method)$

$O(n^2) \cdot Tid(method)$

```

recursiveMax(A, n) {
    if ^n=1 then } 3
        return A[0]
    return max(A[n-1], recursiveMax(A, n-1))
}

```

 = konstant

 = abhängig von
instanz

$$f(n) = \begin{cases} 3 & (n=1) \\ 7 + f(n-1) & (n > 1) \end{cases}$$

$$f(n) = 7 + f(n-1)$$

$$= 7 + (7 + f(n-2))$$

$$= 7 + 7 + (7 + f(n-3))$$

:

$$= 7 + \dots + 7 + f(1)$$

$$= \underbrace{7 + \dots + 7}_{n-1} + 3 = 7 \cdot (n-1) + 3 = 7n - 4$$

```

BinarySearch(A[0..N-1], value, low, high) {
    if (high < low)
        return not found
    mid = (low + high) / 2
    if (A[mid] > value)
        return BinarySearch(A, value, low, mid-1)
    else if (A[mid] < value)
        return BinarySearch(A, value, mid+1, high)
    else
        return mid
}

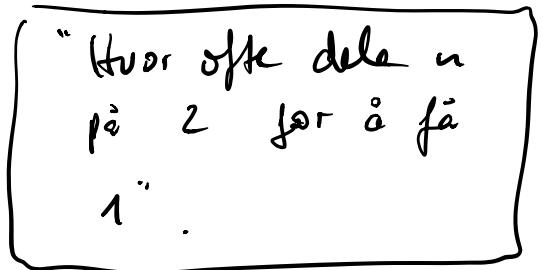
```

$$\begin{aligned}
f(n) &= c + f\left(\frac{n}{2}\right) \\
&= c + c + f\left(\frac{n}{4}\right) \\
&= c + c + c + f\left(\frac{n}{8}\right) \\
&\vdots \\
&= c + \dots + c + f(1) \\
&= c + \dots + c + c \\
&\quad \underbrace{\qquad\qquad\qquad}_{\log_2(n)} \\
&= c \cdot \log_2(n)
\end{aligned}$$

 = konstant
 = avhengig av
instans

$$2^{\log_2(n)} = n$$

"Hvor mange ganger
å gange 2 ned seg
solv for å få n".


 "Hvor ofte dele n
på 2 før å få
1".