Tutorial 4

Example 1

- The space/memory requirements ($S(n)$) of an algorithm is usually measured by investigating the size of the most expensive data structure used (how many memory cells are needed to store it). The characteristic data structure of the given algorithm fib(n) is `table`. In a call fib(n) the size of the table is $n+1$ hence $S(n)$ is $O(n)$.
- **fib**(n) =
  if $n = 0$ **then**
    $fib := 0$
  else
    $fib := 1$
  end if
  $fib_{-1} := 1$
  $fib_{-2} := 0$
  for $i = 2$ to n **do**
    $fib := fib_{-1} + fib_{-2}$
    $fib_{-2} := fib_{-1}$
    $fib_{-1} := fib$
  end for
  return fib

Example 2

Pre: $A[a..b]$ is an array such that $a \leq b$

res:= largest(a, b)

Post: res $\geq A[a..b]$ and res $\in A[a..b]$ and $A' = A$

largest(a,b) =
if $a = b$ **then**
  $res := A[a]$
else
  $mid := \lfloor \frac{a+b}{2} \rfloor$
  $sol_1 :=$ **largest**(a, mid)
  $sol_2 :=$ **largest**$(mid + 1, b)$
  $res :=$**max**(sol_1, sol_2)
end if
return res

Example 3

- Let n be the size of the considered array, i.e., $n = b - a + 1$.

  $W(1) = 1$

  $W(n) = 1 + 2W(\lceil \frac{n}{2} \rceil) \quad \text{for } n > 1$

- We solve the equation only in points where $n = 2^k$ for all $k = 0, 1, 2, \ldots$. Hence we consider the following recurrence equation.

  $W(2^0) = 1$

  $W(2^k) = 1 + 2W(2^{k-1}) \quad \text{for } k > 0$

- We shall use repeated substitutions to solve this system.

$$
\begin{aligned}
W(n) = W(2^k) &= 1 + 2W(2^{k-1}) \\
&= 1 + 2(1 + 2W(2^{k-2})) \\
&= 1 + 2 + 4W(2^{k-2}) \\
&= 2^0 + 2^1 + \ldots + 2^i W(2^{k-i}), \quad (0 \leq i \leq k) \\
&= 2^0 + 2^1 + \ldots + 2^k \underbrace{W(2^{k-k})}_{=W(2^0)=1}, \quad (i = k) \\
&= 2^0 + 2^1 + \ldots + 2^k \\
&= 2^{k+1} - 1 \\
&= 2^{(\log n)+1} - 1 \\
&= 2 \cdot 2^{\log n} - 1 \\
&= 2n - 1 \\
&= O(n)
\end{aligned}
$$

Example 4

$i := 1$
$j := 2$
while $max(i, j) < n$ **do**
  if i knows j **then**
    $i := max(i, j) + 1$
  else
    $j := max(i, j) + 1$
  end if
end while
$candidate := min(i, j)$

*** Now we check whether $candidate$ is indeed a celebrity ***

OK:=true; $i := 1$

```
while OK and i ≤ n do
   if candidate ≠ i then
      if candidate knows i then
         OK := false
      else
         if i does not know candidate then
            OK := false
         end if
      end if
   end if
   i:=i+1
end while
if OK=true then
   return ”candidate is a celebrity”
else
   return ”there is no celebrity”
end if
```