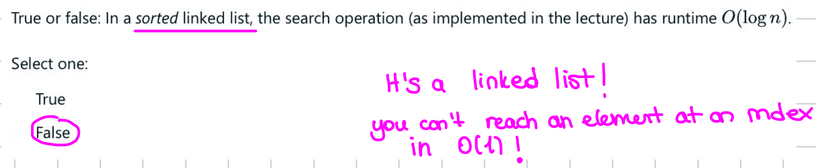
Quiz 7

1)



True or false: In a sorted linked list, the search operation (as implemented in the lecture) has runtime $O(\log n)$.

Select one:

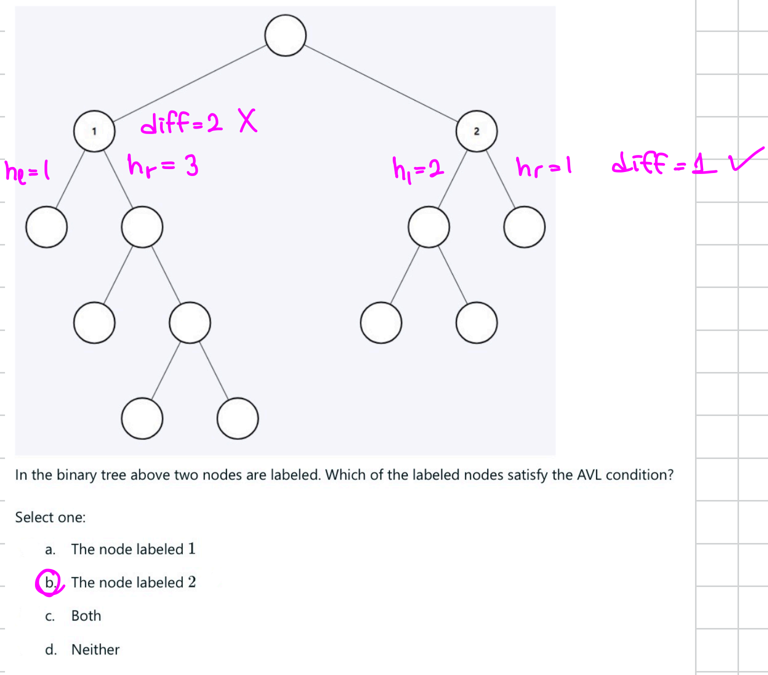
True

[✓ False]

It's a linked list!

you can't reach an element at an index in O(1)!

2)



In the binary tree above two nodes are labeled. Which of the labeled nodes satisfy the AVL condition?

Select one:

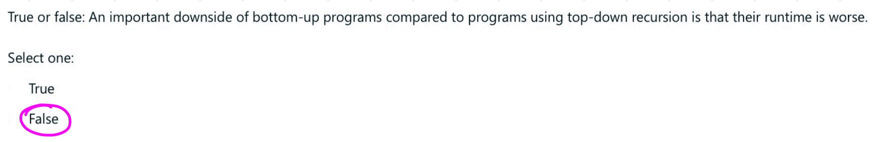
a. The node labeled 1

[✓ b.] The node labeled 2

c. Both

d. Neither

3)



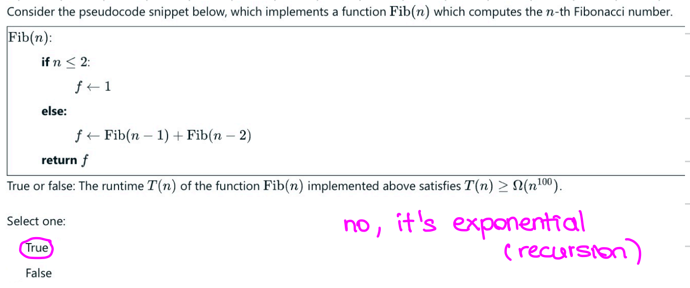
True or false: An important downside of bottom-up programs compared to programs using top-down recursion is that their runtime is worse.

Select one:

True

[✓ False]

4)



Consider the pseudocode snippet below, which implements a function $\mathrm{Fib}(n)$ which computes the n-th Fibonacci number.

```
Fib(n):
    if $n \le 2$:
        $f \leftarrow 1$
    else:
        $f \leftarrow \mathrm{Fib}(n-1) + \mathrm{Fib}(n-2)$
    return $f$
```

True or false: The runtime $T(n)$ of the function $\mathrm{Fib}(n)$ implemented above satisfies $T(n) \ge \Omega(n^{100})$.

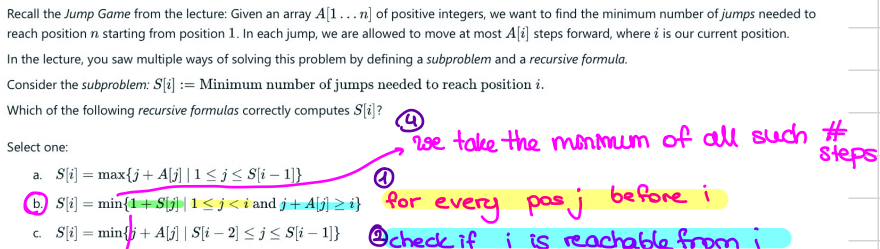
Select one:

[✓ True]

False

no, it's exponential (recursion)

5)



Recall the *Jump Game* from the lecture: Given an array $A[1 \dots n]$ of positive integers, we want to find the minimum number of *jumps* needed to reach position n starting from position 1. In each jump, we are allowed to move at most $A[i]$ steps forward, where i is our current position.

In the lecture, you saw multiple ways of solving this problem by defining a *subproblem* and a *recursive formula*.

Consider the *subproblem*: $S[i] :=$ Minimum number of jumps needed to reach position i.

Which of the following *recursive formulas* correctly computes $S[i]$?

Select one:

a. $S[i] = \max\{j + A[j] \mid 1 \le j \le S[i-1]\}$

[✓ b.] $S[i] = \min\{1 + S[j] \mid 1 \le j < i \text{ and } j + A[j] \ge i\}$

c. $S[i] = \min\{j + A[j] \mid S[i-2] \le j \le S[i-1]\}$

④ we take the minimum of all such # steps

① for every pos j before i

② check if i is reachable from j

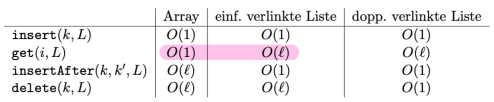
(it's reachable if $j + A[j] \ge i$)

↙ # steps that can be taken at most from pos j

③ if it's reachable we can make one more jump from pos j to pos i and reach i in S[j] + 1 steps

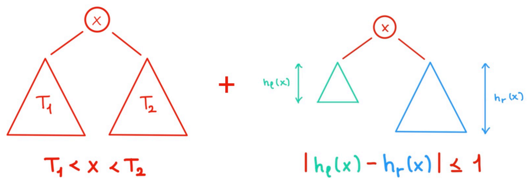
↳ min # steps to reach j

What is used?



	Array	einf. verlinkte Liste	dopp. verlinkte Liste
`insert`(k, L)	$O(1)$	$O(1)$	$O(1)$
`get`(i, L)	$O(1)$	$O(\ell)$	$O(\ell)$
`insertAfter`(k, k', L)	$O(\ell)$	$O(1)$	$O(1)$
`delete`(k, L)	$O(\ell)$	$O(\ell)$	$O(1)$

AVL - Condition

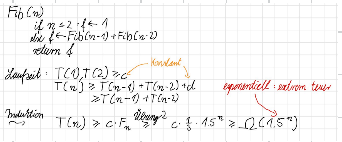


Recursion: exponential runtime

Bottom-up : "dp table calculated with for loop/s"

O(n), O(n²) ← (remember loop counting ...)

Memorizing the subsolutions



```
Fib(n)
    if $n \le 2$ : $f \leftarrow 1$
    else $f \leftarrow \mathrm{Fib}(n-1) + \mathrm{Fib}(n-2)$
    return $f$
```

Laufzeit: $T(1), T(2) \ge c$ Konstant

$T(n) \ge T(n-1) + T(n-2) + d$

$\ge T(n-1) + T(n-2)$

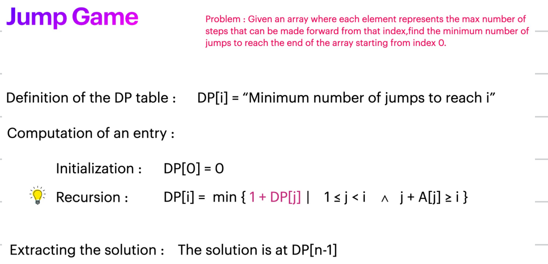
exponentiell: extrem teuer

Induktion ⇝ $T(n) \ge c \cdot F_n \overset{\text{Übung 2}}{\ge} c \cdot \frac{1}{3} \cdot 1.5^n \ge \Omega(1.5^n)$

or $T(n) \ge T(n-1) + T(n-2) + c$

$\ge 2 \cdot T(n-2)$

$\ge 4 \cdot T(n-4) \; \dots \ge \Omega(2^{n/2})$



Jump Game

Problem : Given an array where each element represents the max number of steps that can be made forward from that index,find the minimum number of jumps to reach the end of the array starting from index 0.

Definition of the DP table : DP[i] = “Minimum number of jumps to reach i”

Computation of an entry :

Initialization : DP[0] = 0

Recursion : DP[i] = min { 1 + DP[j] | 1 ≤ j < i ∧ j + A[j] ≥ i }

Extracting the solution : The solution is at DP[n-1]