# THEORETICAL COMPUTER SCIENCE TUTORING (4)

Maurizio Fiusco

TOR VERGATA
UNIVERSITÀ DEGLI STUDI DI ROMA

**Problem 3.1** from the exam held on June 18, 2018

Let $\boldsymbol{L_1} \subseteq \boldsymbol{\Sigma}^*$ be an **acceptable** but undecidable language and let $\boldsymbol{L_2} \subseteq \boldsymbol{\Sigma}^*$ be a decidable language. Consider the following function $\boldsymbol{f}: \sigma^* \to \mathbb{N} : \forall x \in \Sigma^*$

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } x \in L_1 \\ |x| & \text{if } x \notin L_1 \wedge x \in L_2 \\ 0 & \text{otherwise} \end{cases}$$

Show whether $\boldsymbol{f}$ is a computable function

# **Problem 3.1** from the exam held on June 18, 2018

## Claim

$f$ is **not** computable

## Proof

Let's assume for contradiction that $f$ is computable. We could then construct a Turing machine $T_1$ that decides $L_1$

$$x \in \Sigma^* \longrightarrow \boxed{T_f} \longrightarrow f(x)$$

suppose that $T_f$ has three tapes

$x \in \Sigma^*$

working tape

$f(x)$

**Problem 3.1** from the exam held on June 18, 2018

Let's build a recognizer $\boldsymbol{T_1}$

$\boldsymbol{T_1}$

$x \in \Sigma^*$

working tape

$f(\boldsymbol{x})$

Calculate $\boldsymbol{f(x)}$ on the third tape

**if** $\boldsymbol{f(x)} = 1$ ✔ $\boldsymbol{T_1}$ accepts

**else if** $\boldsymbol{f(x)} = |x|$ or $\boldsymbol{f(x)} = 0$ ✘ $\boldsymbol{T_1}$ rejects

$\boldsymbol{T_1}$ **decides** $\boldsymbol{L_1}$ ⚡ contradiction

**Problem 6.1** from Es-LinguaggiEComplessita.pdf (uniroma2.it)

Prove that for every constant $k \in \mathbb{N}$, $\mathbf{2^{n^k}}$ is a **time-constructible** function.

**Definition**

A function $f: \mathbb{N} \to \mathbb{N}$ is **time-constructible** if there exists a Turing machine $T$ of transducer type that, given an input integer $n$ in unary $(1^n)$, writes on the output tape $f(n)$ in unary $\left(1^{f(n)}\right)$ in $dtime(T, n) = O\big(f(n)\big)$

⏳ $O\big(f(n)\big)$

$n$

$f(n)$

**Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]

**Claim**

$\forall$ constant $k \in \mathbb{N}$, $\mathbf{2^{n^k}}$ is a **time-constructible**

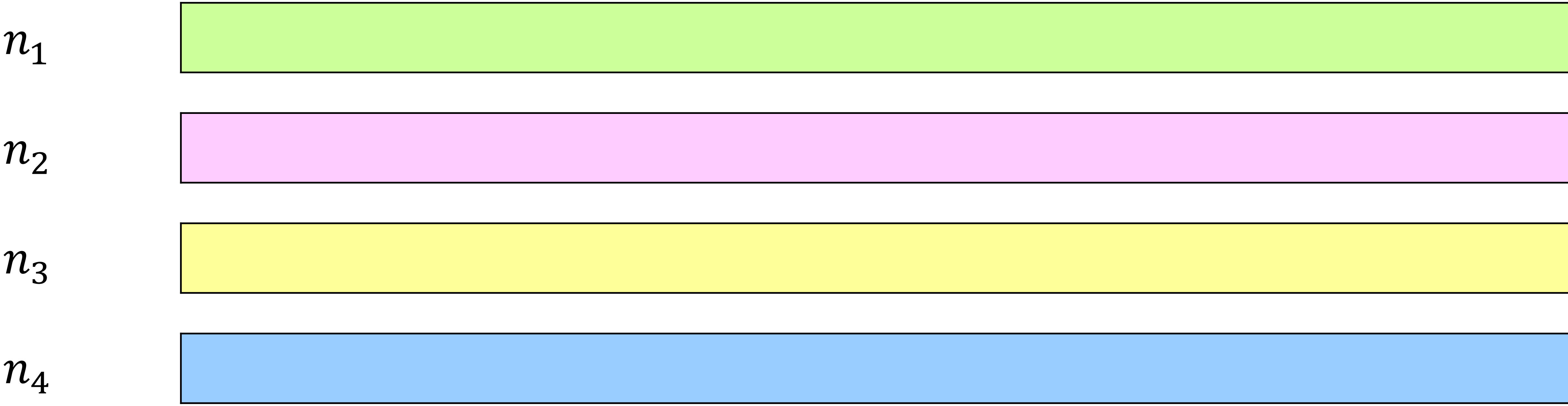
Let’s build a Turing machine that computes $\mathbf{2^{n^k}}$

**Problem 6.1** from Es-LinguaggiEComplessita.pdf (uniroma2.it)

**concatenation**

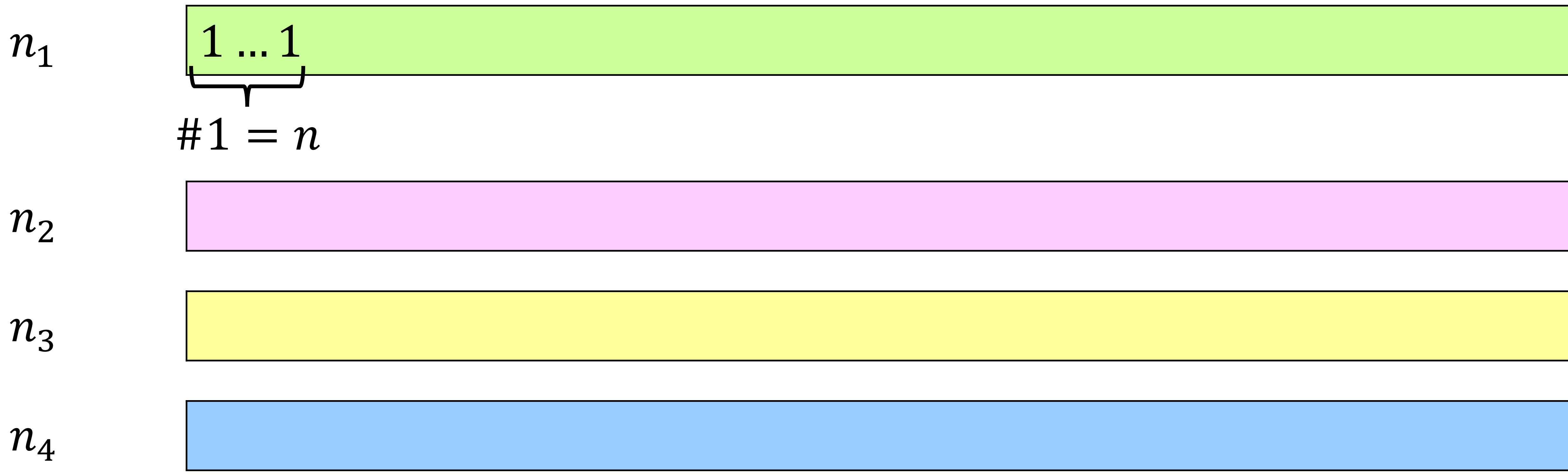
$$\underbrace{1 \ldots 1}_{\#1 = 2^{j}} + \underbrace{1 \ldots 1}_{\#1 = 2^{j}} = \underbrace{1 \ldots 1}_{\#1 = 2^{j+1}}$$

## **Problem 6.1** from Es-LinguaggiEComplessita.pdf (uniroma2.it)




$\boldsymbol{n_1 \leftarrow n}$

**Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]()




$\boldsymbol{n_2 \leftarrow k^{th}}$**-power(**$\boldsymbol{n_1}$**)**

We know how to compute the $k^{th}$ power in $O(n^k)$ steps

**Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]

$n_1$ $\underbrace{1 \dots 1}_{\#1 = n}$

$n_2$ $\underbrace{1 \dots 1}_{\#1 = n^k}$

$n_3$

$n_4$

$\boldsymbol{n_3 \leftarrow 2}$

# **Problem 6.1** from Es-LinguaggiEComplessita.pdf (uniroma2.it)

$n_1$ $\underbrace{1 \dots 1}_{\#1 = n}$

$n_2$ $\underbrace{1 \dots 1}_{\#1 = n^k}$

$n_3$ $\underbrace{11}_{\#1 = 2^1}$

$n_4$

Let $\boldsymbol{i_2}$ be the position of the head on the second tape

$\boldsymbol{i_2 \leftarrow 2}$

# **Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]()

| | |
|---|---|
| $n_1$ | $1 \ldots 1$ |
| $n_2$ | $1\mathbf{1} \ldots 1$ |
| $n_3$ | $11$ |
| $n_4$ | |

**while**$(\boldsymbol{i_2} \leq \boldsymbol{n_2})$ **do**

**Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]()

| | |
|---|---|
| $n_1$ | $1 \dots 1$ |
| $n_2$ | $1\mathbf{1} \dots 1$ |
| $n_3$ | $11$ |
| $n_4$ | |

**while**($\boldsymbol{i_2 \leq n_2}$) **do**

$\quad \boldsymbol{n_4 \leftarrow n_3}$

**Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]

| | |
|---|---|
| $n_1$ | $1 \ldots 1$ |
| $n_2$ | $1\mathbf{1} \ldots 1$ |
| $n_3$ | $11$ |
| $n_4$ | $11$ |

**while**$(\boldsymbol{i_2} \leq \boldsymbol{n_2})$ **do**

$$\boldsymbol{n_4} \leftarrow \boldsymbol{n_3}$$

$$\boldsymbol{n_3} \leftarrow \boldsymbol{n_3} + \boldsymbol{n_4}$$

**Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]

| | |
|---|---|
| $n_1$ | $1 \dots 1$ |
| $n_2$ | $1\mathbf{1} \dots 1$ |
| $n_3$ | $1111$ |
| $n_4$ | $11$ |

**while**$(\boldsymbol{i_2} \leq \boldsymbol{n_2})$ **do**

$$\boldsymbol{n_4} \leftarrow \boldsymbol{n_3}$$
$$\boldsymbol{n_3} \leftarrow \boldsymbol{n_3} + \boldsymbol{n_4}$$
$$\boldsymbol{i_2} \leftarrow \mathbf{i_2} + \mathbf{1}$$

**Problem 6.1** from Es-LinguaggiEComplessita.pdf (uniroma2.it)

| | |
|---|---|
| $n_1$ | 1 … 1 |
| $n_2$ | 11**1** … 1 |
| $n_3$ | 1111 |
| $n_4$ | 11 |

**while**($i_2 \leq n_2$) **do**

$$\boldsymbol{n_4 \leftarrow n_3}$$
$$\boldsymbol{n_3 \leftarrow n_3 + n_4}$$
$$\boldsymbol{i_2 \leftarrow \mathbf{i}_2 + 1}$$

**Problem 6.1** from Es-LinguaggiEComplessita.pdf (uniroma2.it)

$n_1$ $1 \ldots 1$

$n_2$ $11\mathbf{1} \ldots 1$

$n_3$ $\underbrace{1111}_{\#1 = 2^{i_2}}$

$n_4$ $11$

**After each iteration**

# **Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]()

| | |
|---|---|
| $n_1$ | $1 \dots 1$ |
| $n_2$ | $111 \dots \mathbf{1}$ |
| $n_3$ | $\underbrace{1 \dots 1}_{\#1 \,=\, 2^{n_2}}$ |
| $n_4$ | $11$ |

$n_2 = n^k$

**output $\boldsymbol{n_3}$**

**Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]

$n_1 \longleftarrow n$

$n_2 \leftarrow k^{th}$-power$(n_1)$ $\quad O(n^k)$

$n_3 \longleftarrow 2$

$i_2 \leftarrow 2$

while$(i_2 \leq n_2)$ do $\quad 2 \leq i_2 \leq n^k$

  $n_4 \leftarrow n_3$ $\quad \#1 = 2^{i_2-1}$

  $n_3 \leftarrow n_3 + n_4$ $\quad \#1 = 2^{i_2}$

  $i_2 \leftarrow \mathrm{i}_2 + 1$

output $n_3$

$$\sum_{i_2=2}^{n^k} \left[2^{i_2-1} + 2^{i_2}\right]$$

**Problem 6.1** from Es-LinguaggiEComplessita.pdf (uniroma2.it)

$$\sum_{i_2=2}^{n^k}\left[\mathbf{2}^{i_2-1}+\mathbf{2}^{i_2}\right]=\sum_{i_2=2}^{n^k}\mathbf{2}^{i_2-1}+\sum_{i_2=2}^{n^k}\mathbf{2}^{i_2}=$$

$$=\frac{\mathbf{1}}{\mathbf{2}}\sum_{i_2=2}^{n^k}\mathbf{2}^{i_2}+\sum_{i_2=2}^{n^k}\mathbf{2}^{i_2}=\left(\frac{\mathbf{1}}{\mathbf{2}}+\mathbf{1}\right)\sum_{i_2=2}^{n^k}\mathbf{2}^{i_2}=$$

$$=\left(\frac{\mathbf{3}}{\mathbf{2}}\right)\sum_{i_2=2}^{n^k}\mathbf{2}^{i_2}=\left(\frac{3}{2}\right)\left(2^{n^k+1}-1-1-2\right)<$$

$$\sum_{i=0}^{n}\mathbf{2}^{i}=\mathbf{2}^{n+1}-\mathbf{1}$$

$$<3\cdot 2^{n^k}\in O\left(2^{n^k}\right)$$

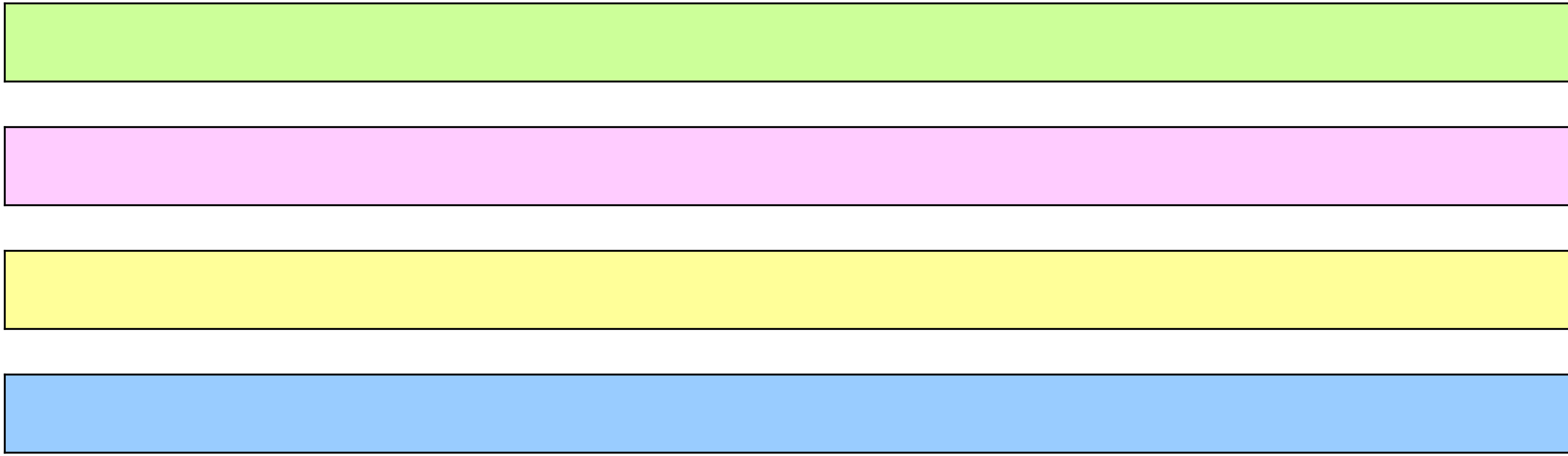
**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

Prove that $f(n) = n^n$ is a **time-constructible** function

**Claim**

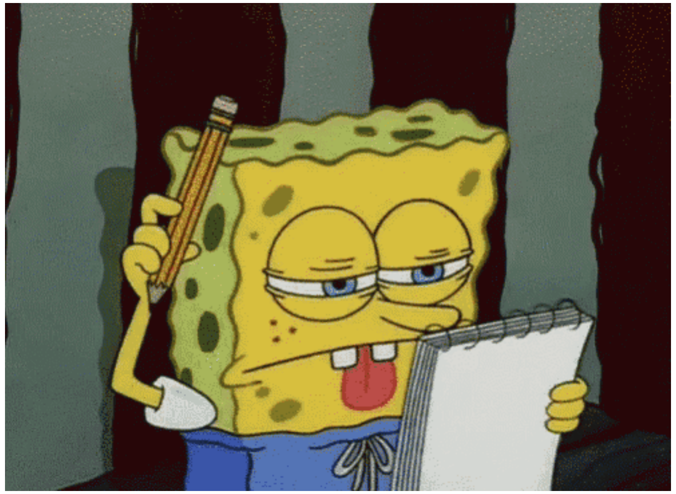
$f(n) = n^n$ is a **time-constructible** function

Let's build a Turing machine that computes $n^n$



# **Problem 6.1** from [Es-LinguaggiEComplessita.pdf (uniroma2.it)]

$$\underbrace{\underbrace{1 \dots 1}_{\#1 = n^j} \overset{\textbf{concatenation}}{+} \dots + \underbrace{1 \dots 1}_{\#1 = n^j}}_{n \text{ blocks}} = \underbrace{1 \dots 1}_{\#1 = n^{j+1}}$$



💡 2 nested cycles

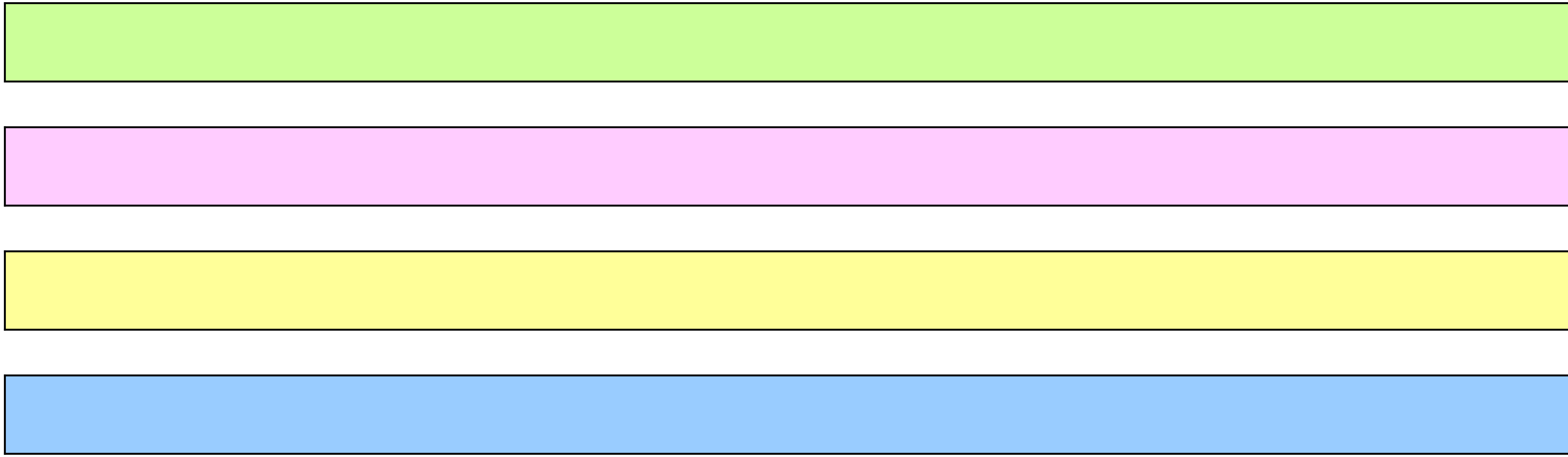
**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

$n_1$

$n_2$

$n_3$

$n_4$



$$\boldsymbol{n_1 \leftarrow n}$$

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

$n_1$ $\underbrace{1 \ldots 1}_{\#1 = n}$

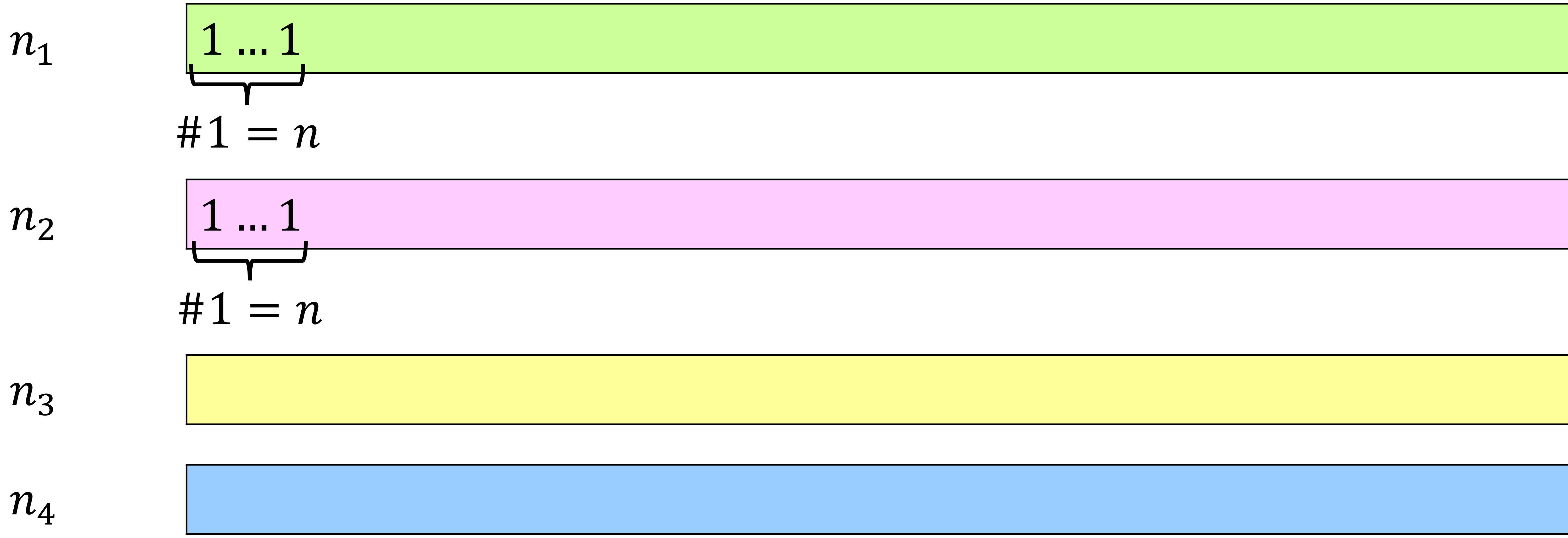
$n_2$

$n_3$

$n_4$

$\boldsymbol{n_2} \leftarrow \boldsymbol{n_1}$

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)




Let $\boldsymbol{i}$ and $\boldsymbol{j}$ be the positions of the heads of the first two tapes

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

$n_1$ $\underbrace{1 \dots 1}_{\#1 = n}$

$n_2$ $\underbrace{1 \dots 1}_{\#1 = n}$

$n_3$

$n_4$

$\boldsymbol{n_3 \leftarrow n_1}$

**Exercise 1** from Antonio Cruciani’s tutoring (lesson 4)

$n_1$ $\underbrace{1 \ldots 1}_{\#1 = n}$

$n_2$ $\underbrace{1 \ldots 1}_{\#1 = n}$

$n_3$ $\underbrace{1 \ldots 1}_{\#1 = n}$

$n_4$

$\boldsymbol{i \leftarrow 2}$

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

| | |
|---|---|
| $n_1$ | $1\mathbf{1}\ldots1$ |
| $n_2$ | $1\ldots1$ |
| $n_3$ | $1\ldots1$ |
| $n_4$ | |

**while**$(\boldsymbol{i} \leq \boldsymbol{n})$ **do**

$\boldsymbol{n_4} \leftarrow \boldsymbol{n_3}$

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

| | |
|---|---|
| $n_1$ | $1\mathbf{1} \ldots 1$ |
| $n_2$ | $1 \ldots 1$ |
| $n_3$ | $1 \ldots 1$ |
| $n_4$ | $1 \ldots 1$ |

**while**$(\boldsymbol{i \leq n})$ **do**

$\boldsymbol{n_4 \leftarrow n_3}$

$\boldsymbol{j \leftarrow 2}$

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

| | |
|---|---|
| $n_1$ | $1\mathbf{1}\dots1$ |
| $n_2$ | $1\mathbf{1}\dots1$ |
| $n_3$ | $1\dots1$ |
| $n_4$ | $1\dots1$ |

**while($i \leq n$) do**

$\quad \boldsymbol{n_4 \leftarrow n_3}$

$\quad \boldsymbol{j \leftarrow 2}$

$\quad$ **while(j $\leq n$) do**

$\quad\quad \boldsymbol{n_3 \leftarrow n_3 + n_4}$

$\quad\quad \boldsymbol{j \leftarrow j + 1}$

# **Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

| | |
|---|---|
| $n_1$ | $1\mathbf{1}\ldots1$ |
| $n_2$ | $11\mathbf{1}\ldots1$ |
| $n_3$ | $1\ldots11\ldots1$ |
| $n_4$ | $1\ldots1$ |

**while**($\boldsymbol{i \le n}$) **do**

  $\boldsymbol{n_4 \leftarrow n_3}$

  $\boldsymbol{j \leftarrow 2}$

  **while**(**j** $\boldsymbol{\le n}$) **do**

    $\boldsymbol{n_3 \leftarrow n_3 + n_4}$

    $\boldsymbol{j \leftarrow j + 1}$

## **Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

$n_1$ — $1\mathbf{1} \ldots 1$

$n_2$ — $1 \ldots \mathbf{1}$

$n_3$ — $1 \ldots 1 \ldots 1 \ldots 1$

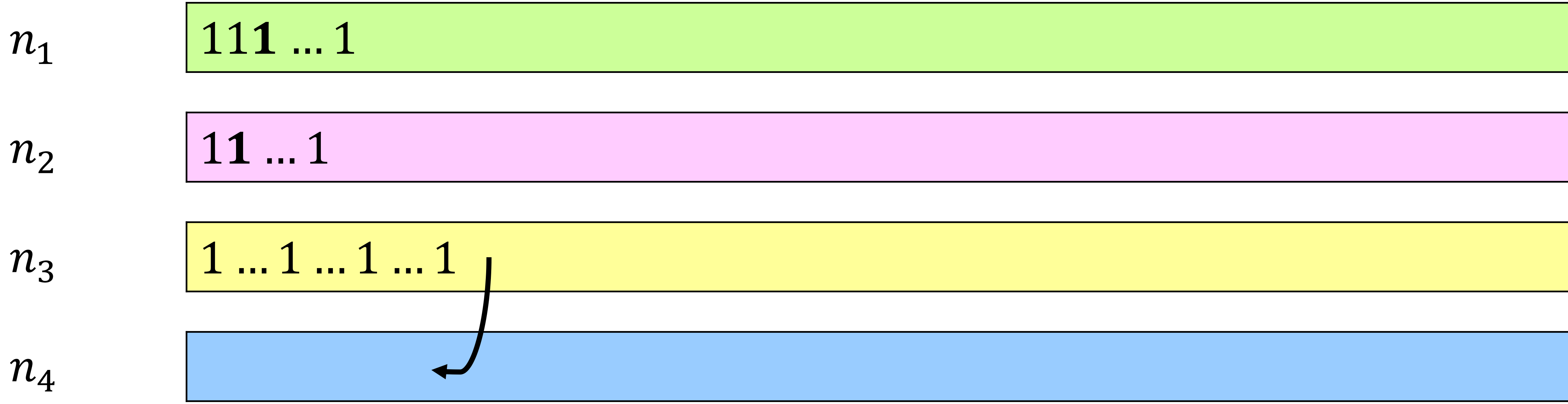
$n_4$ — $1 \ldots 1$

**while**$(i \leq n)$ **do**

- $n_4 \leftarrow n_3$
- $j \leftarrow 2$
- **while**$(j \leq n)$ **do**
  - $n_3 \leftarrow n_3 + n_4$
  - $j \leftarrow j + 1$
- $i \leftarrow i + 1$

$\#1 = n_4 \cdot n$

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)




**while**$(i \leq n)$ **do**

$\quad \boldsymbol{n_4 \leftarrow n_3}$

$\quad \boldsymbol{j \leftarrow 2}$

$\quad$ **while**$(j \leq n)$ **do**

$\qquad \boldsymbol{n_3 \leftarrow n_3 + n_4}$

$\qquad \boldsymbol{j \leftarrow j + 1}$

$\quad \boldsymbol{i \leftarrow i + 1}$

…

## **Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

$n_1$ $111 \ldots \mathbf{1}$

$n_2$ $11 \ldots \mathbf{1}$

$n_3$ $1 \ldots 1 \ldots 1 \ldots 1$

$n_4$

```
while(i ≤ n) do
    n4 ← n3
    j ← 2
    while(j ≤ n) do
        n3 ← n3 + n4
        j ← j + 1
    i ← i + 1
output n3
```

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

$\boldsymbol{n_1, n_2, n_3 \leftarrow n}$ — $\boldsymbol{O(n)}$

$\boldsymbol{i \leftarrow 2}$

**while(**$\boldsymbol{i \leq n}$**) do** — $\boldsymbol{n}$ **times**

- $\boldsymbol{n_4 \leftarrow n_3}$ — $\boldsymbol{\#1 \ = n^{i-1}}$
- $\boldsymbol{j \leftarrow 2}$
- **while(j** $\boldsymbol{\leq n}$**) do** — $\boldsymbol{n}$ **times**
  - $\boldsymbol{n_3 \leftarrow n_3 + n_4}$ — **#1 added** $\boldsymbol{= n^{i-1}}$
  - $\boldsymbol{j \leftarrow j + 1}$
- $\boldsymbol{i \leftarrow i + 1}$

**output** $\boldsymbol{n_3}$

Inner loop: $\boldsymbol{O(n^i)}$

Outer loop: $\sum_{i=2}^{n} n^i$

**Exercise 1** from Antonio Cruciani's tutoring (lesson 4)

$$\sum_{i=2}^{n} n^i = \frac{n^2 - n^{n+1}}{1-n} = \frac{n^{n+1} - n^2}{n-1} \in O(n^n)$$

$$\sum_{k=m}^{n} x^k = \frac{x^m - x^{n+1}}{1-x}$$