Algoritmy

Pseudokód

Algoritmy většinou nebudeme zapisovat jako programy pro stroj RAM, ale spíše jako programy v nějakém vyšším programovacím jazyce.

Nebudeme se vázat na nějaký konkrétní programovací jazyk.

Programy budeme zapisovat pomocí **pseudokódu**, jehož syntaxi si budeme libovolně přizpůsobovat podle potřeby (např. použití libovolné matematické notace, slovních popisů, apod.).

Příklad:

Algoritmus: Algoritmus pro nalezení největšího prvku v poli

```
FIND-MAX (A, n):
    k := 0
    for i := 1 to n − 1 do
        if A[i] > A[k] then
            k := i
    return A[k]
```

Algoritmy

Poznámka:

Z hlediska analýzy toho, jak daný algoritmus funguje, většinou není příliš podstatný rozdíl v tom, jestli algoritmus:

- čte vstupní data z nějakého vstupního zařízení (např. ze souboru na disku, z klávesnice, apod.)
- zapisuje data na nějaké výstupní zařízení (např. do souboru, na obrazovku, apod.)

nebo

- čte vstupní data z paměti (např. jsou mu předány jako parametry)
- zapisuje data na do paměti (např. je vrátí jako návratovou hodnotu)

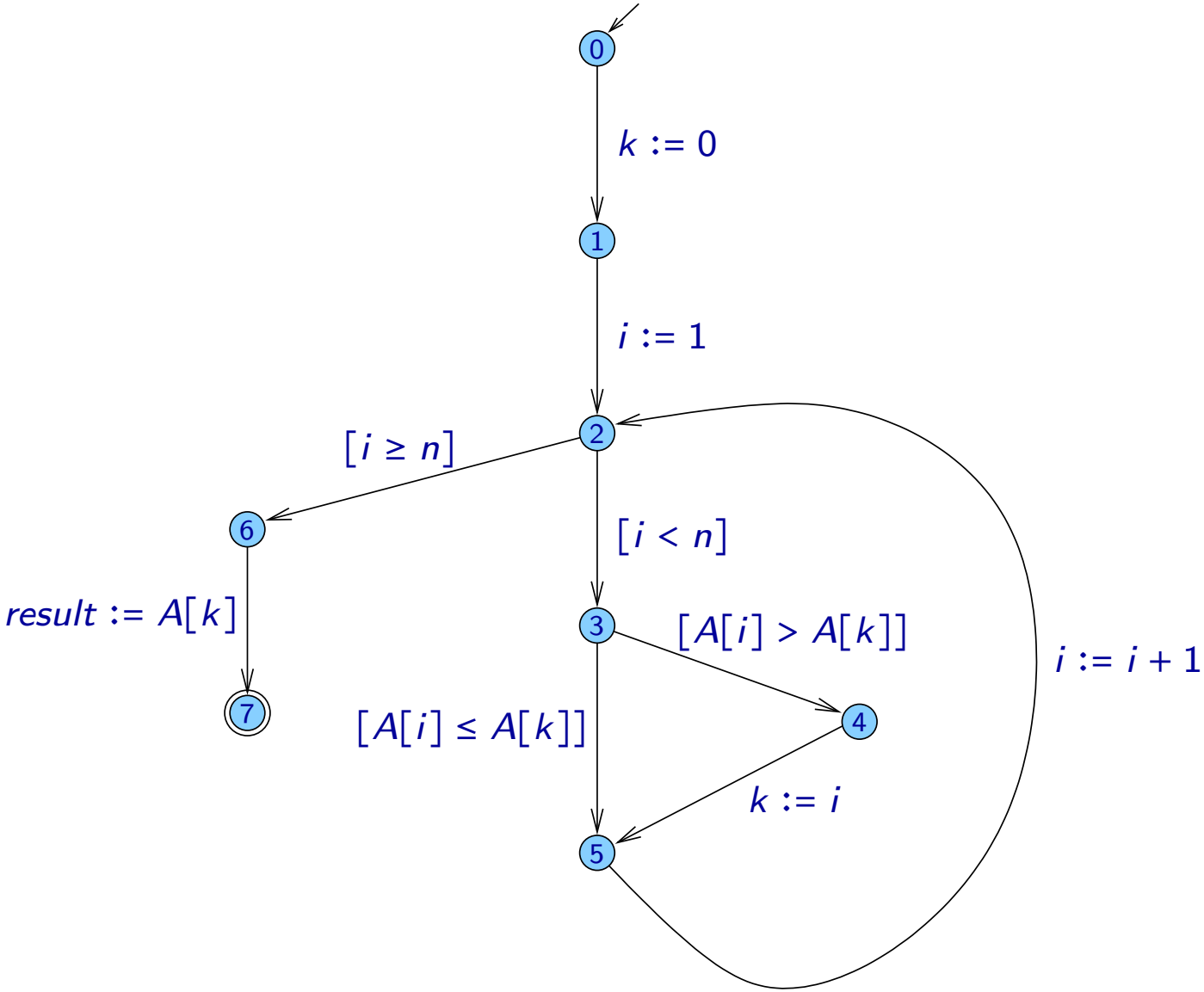
V pseudokódu tak tedy většinou budou vstupní data předávána jako argumenty dané funkce a výstup bude představován návratovou hodnotou této funkce.

Řídící tok

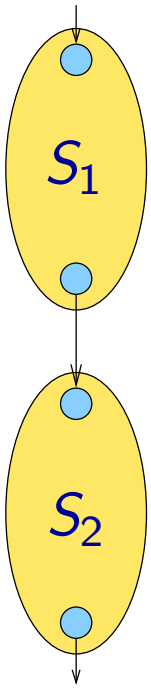
Instrukce lze zhruba rozdělit na dvě skupiny:

- instrukce přímo pracující s daty:
  - přiřazení
  - vyhodnocení hodnot výrazů v podmínkách
  - čtení vstupu, zápis na výstup
  - ...
- instrukce ovlivňující **řídící tok** — určují, které instrukce se budou provádět, v jakém pořadí, apod.:
  - větvení (if, switch, ...)
  - cykly (while, do .. while, for, ...)
  - uspořádání instrukcí do bloků
  - návraty z podpogramů (return, ...)
  - ...

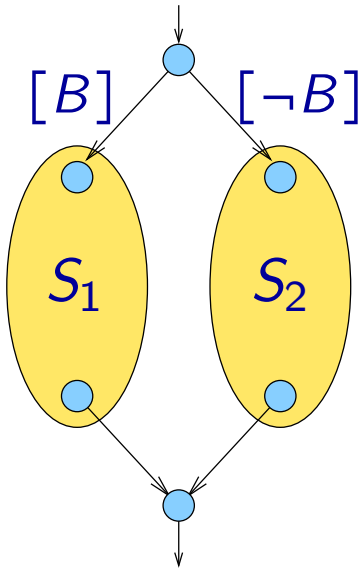
Graf řídícího toku



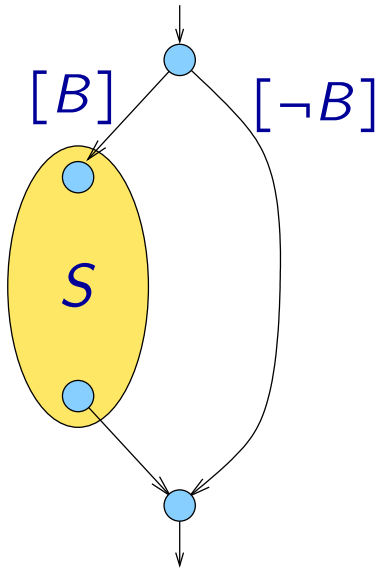
Některé základní konstrukce strukturovaného programování



$S_1; S_2$

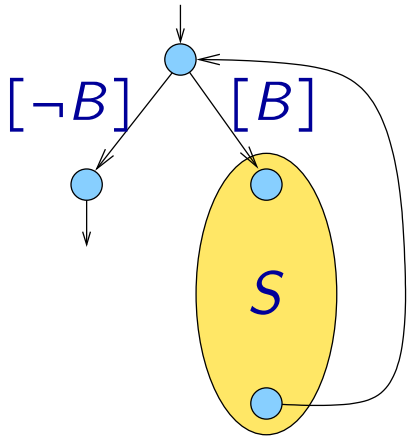


if B **then** S_1 **else** S_2

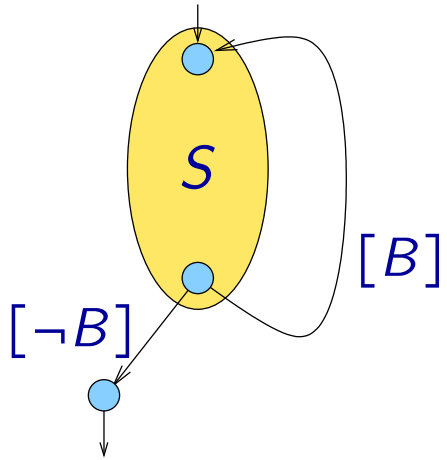


if B **then** S

Některé základní konstrukce strukturovaného programování

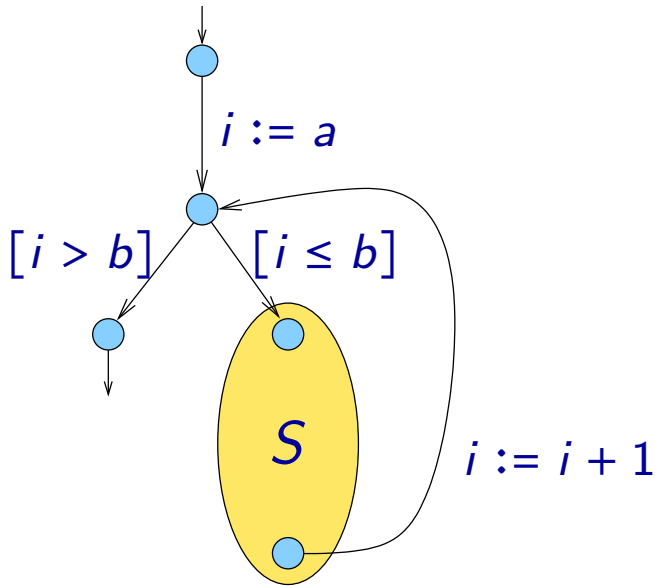


while B **do** S



do S **while** B

Některé základní konstrukce strukturovaného programování



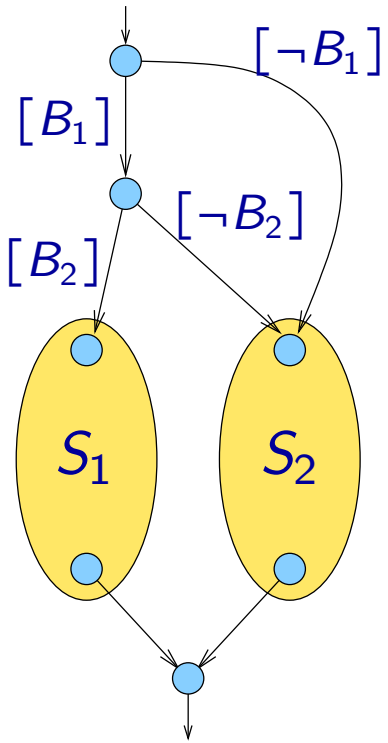
$i := a$
while $i \leq b$ **do**
  S
  $i := i + 1$

for $i := a$ **to** b **do** S

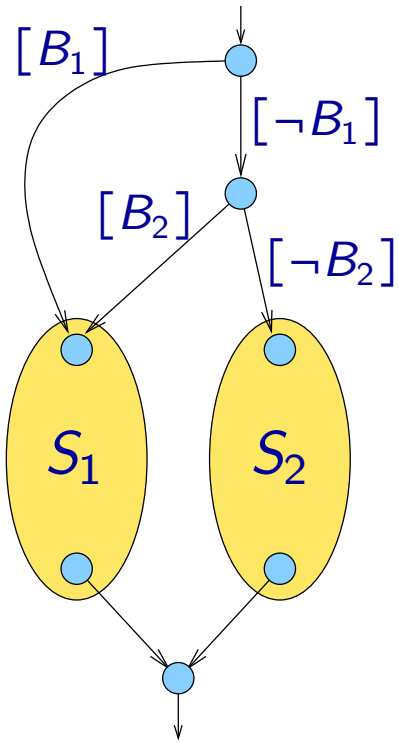
Některé základní konstrukce strukturovaného programování

Zkrácené vyhodnocování složených podmínek, např.:

while $i < n$ **and** $A[i] > x$ **do** ...



if B_1 **and** B_2 **then** S_1 **else** S_2



if B_1 **or** B_2 **then** S_1 **else** S_2

Řídící tok realizovaný pomocí goto

- **goto** ℓ — **nepodmíněný skok**
- **if** B **then goto** ℓ — **podmíněný skok**

Příklad:

0: $k := 0$
1: $i := 1$
2: **goto** *6*
3: **if** $A[i] \leq A[k]$ **then goto** *5*
4: $k := i$
5: $i := i + 1$
6: **if** $i < n$ **then goto** *3*
7: **return** $A[k]$

Řídící tok realizovaný pomocí goto

- **goto** ℓ — **nepodmíněný skok**
- **if** B **then goto** ℓ — **podmíněný skok**

Příklad:

$$
\begin{array}{rl}
\mathit{start}: & k := 0 \\
& i := 1 \\
& \textbf{goto } \mathit{L3} \\
\mathit{L1}: & \textbf{if } A[i] \leq A[k] \textbf{ then goto } \mathit{L2} \\
& k := i \\
\mathit{L2}: & i := i + 1 \\
\mathit{L3}: & \textbf{if } i < n \textbf{ then goto } \mathit{L1} \\
& \textbf{return } A[k]
\end{array}
$$

Vyhodnocení složitých výrazů

Vyhodnocení složitého výrazu, jako třeba

$$A[i + s] := (B[3 * j + 1] + x) * y + 8$$

může být na nižší úrovni nahrazeno posloupností jednodušších příkazů, jako třeba

$$\begin{aligned} t_1 &:= i + s \\ t_2 &:= 3 * j \\ t_2 &:= t_2 + 1 \\ t_3 &:= B[t_2] \\ t_3 &:= t_3 + x \\ t_3 &:= t_3 * y \\ t_3 &:= t_3 + 8 \\ A[t_1] &:= t_3 \end{aligned}$$

Výpočet algoritmu

Konfigurace — popis celkového stavu stroje v nějakém okamžiku během výpočtu

Příklad: Konfigurace tvaru

$$(q, mem)$$

kde

- q — aktuální řídící stav
- mem — představuje aktuální obsah paměti stroje — jaké hodnoty jsou momentálně přiřazeny jednotlivým proměnným.

Příklad obsahu paměti mem:

$$\langle A\colon [3,8,1,3,6],\quad n\colon 5,\quad i\colon 1,\quad k\colon 0,\quad result\colon ?\rangle$$

Výpočet algoritmu

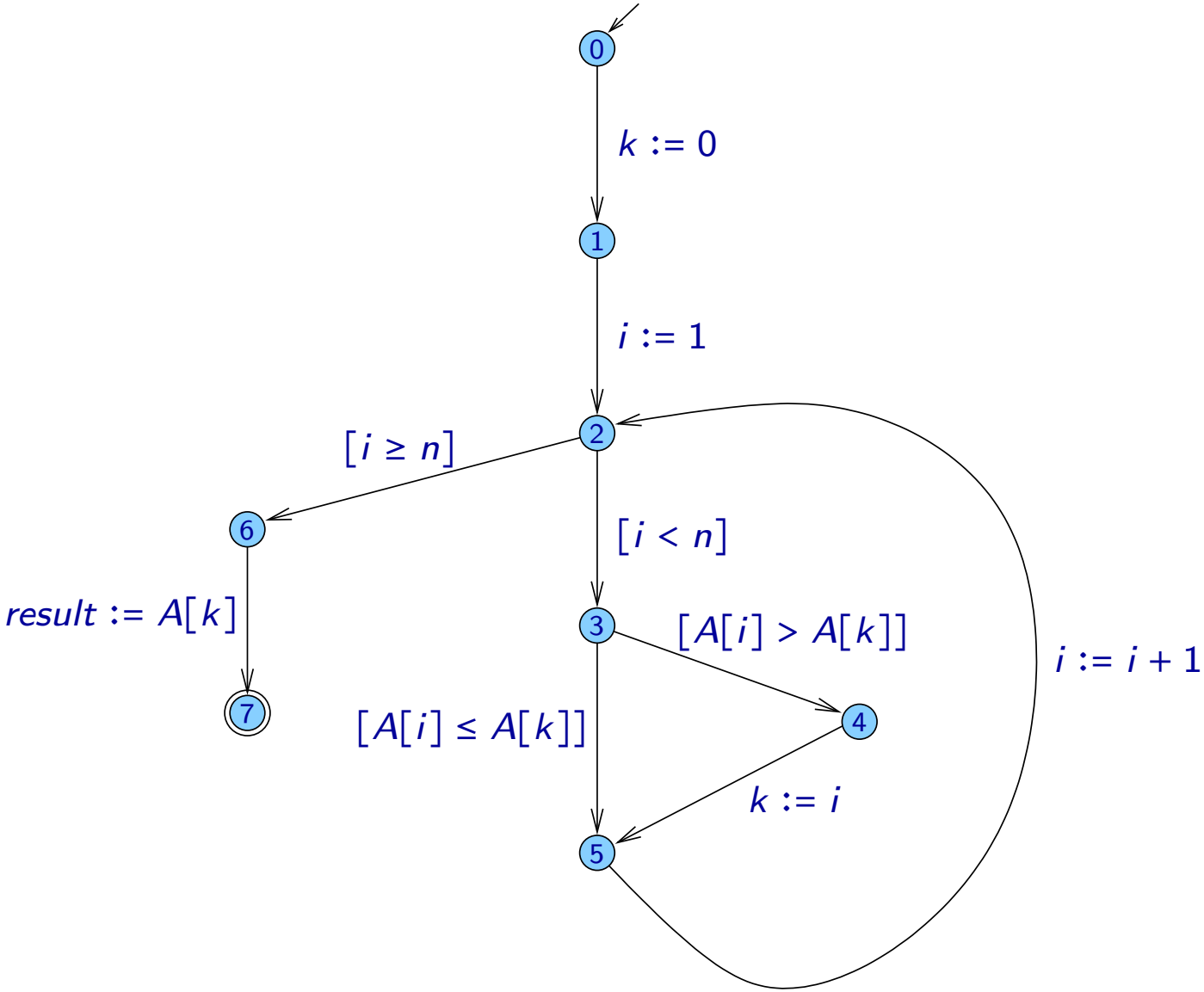
Příklad konfigurace:

$$(2,\ \langle A\colon [3,8,1,3,6],\quad n\colon 5,\quad i\colon 1,\quad k\colon 0,\quad \mathit{result}\colon ?\rangle)$$

Výpočet stroje $\mathcal{M}$ provádějícího algoritmus Alg, kde zpracovává vstup w, je posloupnost konfigurací.

- Začíná se v **počáteční konfiguraci**.
- Každým krokem stroj přechází z jedné konfigurace do další.
- Výpočet končí v **koncové konfiguraci**.

Výpočet algoritmu



Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3, 8, 1, 3, 6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad \mathit{result}\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad \mathit{result}\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad \mathit{result}\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6], \quad n\colon 5, \quad i\colon ?, \quad k\colon ?, \quad result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6], \quad n\colon 5, \quad i\colon ?, \quad k\colon 0, \quad result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6], \quad n\colon 5, \quad i\colon 1, \quad k\colon 0, \quad result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6], \quad n\colon 5, \quad i\colon 1, \quad k\colon 0, \quad result\colon ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon ?,\ result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon 0,\ result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 1,\ result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon ?,\ result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon 0,\ result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 1,\ result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_{11}: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } ?, \quad result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } ?, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 0, \quad result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 1, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 2, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_{11}: $(5, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 3, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$
α_{12}: $(2, \langle A\text{: } [3,8,1,3,6], \quad n\text{: } 5, \quad i\text{: } 4, \quad k\text{: } 1, \quad result\text{: } ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon ?,\ result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon 0,\ result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 1,\ result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_9: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{10}: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{11}: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{12}: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 4,\ k\colon 1,\ result\colon ?\rangle)$
α_{13}: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 4,\ k\colon 1,\ result\colon ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon ?, \; k\colon ?, \; result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon ?, \; k\colon 0, \; result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 1, \; result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_9: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{10}: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{11}: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{12}: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{13}: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{14}: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon ?,\ result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon 0,\ result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 1,\ result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_9: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{10}: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{11}: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{12}: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 4,\ k\colon 1,\ result\colon ?\rangle)$
α_{13}: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 4,\ k\colon 1,\ result\colon ?\rangle)$
α_{14}: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 4,\ k\colon 1,\ result\colon ?\rangle)$
α_{15}: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 5,\ k\colon 1,\ result\colon ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon ?, \; k\colon ?, \; result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon ?, \; k\colon 0, \; result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 1, \; result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_9: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{10}: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{11}: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{12}: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{13}: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{14}: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{15}: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 5, \; k\colon 1, \; result\colon ?\rangle)$
α_{16}: $(6, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 5, \; k\colon 1, \; result\colon ?\rangle)$

Výpočet algoritmu

Příklad: Výpočet, kde algoritmus FIND-MAX zpracovává vstup, kde $A = [3, 8, 1, 3, 6]$ a $n = 5$.

α_0: $(0, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon ?, \; k\colon ?, \; result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon ?, \; k\colon 0, \; result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 0, \; result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 1, \; k\colon 1, \; result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 2, \; k\colon 1, \; result\colon ?\rangle)$
α_9: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{10}: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{11}: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 3, \; k\colon 1, \; result\colon ?\rangle)$
α_{12}: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{13}: $(3, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{14}: $(5, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 4, \; k\colon 1, \; result\colon ?\rangle)$
α_{15}: $(2, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 5, \; k\colon 1, \; result\colon ?\rangle)$
α_{16}: $(6, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 5, \; k\colon 1, \; result\colon ?\rangle)$
α_{17}: $(7, \langle A\colon [3,8,1,3,6], \; n\colon 5, \; i\colon 5, \; k\colon 1, \; result\colon 8\rangle)$

Výpočet algoritmu

Provedením instrukce I se přejde z konfigurace α do konfigurace α':

$$\alpha \xrightarrow{I} \alpha'$$

Výpočet může být:

- **Konečný**:

$$\alpha_0 \xrightarrow{I_0} \alpha_1 \xrightarrow{I_1} \alpha_2 \xrightarrow{I_2} \alpha_3 \xrightarrow{I_3} \alpha_4 \xrightarrow{I_4} \cdots \xrightarrow{I_{t-2}} \alpha_{t-1} \xrightarrow{I_{t-1}} \alpha_t$$

  kde α_t je buď koncová konfigurace nebo konfigurace, kde došlo k chybě a není možné pokračovat

- **Nekonečný**:

$$\alpha_0 \xrightarrow{I_0} \alpha_1 \xrightarrow{I_1} \alpha_2 \xrightarrow{I_2} \alpha_3 \xrightarrow{I_3} \alpha_4 \xrightarrow{I_4} \cdots$$

Výpočet algoritmu

Výpočet je možné popsat dvěma různými způsoby:

- jako posloupnost konfigurací $\alpha_0, \alpha_1, \alpha_2, \ldots$
- jako posloupnost provedených instrukcí $I_0, I_1, I_2, \ldots$

Churchova-Turingova teze

Z přechozího by mělo být jasné, že:

- Program v libovolném programovacím jazyce je možné přeložit do podoby programu pro stroj RAM.
- Činnost stroje RAM je možné simulovat Turingovým strojem.

Činost každého programu v nějakém libovolném programovacím jazyce je tedy možné vykonávat Turingovým strojem.

Churchova-Turingova teze

Každý algoritmus je možné realizovat nějakým Turingovým strojem.

Není to věta, kterou by bylo možno dokázat v matematickém smyslu – není formálně definováno, co je to algoritmus.

Tezi formulovali nezávisle na sobě v polovině 30. let 20. století Alan Turing a Alonzo Church.

Churchova-Turingova teze

Příklady matematických formalismů zachycujících pojem algoritmus:

- Turingovy stroje
- stroje RAM
- lambda kalkulus
- rekurzivní funkce
- ...

Dále můžeme uvést:

- Libovolný (obecný) programovací jazyk (jako např. C, Java, Python, Lisp, Haskell, Prolog apod.).

Všechny tyto modely jsou ekvivalentní z hlediska algoritmů, které jsou schopny realizovat.

Dokazování korektnosti algoritmů

Korektnost algoritmu

Algoritmy slouží k řešení **problémů**.

- **Problém** — specifikace toho, **co** má algoritmus dělat:
  - Popis vstupu
  - Popis výstupu
  - Vztah mezi vstupy a výstupy
- **Algoritmus** — konkrétní postup, **jak** při výpočtu postupovat

Algoritmus je korektním řešením daného problému, jestliže se pro všechny vstupy zastaví a vydá správný výsledek.

Příklad:

Problém: Problém třídění
Algoritmus: Quicksort

Korektnost algoritmu

Příklad:

Problém nalezení maximálního prvku v poli:

Vstup: Pole A indexované od nuly a číslo n udávající počet prvků v tomto poli, přičemž se předpokládá, že $n \geq 1$.

Výstup: Hodnota *result*, která je hodnotou maximálního prvku v poli A, tj. hodnota *result*, pro kterou platí:

- $A[j] \leq result$ pro všechna $j \in \mathbb{N}$, kde $0 \leq j < n$, a
- existuje $j \in \mathbb{N}$ takové, že $0 \leq j < n$ a $A[j] = result$.

Instance problému — konkrétní vstup, např.

$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$

Pro tuto instanci je výstupem hodnota 11.

Korektnost algoritmu

Algoritmus: Algoritmus pro nalezení největšího prvku v poli

```
FIND-MAX (A, n):
    k := 0
    for i := 1 to n − 1 do
        if A[i] > A[k] then
            k := i
    return A[k]
```

Korektnost algoritmu

Definice

Algoritmus *Alg* **řeší** problém *P*, jestliže pro **každou** instanci *w* problému *P* jsou splněny následující dvě podmínky:

(a) Výpočet algoritmu *Alg* nad vstupem *w* se po konečném počtu kroků (korektně) zastaví.

(b) Algoritmus *Alg* vygeneruje pro vstup *w* výstup, který odpovídá podmínkám kladeným na výstup ve specifikaci problému *P*.

Algoritmus, který řeší problém *P*, je korektním řešením tohoto problému.

Korektnost algoritmu

Algoritmus *Alg* **není** korektním řešení problému *P*, jestliže existuje vstup *w* takový, že při výpočtu nad tímto vstupem nastane některá z následujících chyb:

- provedení nějaké chybné nepovolené operace (přístup k prvku pole mimo povolený rozsah indexů, dělení nulou, . . .),
- vygenerovaný výstup neodpovídá podmínkám specifikovaným v zadání problému *P*,
- výpočet se nikdy nezastaví.

Testování — spustění algoritmu nad různými vstupy a zkontrolování, zda se algoritmus pro tyto vstupy chová „správně“.

Testování může prokázat přítomnost chyb, ale ne to, že se algoritmus chová korektně pro **všechny** vstupy.

Korektnost algoritmu

Typicky je množina možných instancí daného problému nekonečná (nebo přinejmenším velmi velká), takže není možné otestovat činnost algoritmu na všech instancích.

Pro zdůvodnění a ověření toho, že algoritmus je korektním řešením daného problému je třeba podat **důkaz**, který bere v úvahu všechny možné výpočty na všech možných vstupech.

Důkaz korektnosti algoritmu je obecně vhodné rozdělit na dvě části:

- Zdůvodnění toho, že algoritmus pro žádný vstup nikdy neudělá nic „špatně“:
  - během výpočtu nedojde k žádné chybné operaci
  - pokud program skončí, výstup bude „správně“
- Zdůvodnění toho, že se algoritmus pro každý vstup po konečném počtu kroků zastaví.

Invarianty

Uvažujme libovolný systém skládající se z:

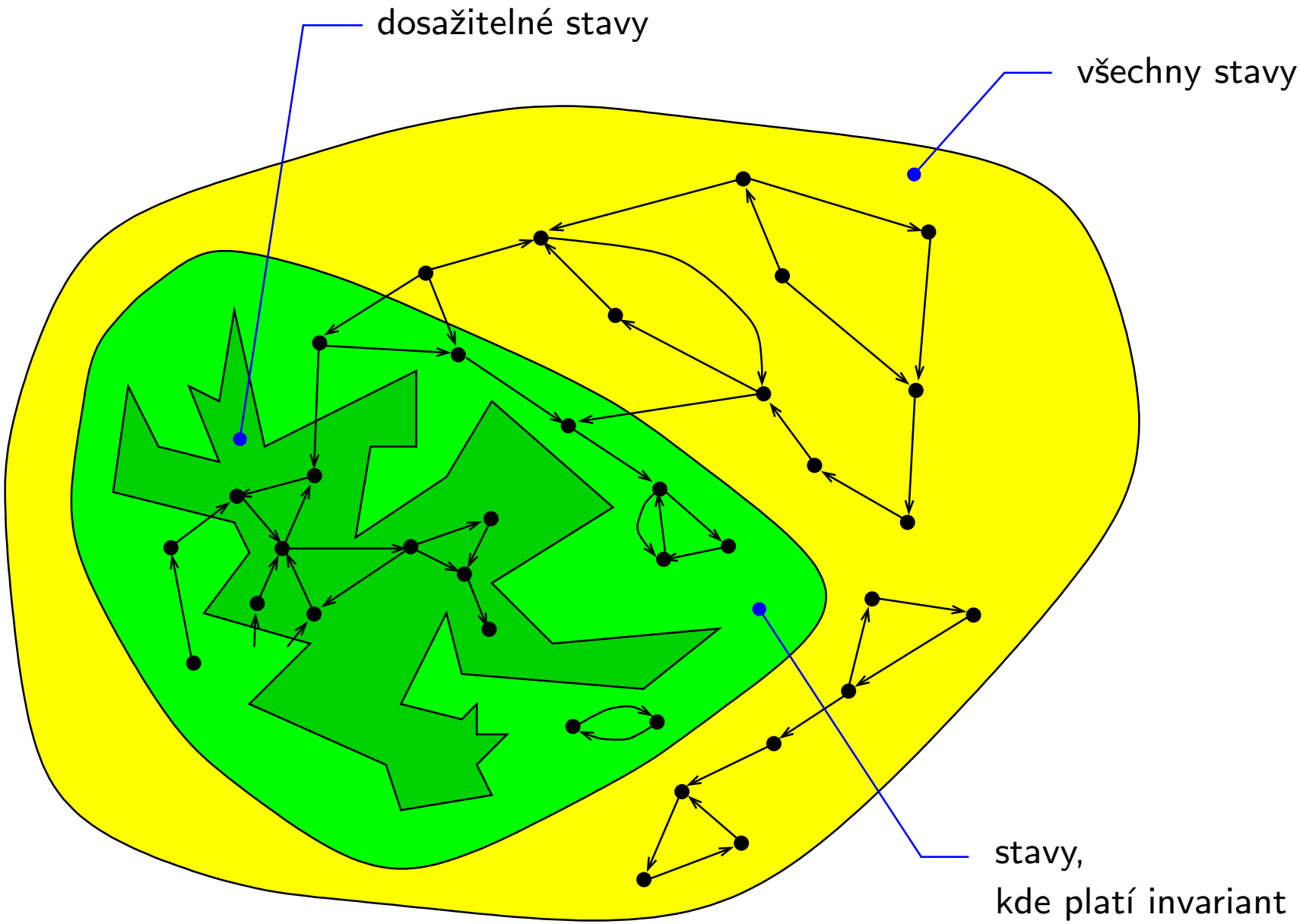
- množiny stavů (či konfigurací) — může být nekonečná
- přechodů mezi těmito stavy
- některé ze stavů jsou určené jako počáteční

Stav je **dosažitelný**, jestliže je možné se do něj dostat z některého počátečního stavu použitím nějaké posloupnosti přechodů.

Invariant je nějaká podmínka vymezující nějakou podmnožinu stavů taková, že platí ve všech dosažitelných stavech:

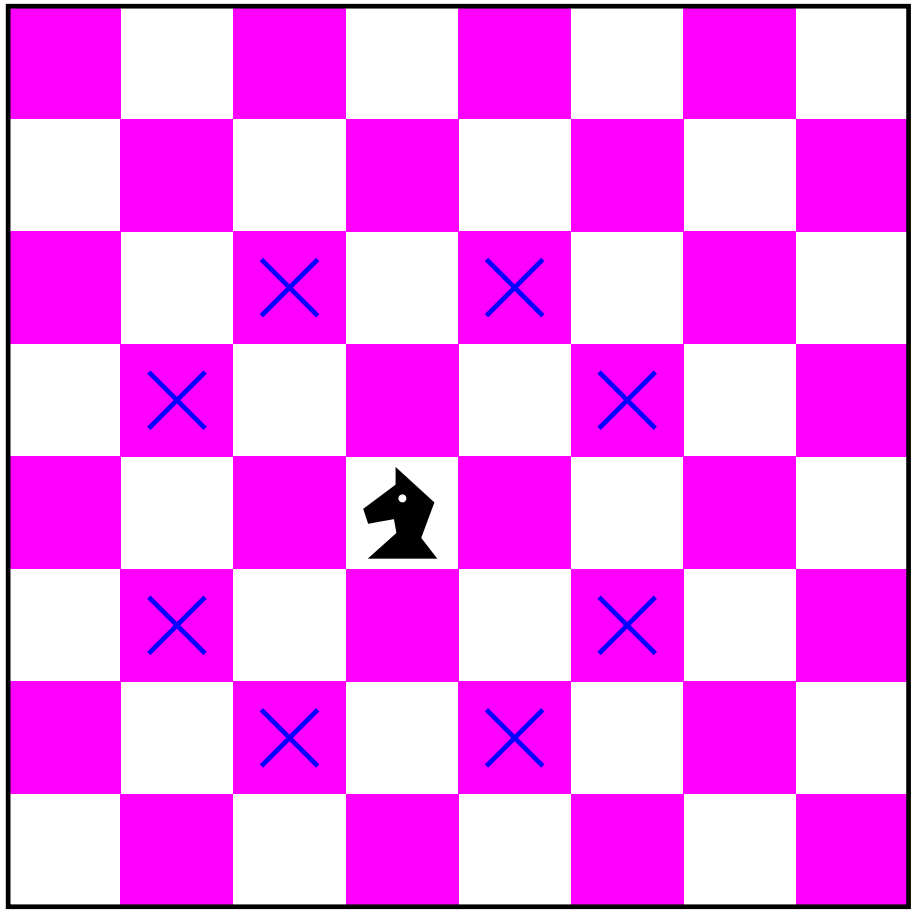
- je splněna ve všech počátečních stavech
- pokud je splněna v nějakém stavu a z tohoto stavu systém přejde jedním krokem do nějakého dalšího stavu, bude splněna i v tomto stavu

Invarianty


dosažitelné stavy
všechny stavy
stavy,
kde platí invariant

Invarianty

Příklad: Budeme skákat s figurkou jezdce po šachovnici a zároveň budeme počítat počty provedených tahů, přičemž začínáme na nějakém bílém poli v nejlevějším sloupci:



Invarianty

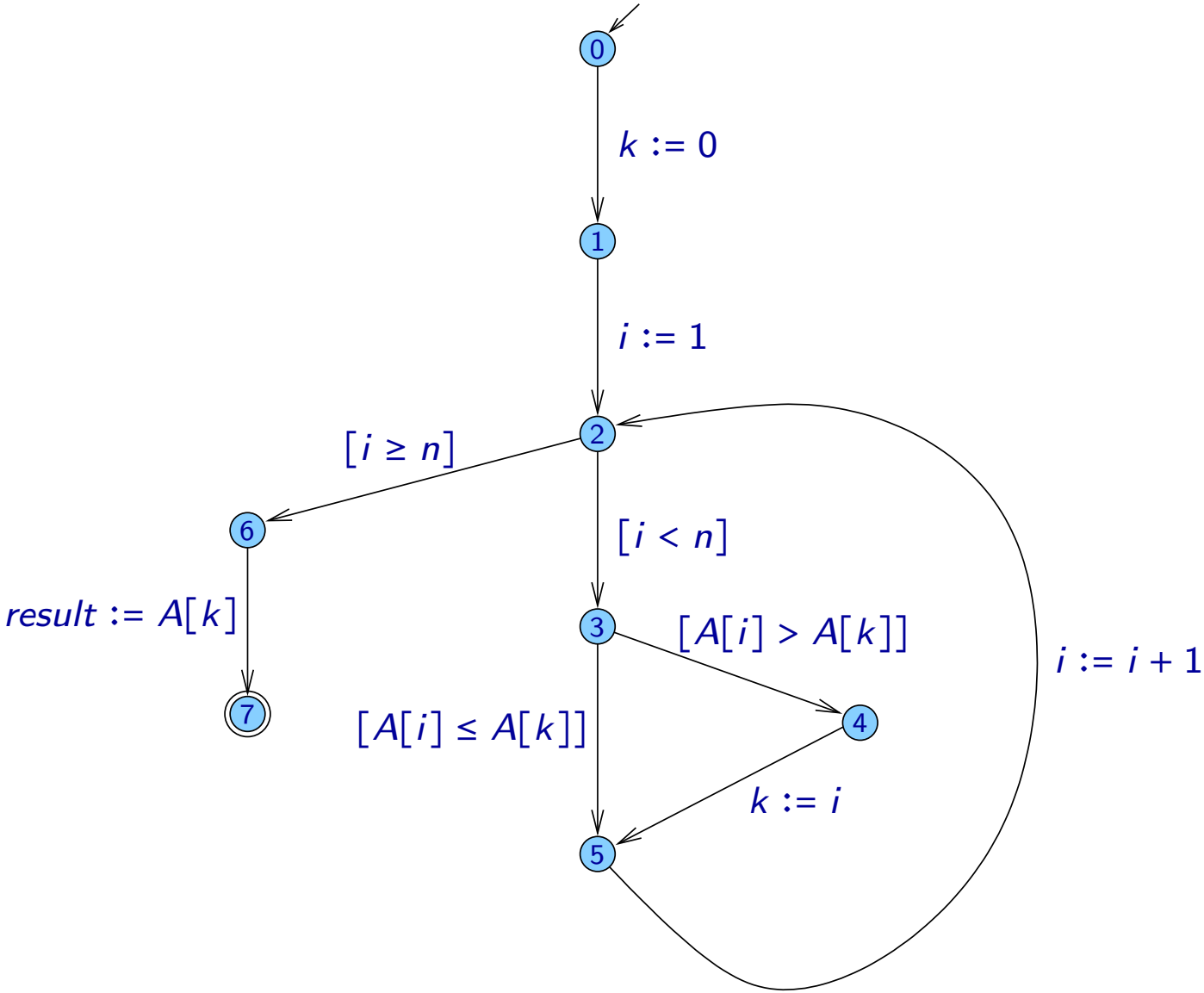
- **Stavy** — dvojice skládající se z aktuální pozice figurky jezdce na šachovnici a hodnoty čítače udávající počet zatím provedených tahů
- **Přechody** — provedení jednoho tahu jezdcem (podle pravidel šachu) a zvýšení čítače o jedna
- **Počáteční stavy** — jezdec se nachází na některém bílém poli v nejlevějším sloupci a hodnota čítače ja 0

Platí zde například následující invariant:

- jestliže je hodnota čítače sudá, jezdec se nachází na bílém poli
- jestliže je hodnota čítače lichá, jezdec se nachází na černém poli

Invarianty

Příklad: Algoritmus FIND-MAX reprezentovaný formou grafu řídícího toku



Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0,\ \langle A\text{: } [3, 8, 1, 3, 6],\ \ n\text{: } 5,\ \ i\text{: } ?,\ \ k\text{: } ?,\ \ \mathit{result}\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0,\ \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } ?,\ \mathit{result}\text{: } ?\rangle)$
α_1: $(1,\ \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } 0,\ \mathit{result}\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } ?,\ \mathit{result}\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } 0,\ \mathit{result}\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 0,\ \mathit{result}\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } ?, \; \mathit{result}\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } 0, \; \mathit{result}\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; \mathit{result}\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; \mathit{result}\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } ?, \; result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } ?, \; result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 1, \; result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } ?, \; result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 1, \; result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 2, \; k\text{: } 1, \; result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } ?,\ result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } 0,\ result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 0,\ result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 0,\ result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 0,\ result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 1,\ result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 2,\ k\text{: } 1,\ result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 2,\ k\text{: } 1,\ result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } ?,\ \mathit{result}\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } ?,\ k\text{: } 0,\ \mathit{result}\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 0,\ \mathit{result}\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 0,\ \mathit{result}\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 0,\ \mathit{result}\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 1,\ k\text{: } 1,\ \mathit{result}\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 2,\ k\text{: } 1,\ \mathit{result}\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 2,\ k\text{: } 1,\ \mathit{result}\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6],\ n\text{: } 5,\ i\text{: } 2,\ k\text{: } 1,\ \mathit{result}\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } ?, \ result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } ?, \ result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } ?, \; result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } ?, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 0, \; result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 1, \; k\text{: } 1, \; result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 2, \; k\text{: } 1, \; result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 2, \; k\text{: } 1, \; result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 2, \; k\text{: } 1, \; result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 3, \; k\text{: } 1, \; result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 3, \; k\text{: } 1, \; result\text{: } ?\rangle)$
α_{11}: $(5, \langle A\text{: } [3,8,1,3,6], \; n\text{: } 5, \; i\text{: } 3, \; k\text{: } 1, \; result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon ?,\ result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon ?,\ k\colon 0,\ result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 0,\ result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 1,\ k\colon 1,\ result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 2,\ k\colon 1,\ result\colon ?\rangle)$
α_9: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{10}: $(3, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{11}: $(5, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 3,\ k\colon 1,\ result\colon ?\rangle)$
α_{12}: $(2, \langle A\colon [3,8,1,3,6],\ n\colon 5,\ i\colon 4,\ k\colon 1,\ result\colon ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } ?, \ result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{11}: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{12}: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{13}: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } ?, \ result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{11}: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{12}: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{13}: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{14}: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

$$
\begin{aligned}
\alpha_0&: (0, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon ?, \ k\colon ?, \ \mathit{result}\colon ?\rangle)\\
\alpha_1&: (1, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon ?, \ k\colon 0, \ \mathit{result}\colon ?\rangle)\\
\alpha_2&: (2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 0, \ \mathit{result}\colon ?\rangle)\\
\alpha_3&: (3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 0, \ \mathit{result}\colon ?\rangle)\\
\alpha_4&: (4, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 0, \ \mathit{result}\colon ?\rangle)\\
\alpha_5&: (5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_6&: (2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 2, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_7&: (3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 2, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_8&: (5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 2, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_9&: (2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 3, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_{10}&: (3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 3, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_{11}&: (5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 3, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_{12}&: (2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 4, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_{13}&: (3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 4, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_{14}&: (5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 4, \ k\colon 1, \ \mathit{result}\colon ?\rangle)\\
\alpha_{15}&: (2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 5, \ k\colon 1, \ \mathit{result}\colon ?\rangle)
\end{aligned}
$$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon ?, \ k\colon ?, \ result\colon ?\rangle)$
α_1: $(1, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon ?, \ k\colon 0, \ result\colon ?\rangle)$
α_2: $(2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 0, \ result\colon ?\rangle)$
α_3: $(3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 0, \ result\colon ?\rangle)$
α_4: $(4, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 0, \ result\colon ?\rangle)$
α_5: $(5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 1, \ k\colon 1, \ result\colon ?\rangle)$
α_6: $(2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 2, \ k\colon 1, \ result\colon ?\rangle)$
α_7: $(3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 2, \ k\colon 1, \ result\colon ?\rangle)$
α_8: $(5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 2, \ k\colon 1, \ result\colon ?\rangle)$
α_9: $(2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 3, \ k\colon 1, \ result\colon ?\rangle)$
α_{10}: $(3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 3, \ k\colon 1, \ result\colon ?\rangle)$
α_{11}: $(5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 3, \ k\colon 1, \ result\colon ?\rangle)$
α_{12}: $(2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 4, \ k\colon 1, \ result\colon ?\rangle)$
α_{13}: $(3, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 4, \ k\colon 1, \ result\colon ?\rangle)$
α_{14}: $(5, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 4, \ k\colon 1, \ result\colon ?\rangle)$
α_{15}: $(2, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 5, \ k\colon 1, \ result\colon ?\rangle)$
α_{16}: $(6, \langle A\colon [3,8,1,3,6], \ n\colon 5, \ i\colon 5, \ k\colon 1, \ result\colon ?\rangle)$

Invarianty

Výpočet pro vstup $A = [3, 8, 1, 3, 6]$ a $n = 5$ jako posloupnost konfigurací:

α_0: $(0, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } ?, \ result\text{: } ?\rangle)$
α_1: $(1, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } ?, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_2: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_3: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_4: $(4, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 0, \ result\text{: } ?\rangle)$
α_5: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 1, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_6: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_7: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_8: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 2, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_9: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{10}: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{11}: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 3, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{12}: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{13}: $(3, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{14}: $(5, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 4, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{15}: $(2, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 5, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{16}: $(6, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 5, \ k\text{: } 1, \ result\text{: } ?\rangle)$
α_{17}: $(7, \langle A\text{: } [3,8,1,3,6], \ n\text{: } 5, \ i\text{: } 5, \ k\text{: } 1, \ result\text{: } 8\rangle)$

Invarianty

- **Stavy** — konfigurace skládající se ze stavu řídící jednotky a obsahu paměti reprezentovaného hodnotami jednotlivých proměnných.
- **Přechody** — dány jednotlivými instrukcemi na hranách grafu, mění zároveň řídící stav i obsah paměti přiřazováním hodnot do proměnných
- **Počáteční stavy** — všechny možné počáteční konfigurace pro všechny možné vstupní instance, které jsou přípustné podle specifikace problému

Invarianty budou mít formu tvrzení vyjadřujících se o konfiguracích, tj. o stavech řídící jednotky a o hodnotách jednotlivých proměnných, např.

- Pokud je stav řídící jednotky 2, pak v dané konfiguraci platí $1 \leq i \leq n$, $0 \leq k < i$ a $A[k]$ je největší z prvků $A[0], A[1], \ldots, A[i-1]$.

Invarianty

U systémů, kde je součástí konfigurace nějaký řídící stav, může být výhodné formulovat invarianty ve formě:

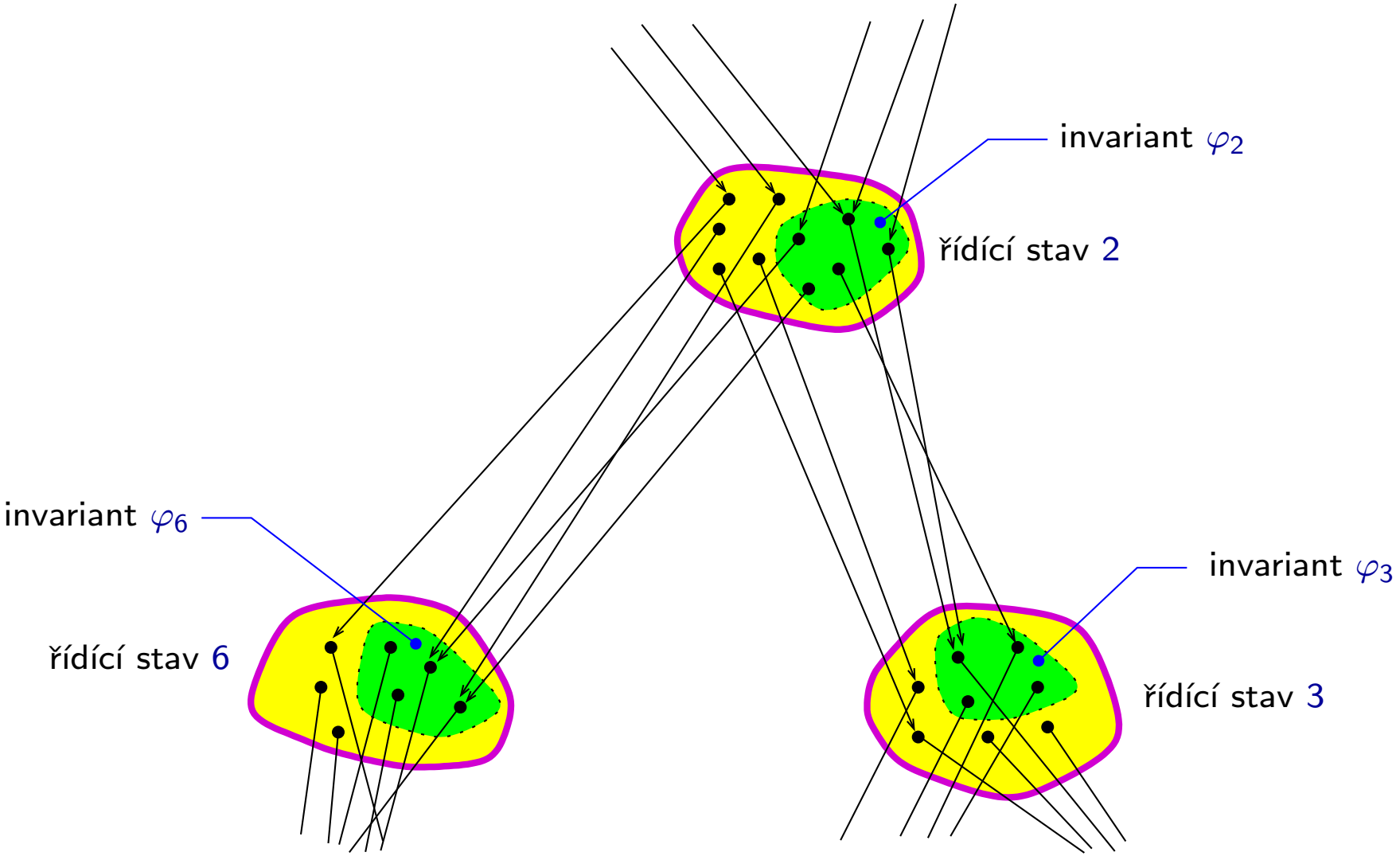
- jestliže je stav řídící jednotky 0, pak platí φ_0
- jestliže je stav řídící jednotky 1, pak platí φ_1

  $\vdots$
- jestliže je stav řídící jednotky r, pak platí φ_r

přičemž tvrzení $\varphi_0, \varphi_1, \ldots, \varphi_r$ se vyjadřují pouze o obsahu paměti, nikoli o řídících stavech.

Konfigurace můžeme rozdělit do (konečně mnoha) skupin podle stavů řídící jednotky.

Invarianty


invariant φ_2
řídící stav 2
invariant φ_6
řídící stav 6
invariant φ_3
řídící stav 3

Invarianty

Invariant — podmínka, která musí být v určitém místě kódu algoritmu vždy (tj. ve všech možných výpočtech pro všechny možné vstupy) splněna v okamžiku, kdy algoritmus tímto místem prochází.

Invarianty můžeme zapisovat formulemi predikátové logiky:

- **volné** proměnné odpovídají proměnným programu
- **valuace** je dána hodnotami proměnných programu v dané konfiguraci

Příklad: Formule

$$(1 \leq i) \wedge (i \leq n)$$

bude platit například v konfiguraci, kde proměnná i má hodnotu 5 a proměnná n má hodnotu 14.

Invarianty

Zjištěné invarianty mohou sloužit k řadě různých účelů:

- Napomohou lepšímu porozumění chování algoritmu.
- Lze pomocí nich ověřit, že nenastanou určité typy chyb — např. přístup k poli mimo povolené rozsahy indexů, dělení nulou, ...

  Ověříme, že v místech v kódu, kde by mohla daná chyba nastat, budou platit invarianty, které zaručí, že proměnné budou mít vždy takové hodnoty, aby chyba nenastala.

  Příklad: Při přístupu k prvku $A[i]$ bude vždy platit $0 \leq i < n$, kde n je délka pole.
- Invariant, který bude platit v koncových konfiguracích, zaručí, že výstup algoritmu bude odpovídat specifikaci v zadání problému.
- Při analýze výpočetní složitosti napomohou při zkoumání toho, kolikrát se provedou které instrukce nebo jak velké množství paměti je při výpočtu potřeba.

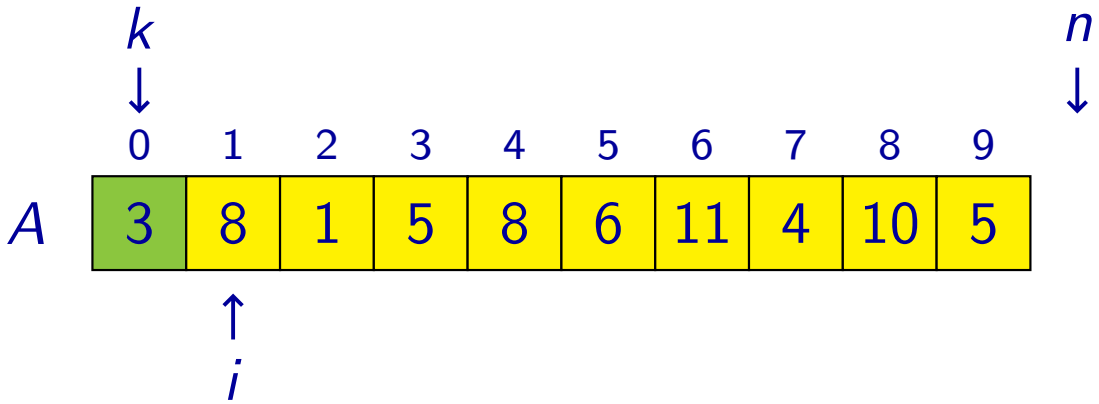
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



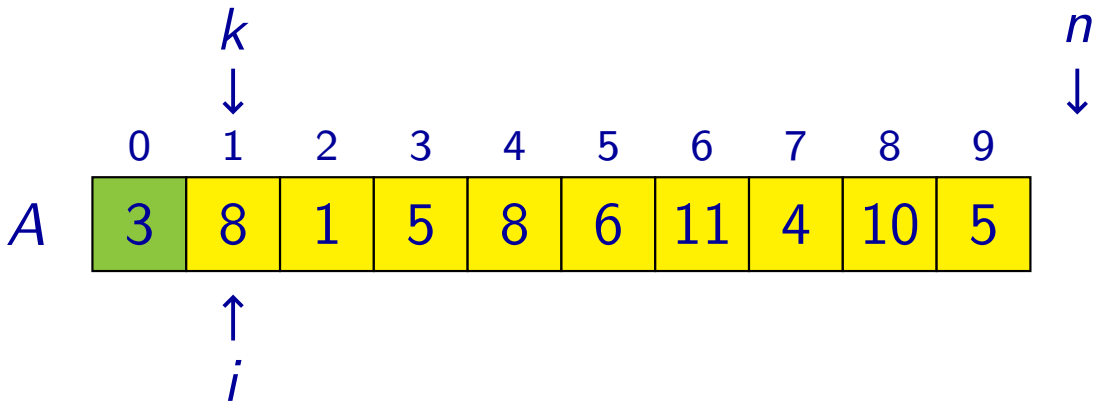
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



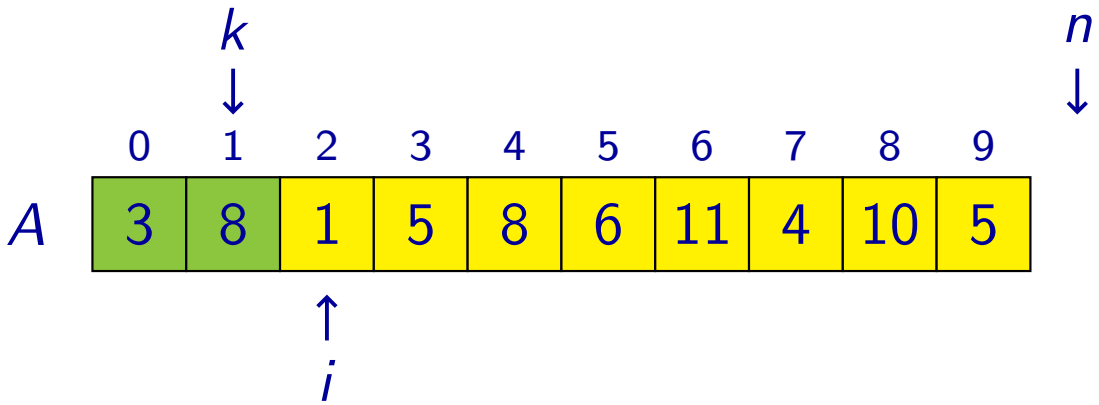
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



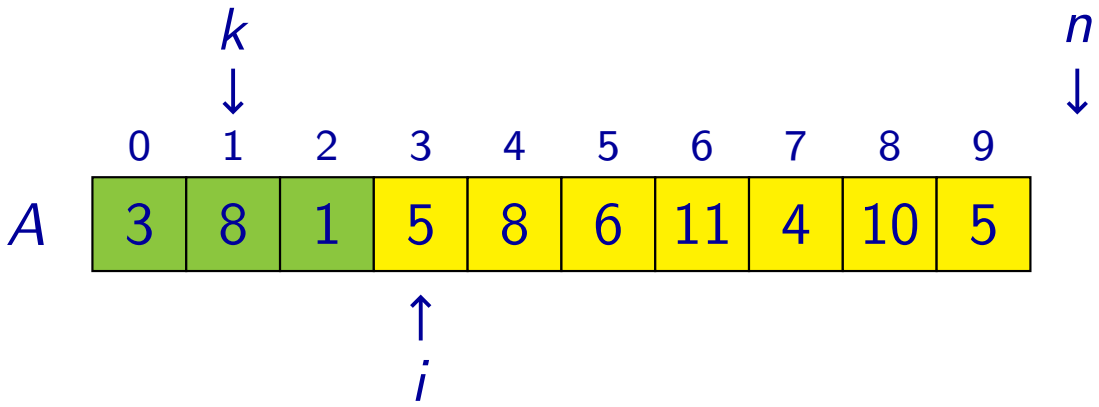
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



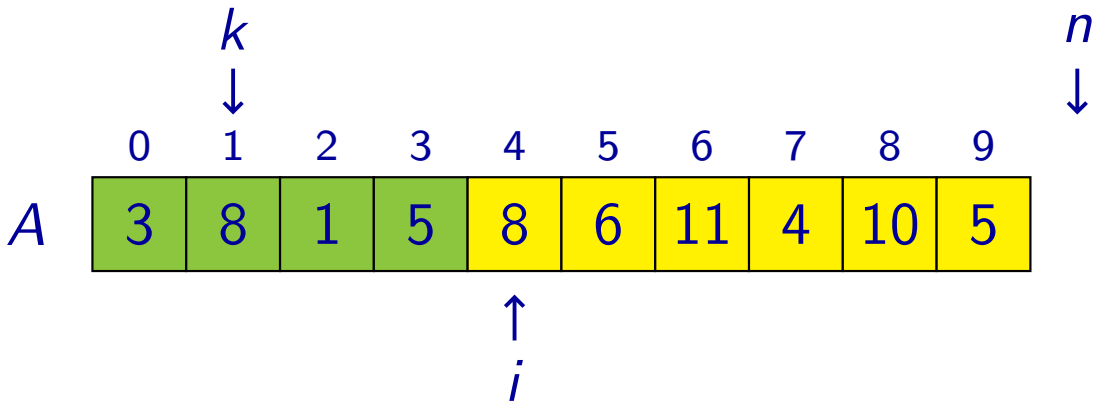
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



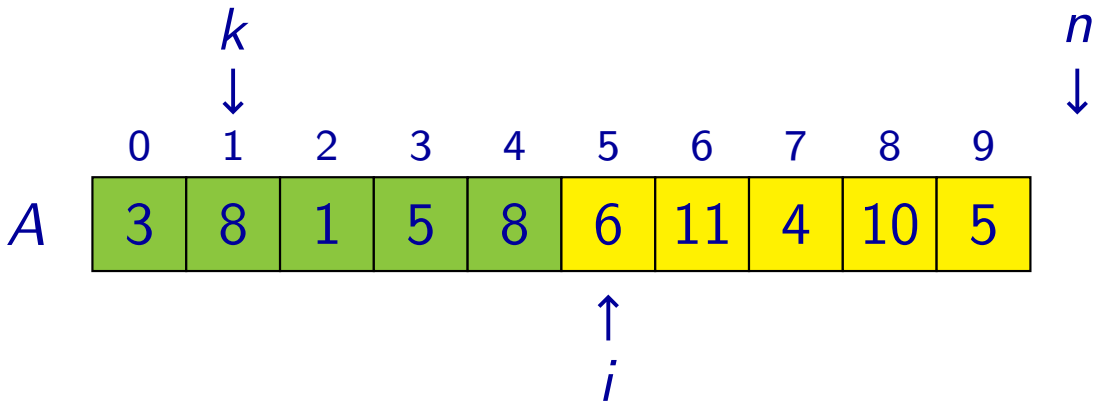
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



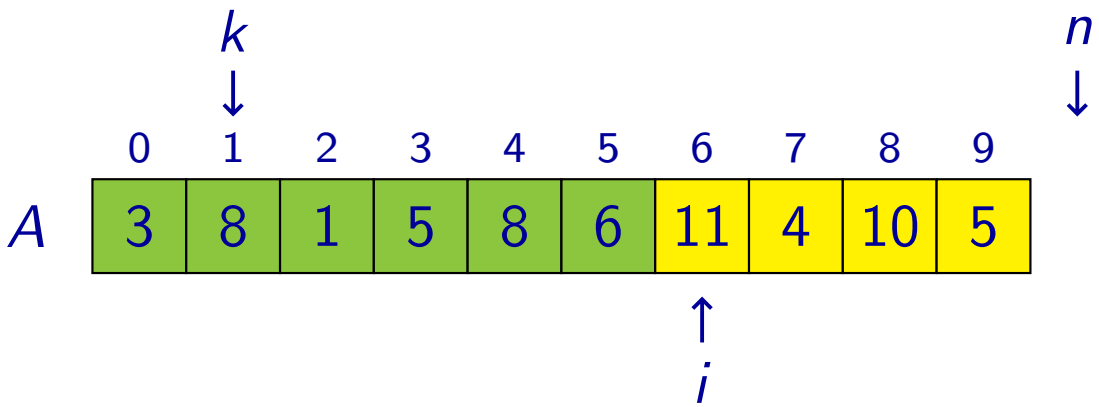
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



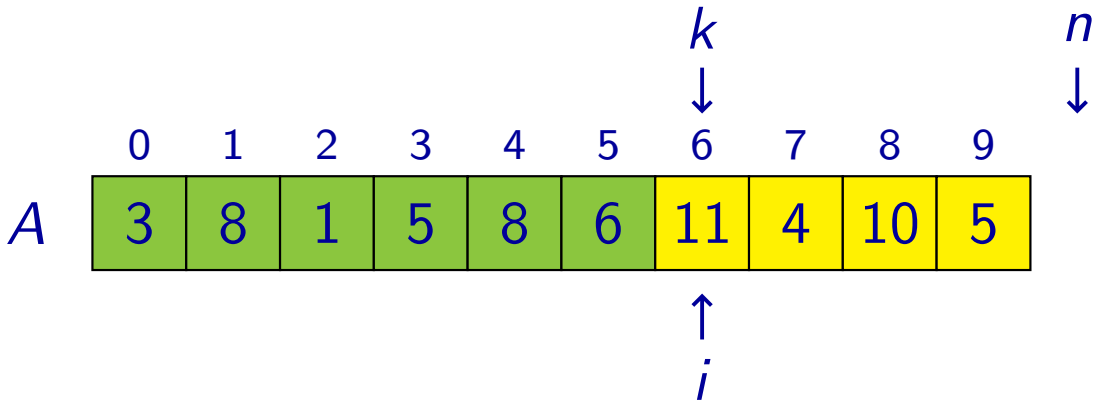
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



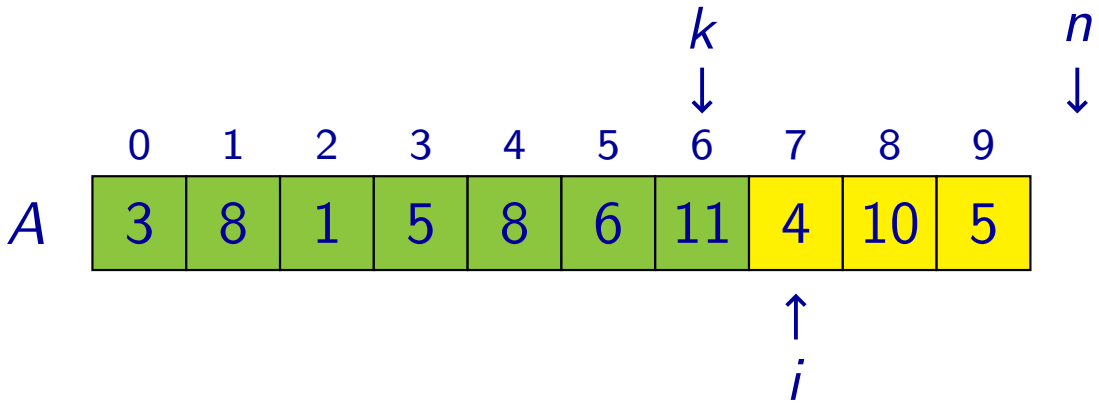
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



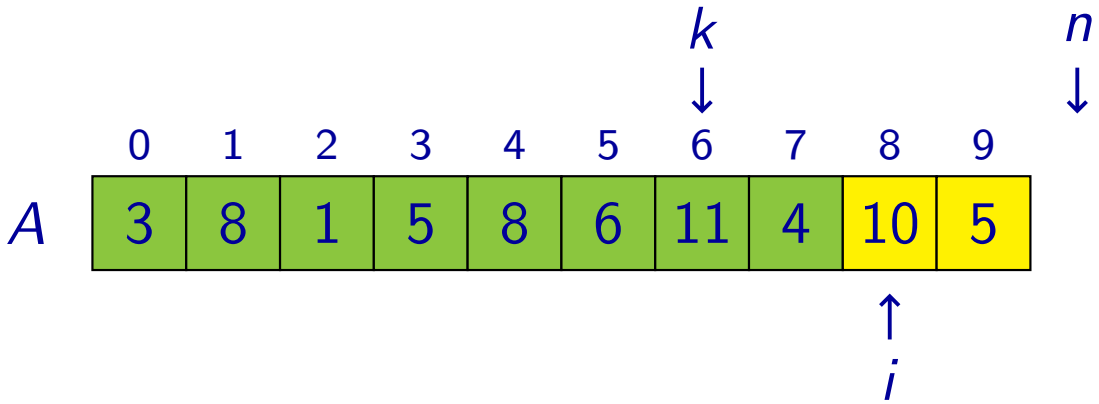
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



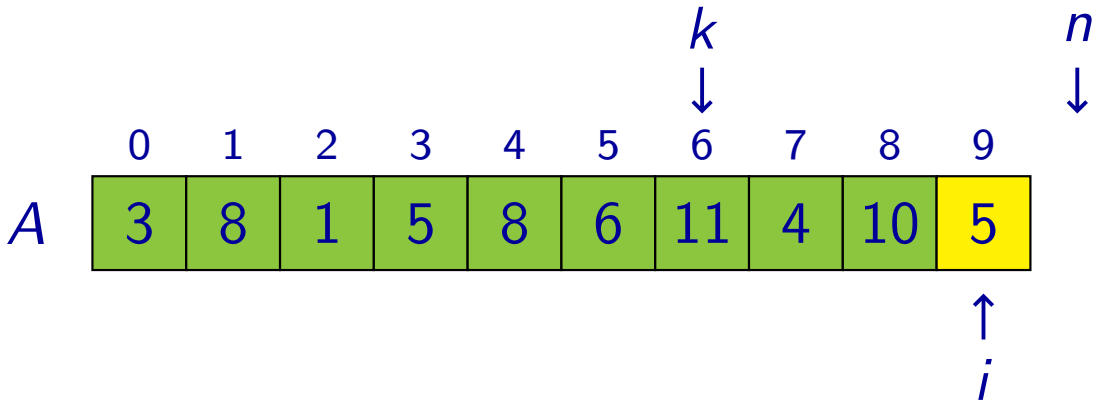
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



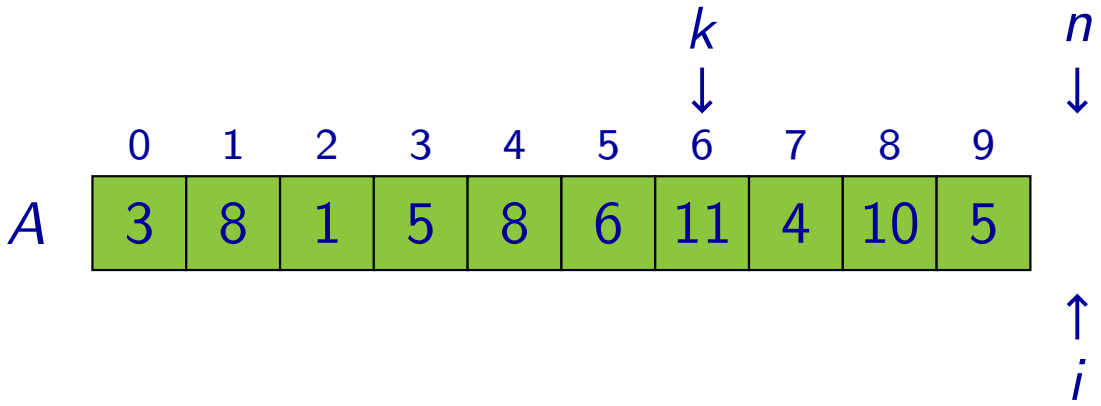
Invarianty

Stanovení invariantů není úplně mechanický proces. Vyžaduje určité pochopení chování algoritmu.

Před formulováním hypotéz o tom, jaké invarianty platí v jednotlivých řídících stavech, může být vhodné se podívat na to, jak se daný algoritmus chová na nějakých konkrétních vstupech.

Příklad: Výpočet algoritmu FIND-MAX pro vstup

$$A = [\,3, 8, 1, 5, 8, 6, 11, 4, 10, 5\,],\ n = 10.$$



Invarianty

Příklady invariantů:

- invariant v řídícím stavu q zapíšeme formulí φ_q

Invarianty v jednotlivých řídících stavech (zatím jen hypotézy):

- φ_0: $(n \geq 1)$
- φ_1: $(n \geq 1) \wedge (k = 0)$
- φ_2: $(n \geq 1) \wedge (1 \leq i \leq n) \wedge (0 \leq k < i)$
- φ_3: $(n \geq 1) \wedge (1 \leq i < n) \wedge (0 \leq k < i)$
- φ_4: $(n \geq 1) \wedge (1 \leq i < n) \wedge (0 \leq k < i)$
- φ_5: $(n \geq 1) \wedge (1 \leq i < n) \wedge (0 \leq k \leq i)$
- φ_6: $(n \geq 1) \wedge (i = n) \wedge (0 \leq k < n)$
- φ_7: $(n \geq 1) \wedge (i = n) \wedge (0 \leq k < n)$

Invarianty

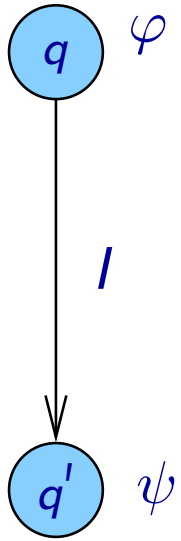
Zkontrolování toho, že invarianty opravdu platí:

- Musíme zkontrolovat, zda invarianty platí v počátečních konfiguracích — toto je většinou jednoduché.
- Pro každou instrukci algoritmu je třeba zkontrolovat, zda za předpokladu, že bude platit příslušný invariant před provedením této instrukce, bude platit i příslušný invariant po provedení této instrukce.

Předpokládejme algoritmus ve formě grafu řídícího toku:

- hrany odpovídají instrukcím
- vezměme si hranu ze stavu q do stavu q' označenou instrukcí I
- řekněme, že (zatím neověřené) invarianty pro stavy q a q' jsou vyjádřeny formulemi φ a ψ

Invarianty



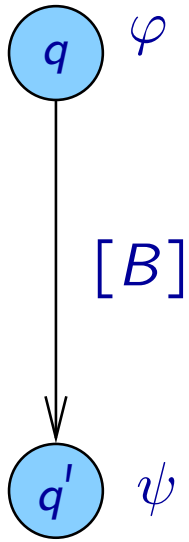
- pro tuto hranu musíme zkontrolovat, že pro všechny konfigurace $\alpha = (q, mem)$ a $\alpha' = (q', mem')$ takové, že $\alpha \xrightarrow{l} \alpha'$, platí, že pokud
  - v konfiguraci α platí φ,

  pak
  - v konfiguraci α' platí ψ

Invarianty

Zkontrolování instrukcí, které jsou testy podmínek:

- hrana označená testem podmínky $[B]$

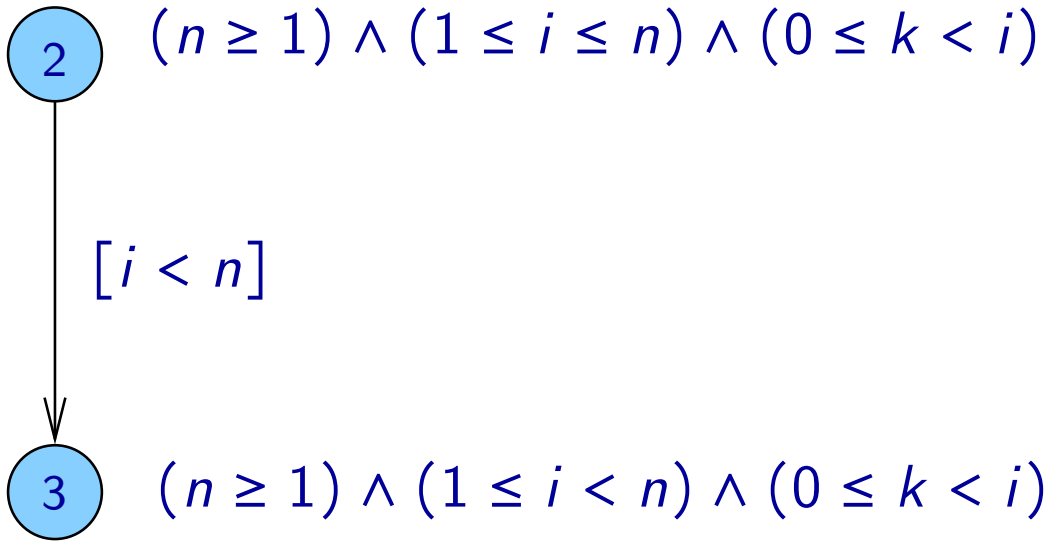


Obsah paměti se nemění, takže stačí ověřit, že platí implikace

$$(\varphi \wedge B) \Rightarrow \psi$$

Invarianty

Příklad:

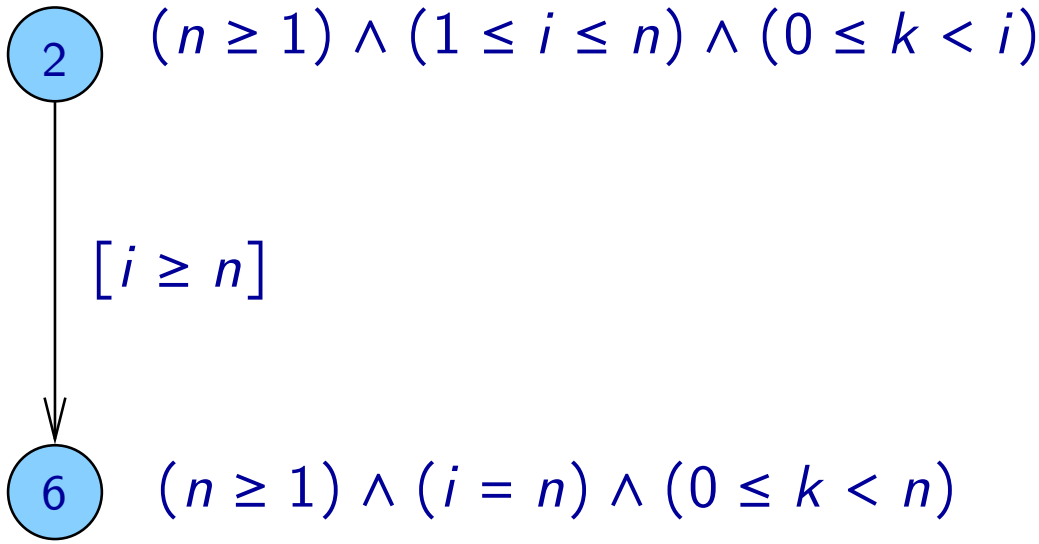


Stačí ověřit, že platí následující implikace:

- Jestliže $(n \geq 1) \wedge (1 \leq i \leq n) \wedge (0 \leq k < i) \wedge (i < n)$, pak $(n \geq 1) \wedge (1 \leq i < n) \wedge (0 \leq k < i)$.

Invarianty

Příklad:



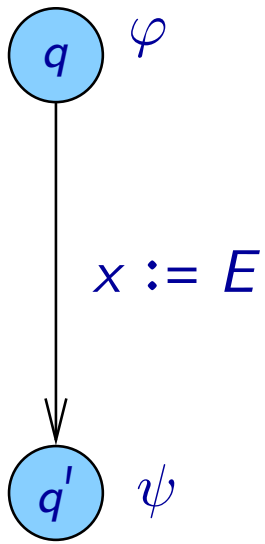
Stačí ověřit, že platí následující implikace:

- Jestliže $(n \geq 1) \wedge (1 \leq i \leq n) \wedge (0 \leq k < i) \wedge (i \geq n)$, pak $(n \geq 1) \wedge (i = n) \wedge (0 \leq k < n)$.

Invarianty

Zkontrolování instrukcí, které přiřazují hodnoty proměnným (mění obsah paměti):

- hrana označená přiřazením $x := E$



Je třeba rozlišovat mezi hodnotou proměnné x před tímto přiřazením a po tomto přiřazení.

Invarianty

Pro následující konstrukce budeme potřebavat operaci **substituce** na formulích:

$$\varphi[E/x]$$

označuje formuli, kterou dostaneme z formule φ dosazením výrazu E za všechny volné výskyty proměnné x ve formuli φ.

Příklad: Řekněme, že φ je formule $(1 \leq i) \wedge (i \leq n)$.

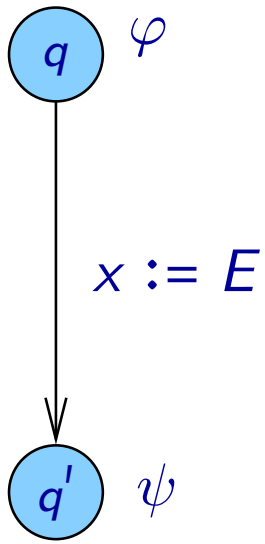
Zápis $\varphi[i'/i]$ pak označuje formuli

$$(1 \leq i') \wedge (i' \leq n)$$

a zápis $\varphi[(i+1)/i]$ formuli

$$(1 \leq i+1) \wedge (i+1 \leq n)$$

Invarianty



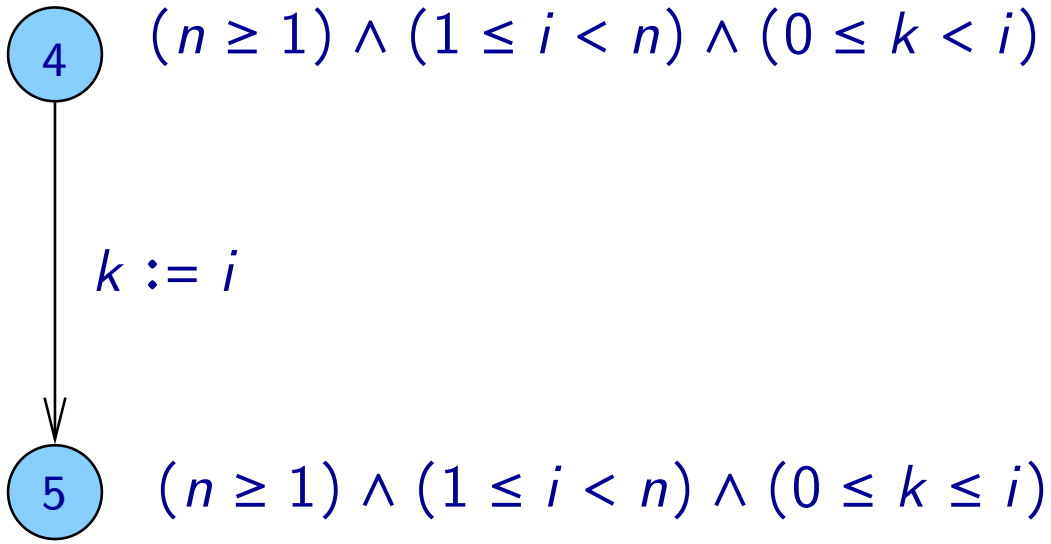
Zavedeme novou proměnnou x' reprezentující hodnotu proměnné x po provedení tohoto přiřazení.

Je třeba ověřit následující implikaci:

$$(\varphi \wedge (x' = E)) \Rightarrow \psi[x'/x]$$

Invarianty

Příklad:

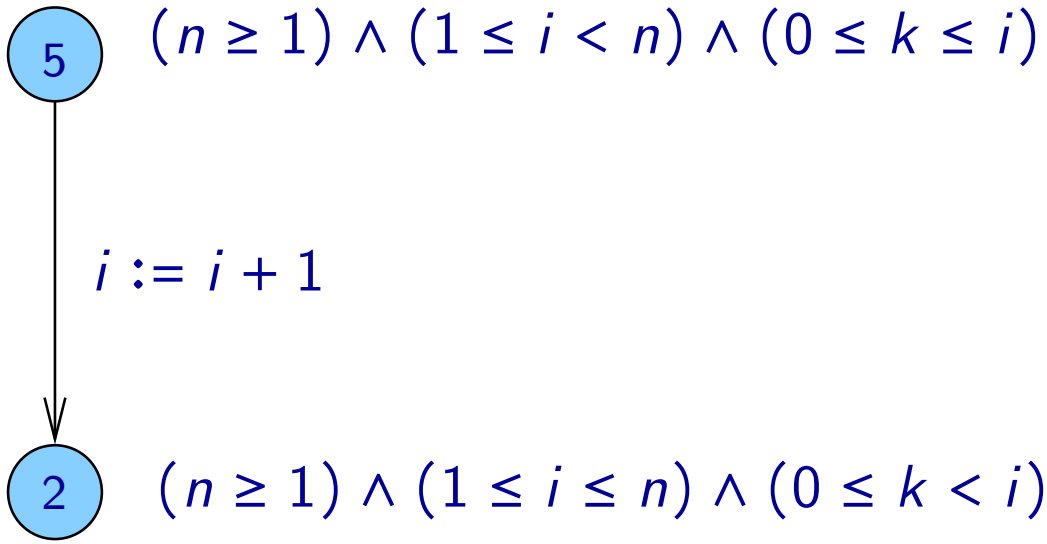


Stačí ověřit, že platí následující implikace:

- Jestliže $(n \geq 1) \wedge (1 \leq i < n) \wedge (0 \leq k < i) \wedge (k' = i)$, pak $(n \geq 1) \wedge (1 \leq i < n) \wedge (0 \leq k' \leq i)$.

Invarianty

Příklad:



Stačí ověřit, že platí následující implikace:

- Jestliže $(n \geq 1) \wedge (1 \leq i < n) \wedge (0 \leq k \leq i) \wedge (i' = i + 1)$, pak $(n \geq 1) \wedge (1 \leq i' \leq n) \wedge (0 \leq k < i')$.

Invarianty

Dokončení ověření toho, že algoritmus FIND-MAX vrací správný výsledek (za předpokladu, že skončí):

- ψ_0: φ_0
- ψ_1: $\varphi_1 \wedge (\forall j \in \mathbb{N})(0 \leq j < 1 \rightarrow A[j] \leq A[k])$
- ψ_2: $\varphi_2 \wedge (\forall j \in \mathbb{N})(0 \leq j < i \rightarrow A[j] \leq A[k])$
- ψ_3: $\varphi_3 \wedge (\forall j \in \mathbb{N})(0 \leq j < i \rightarrow A[j] \leq A[k])$
- ψ_4: $\varphi_4 \wedge (\forall j \in \mathbb{N})(0 \leq j < i \rightarrow A[j] \leq A[k]) \wedge (A[i] > A[k])$
- ψ_5: $\varphi_5 \wedge (\forall j \in \mathbb{N})(0 \leq j \leq i \rightarrow A[j] \leq A[k])$
- ψ_6: $\varphi_6 \wedge (\forall j \in \mathbb{N})(0 \leq j < n \rightarrow A[j] \leq A[k])$
- ψ_7: $\varphi_7 \wedge (\mathit{result} = A[k]) \wedge (\forall j \in \mathbb{N})(0 \leq j < n \rightarrow A[j] \leq \mathit{result}) \wedge (\exists j \in \mathbb{N})(0 \leq j < n \wedge A[j] = \mathit{result})$

Invarianty

Často není třeba specifikovat invarianty ve všech řídících stavech, ale jen v některých „důležitých“ — zejména stavy, kde se vstupuje do nebo vystupuje z cyklů:

Je pak třeba ověřit:

- Že invariant platí před vstupem do cyklu.
- Že pokud invariant platí před provedením cyklu, tak bude platit i po jeho provedení.
- Že invariant platí při opuštění cyklu.

Invarianty

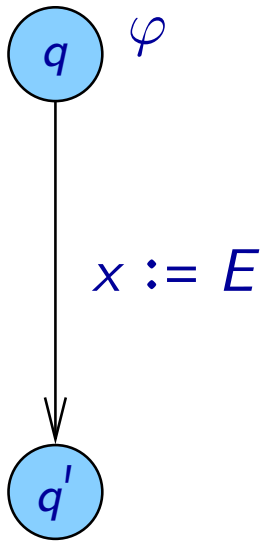
Příklad: V algoritmu FIND-MAX je takovým „důležitým“ stavem stav 2.

Ve stavu 2 platí:

- $n \geq 1$
- $1 \leq i \leq n$
- $0 \leq k < i$
- Pro všechna j taková, že $0 \leq j < i$, platí $A[j] \leq A[k]$.

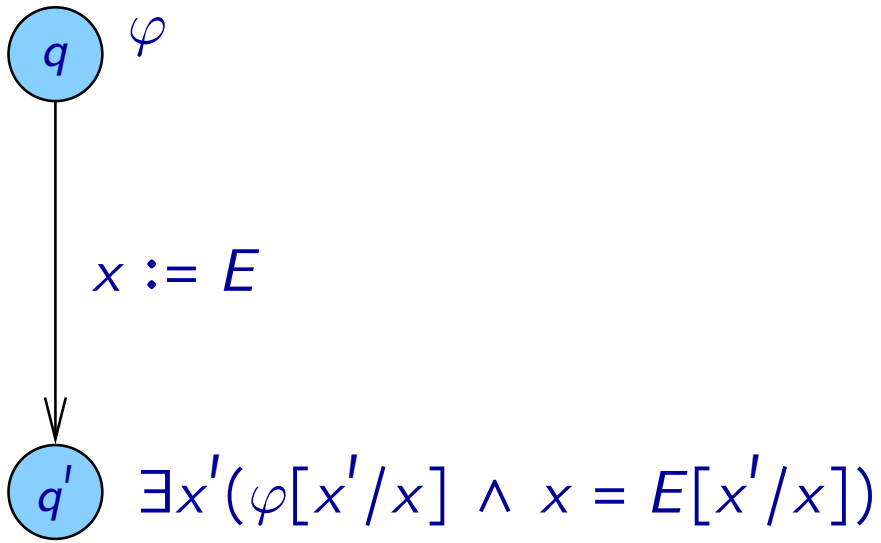
Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



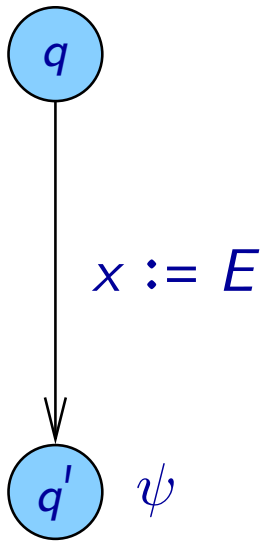
Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



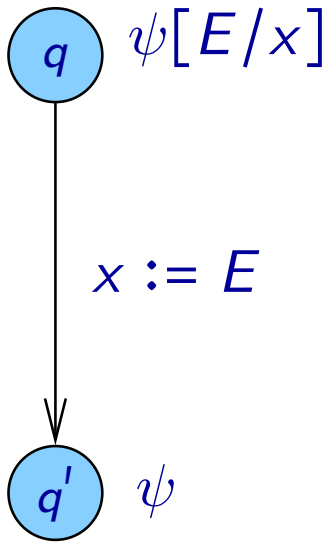
Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



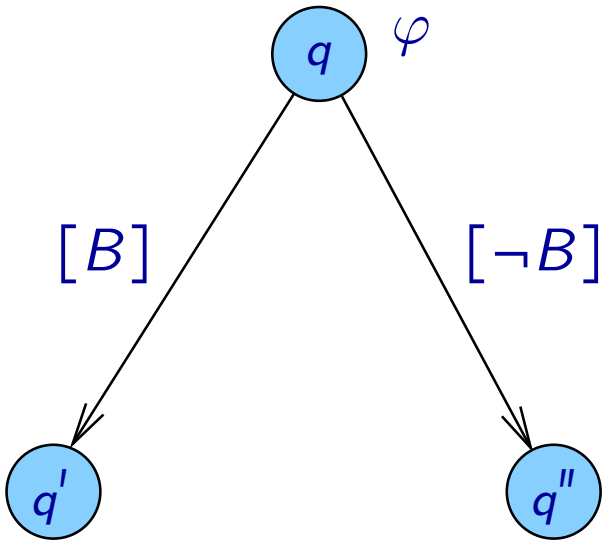
Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



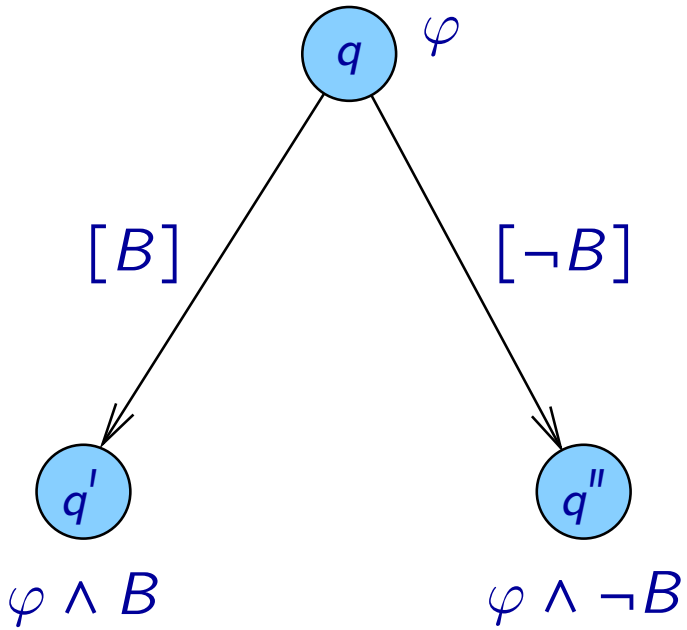
Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



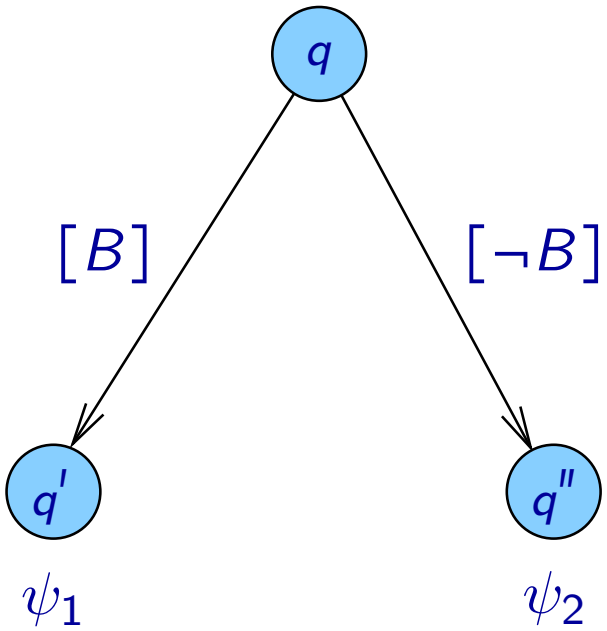
Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



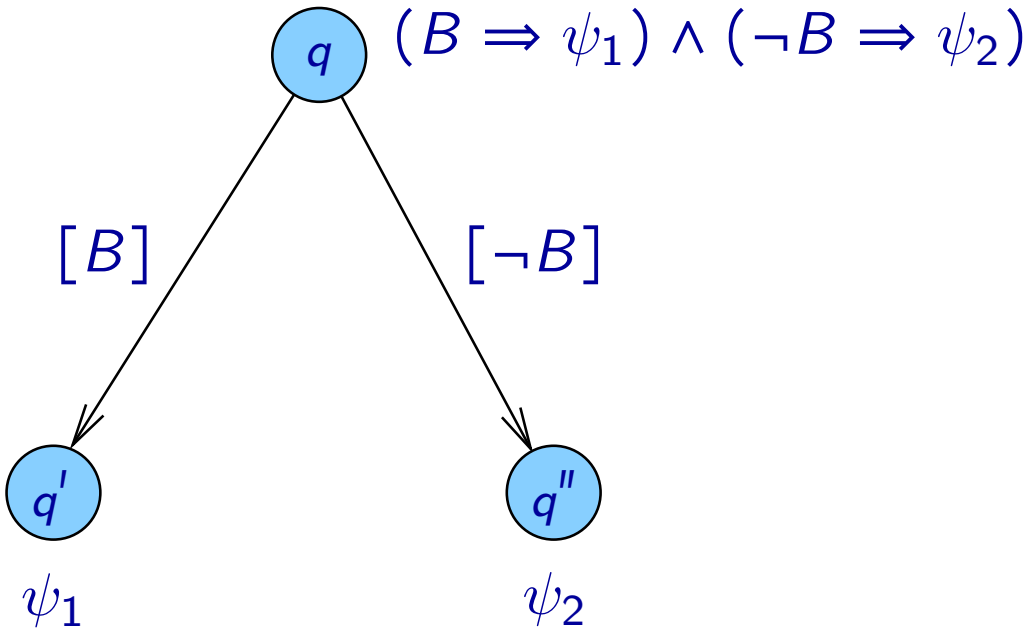
Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



Invarianty

Příklady toho, jak určit invarianty u některých dalších stavů, pokud už u některých stavů invarianty máme:



Invarianty

Příklad:

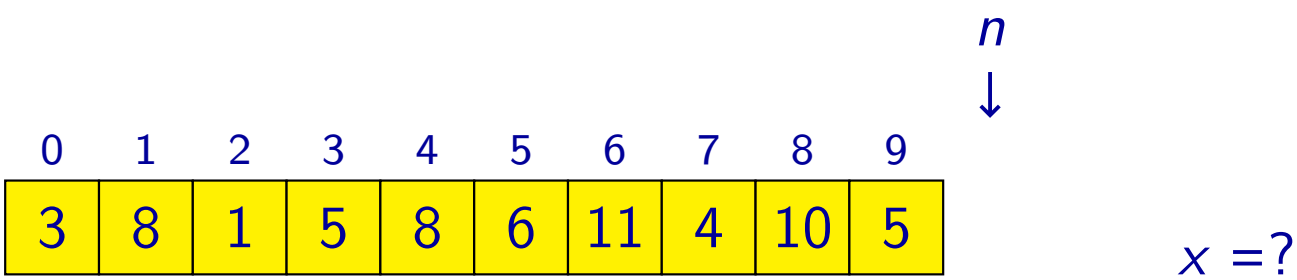
Algoritmus: Třídění přímým vkládáním

```
INSERTION-SORT (A, n):
    for j := 1 to n − 1 do
        x := A[j]
        i := j − 1
        while i ≥ 0 and A[i] > x do
            A[i + 1] := A[i]
            i := i − 1
        A[i + 1] := x
```

Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

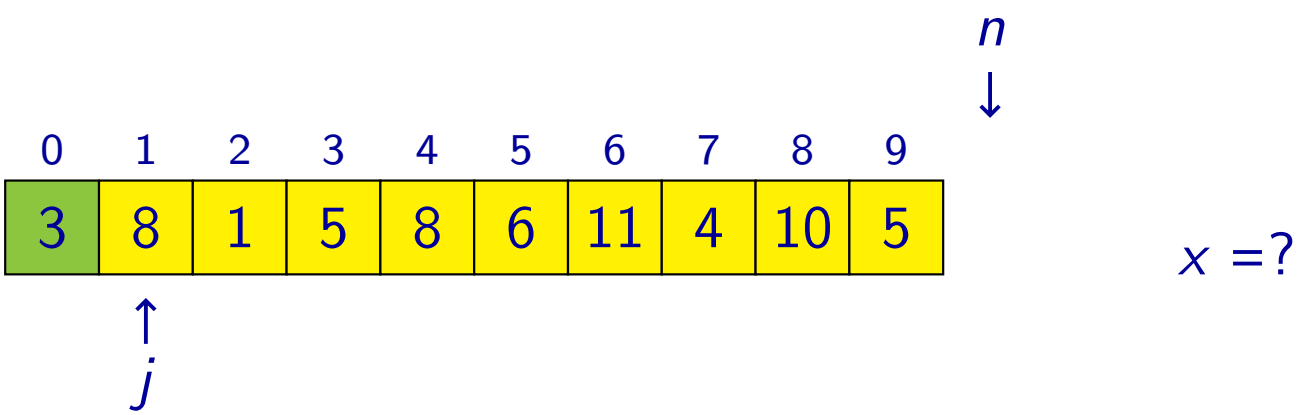
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

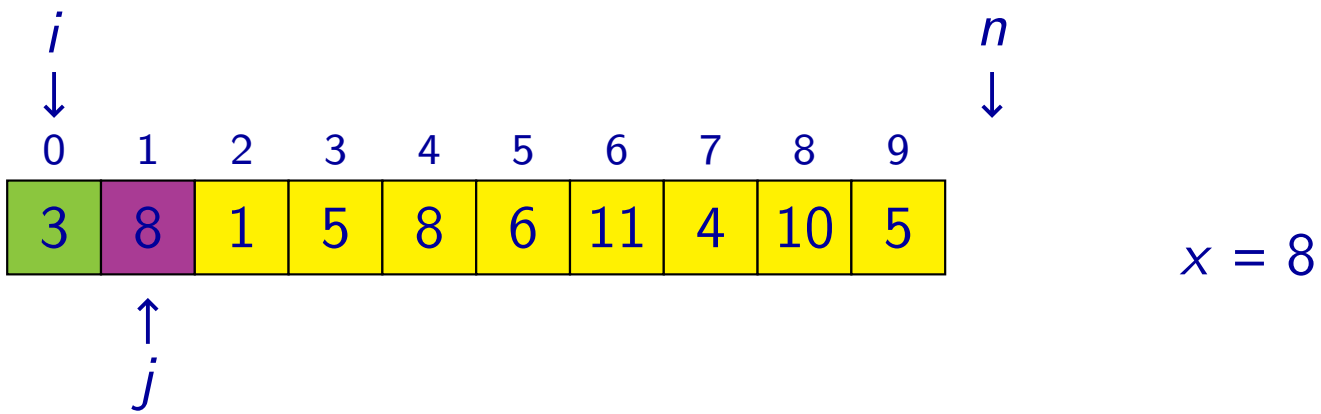
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

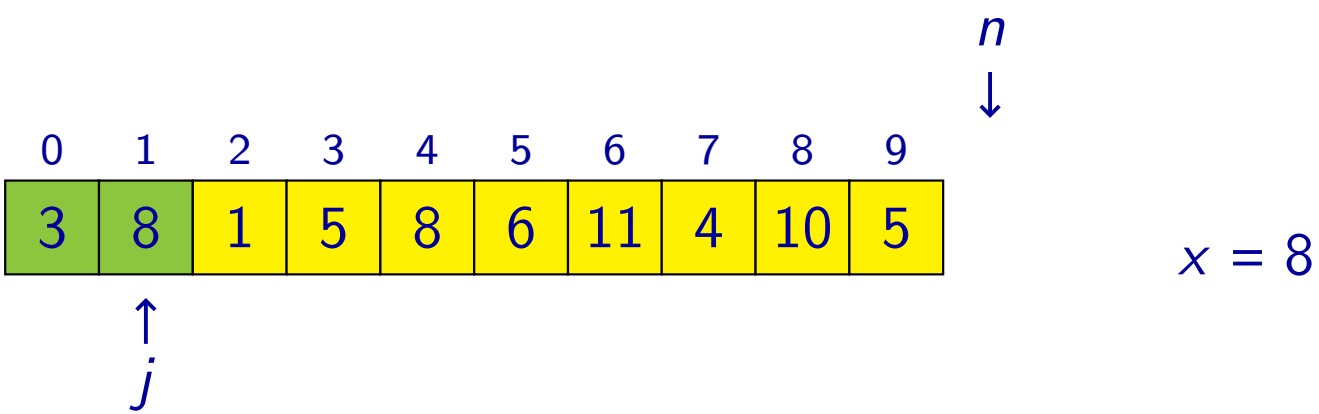
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

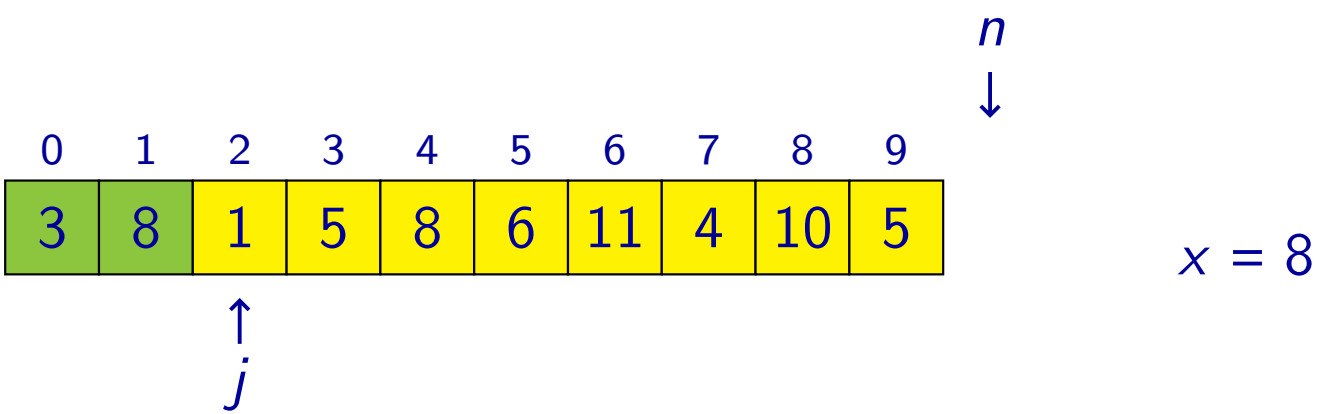
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

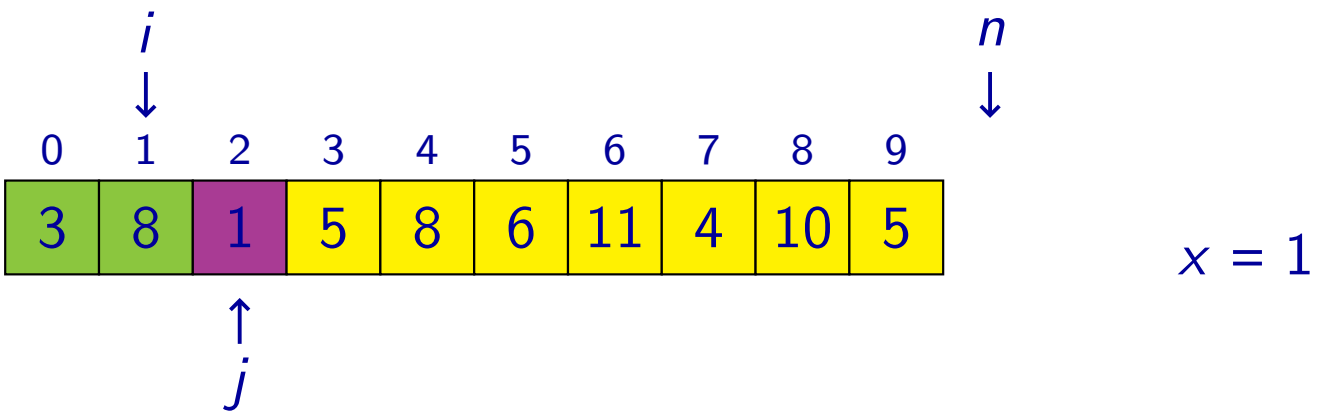
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

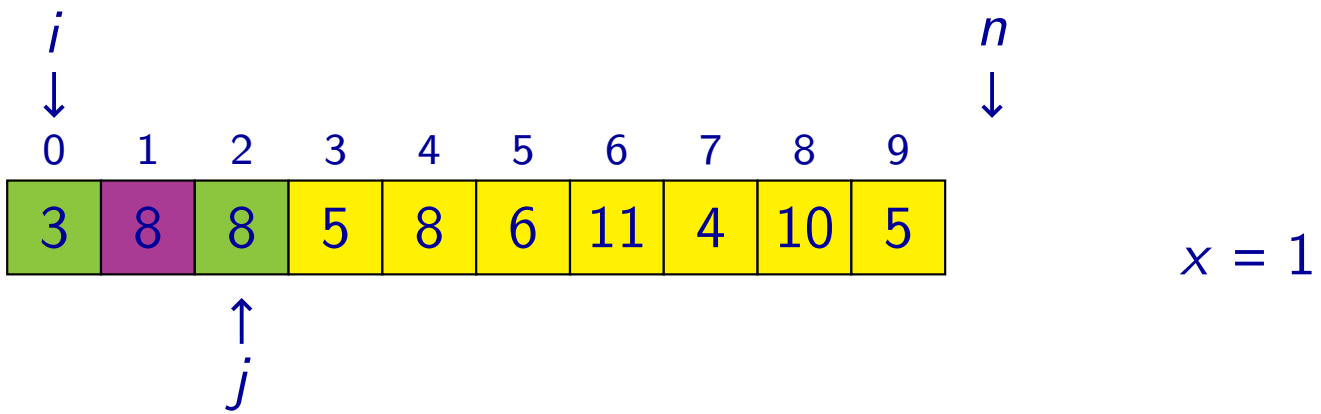
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

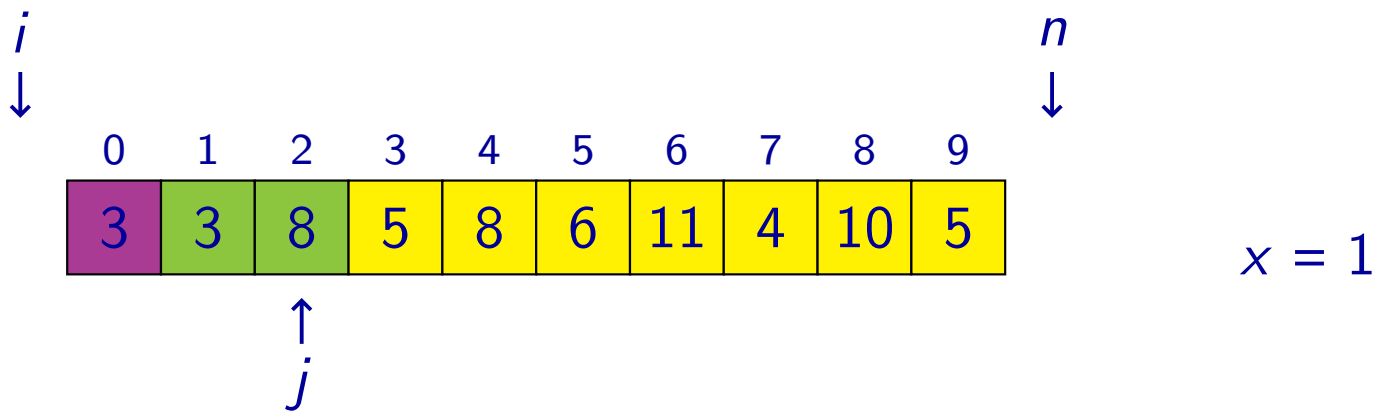
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

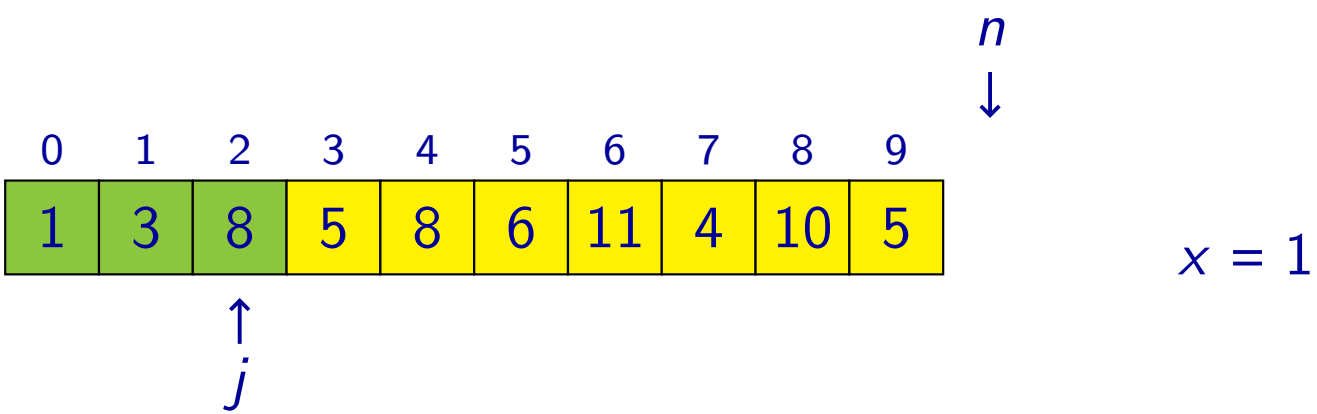
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

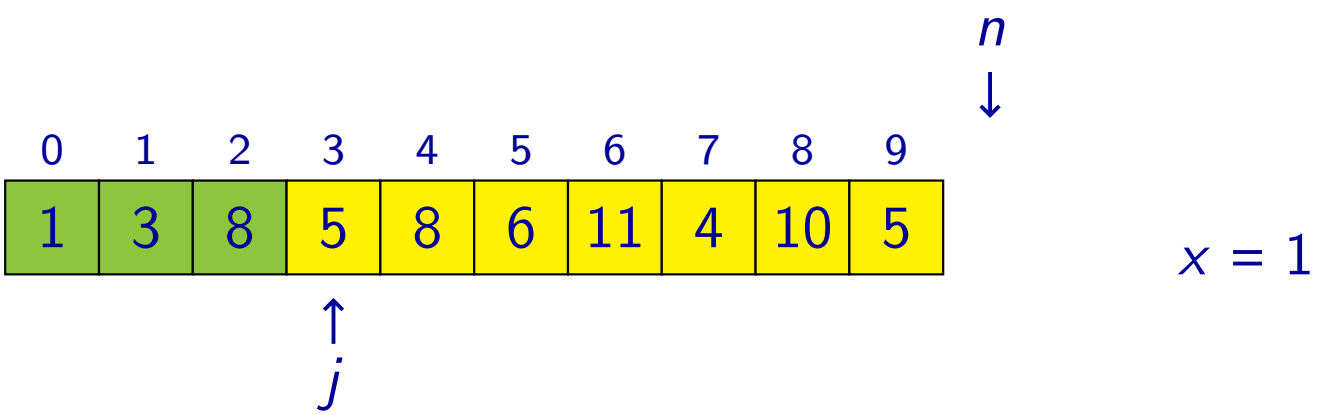
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

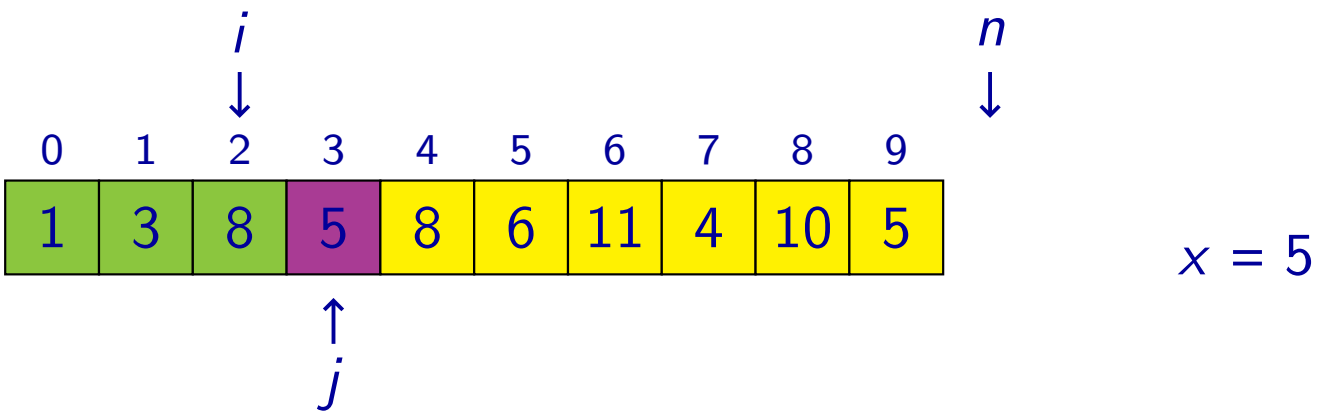
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

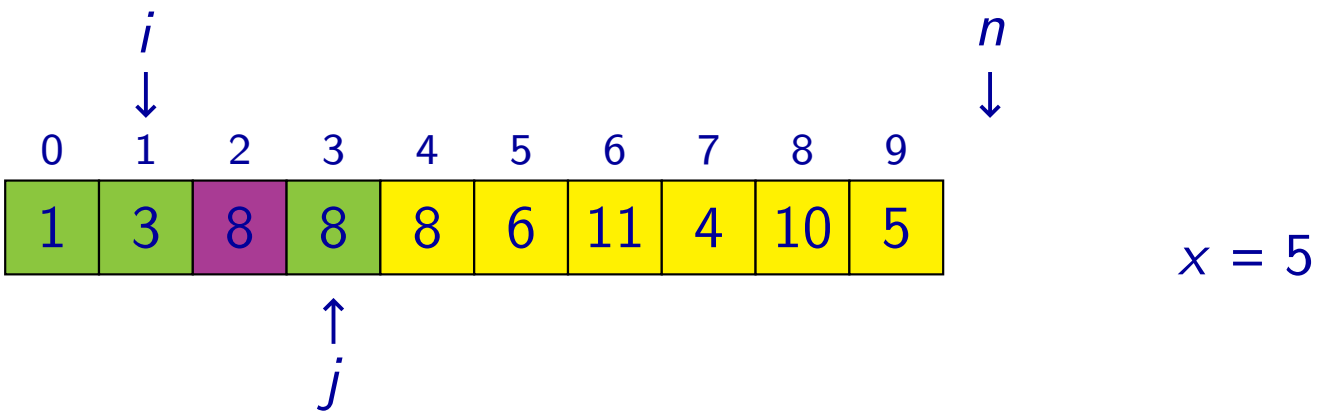
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

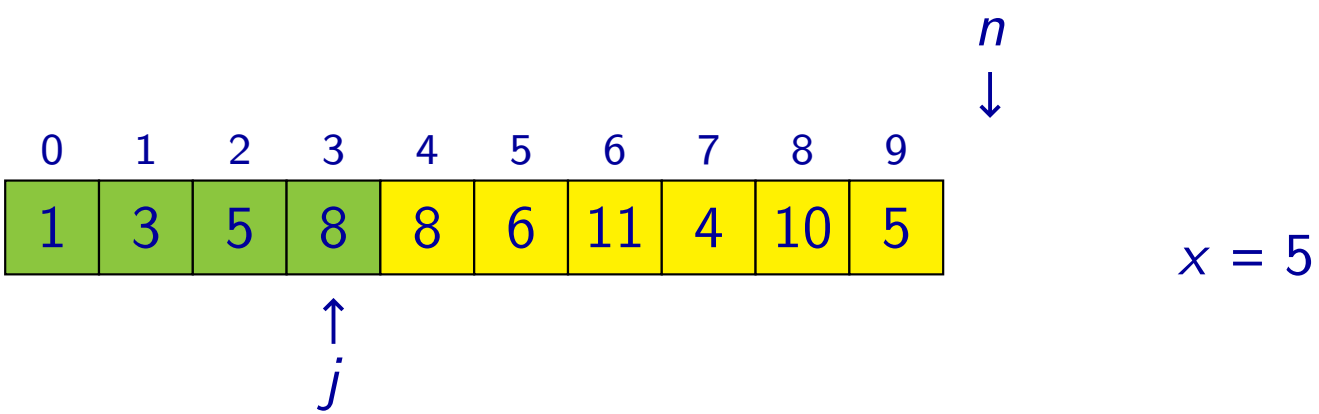
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

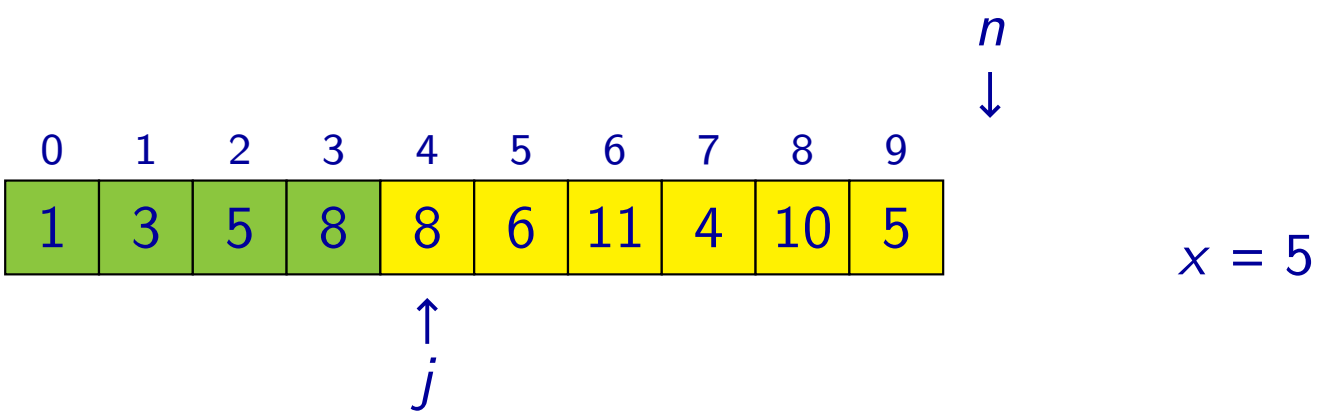
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

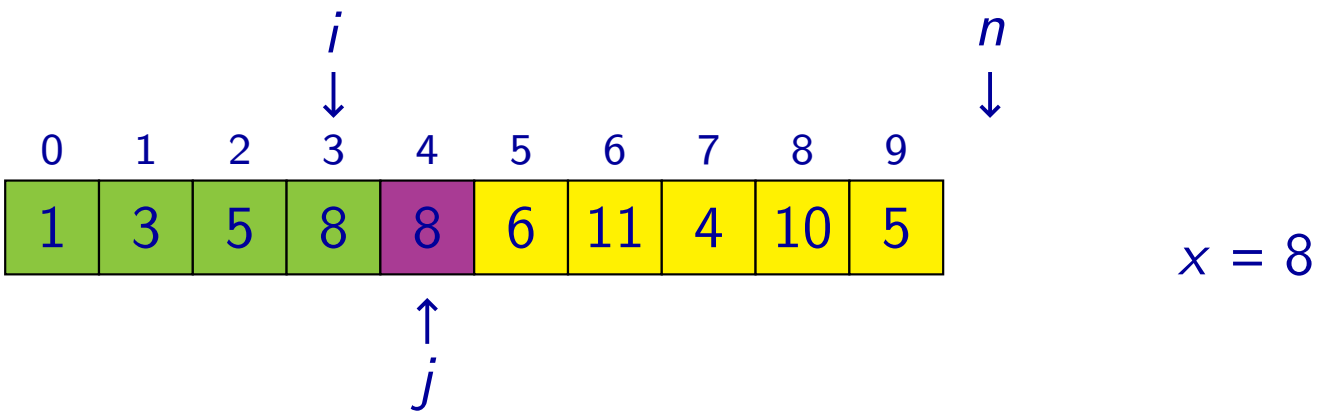
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

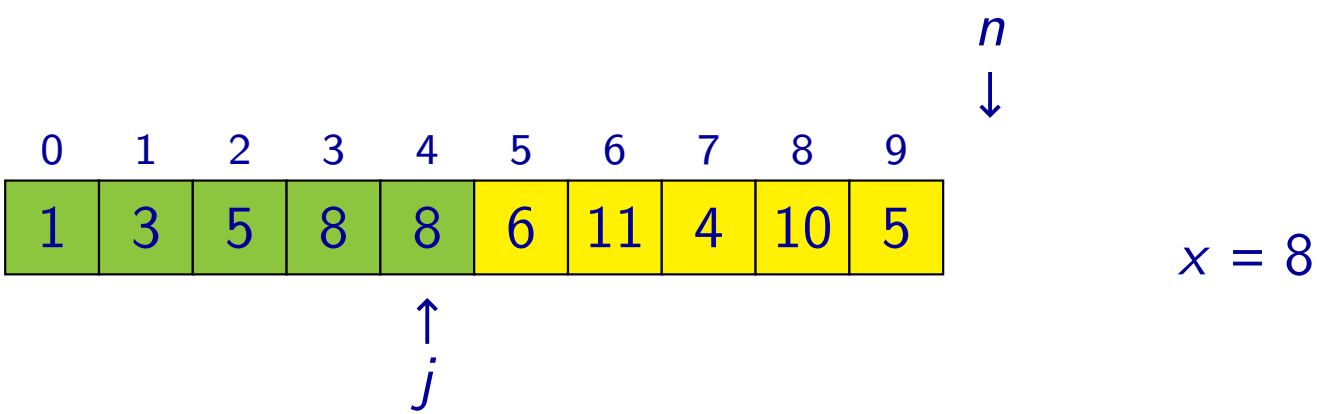
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

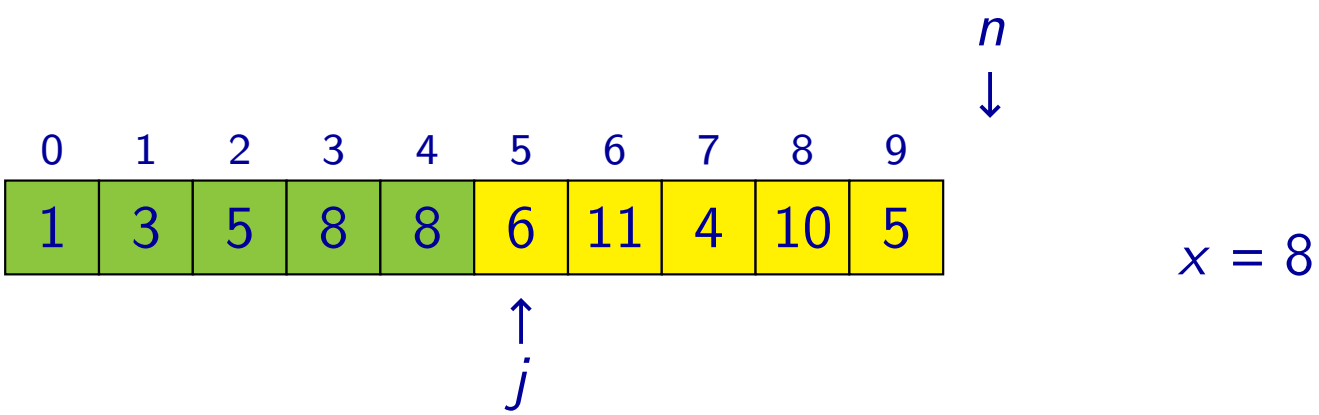
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$

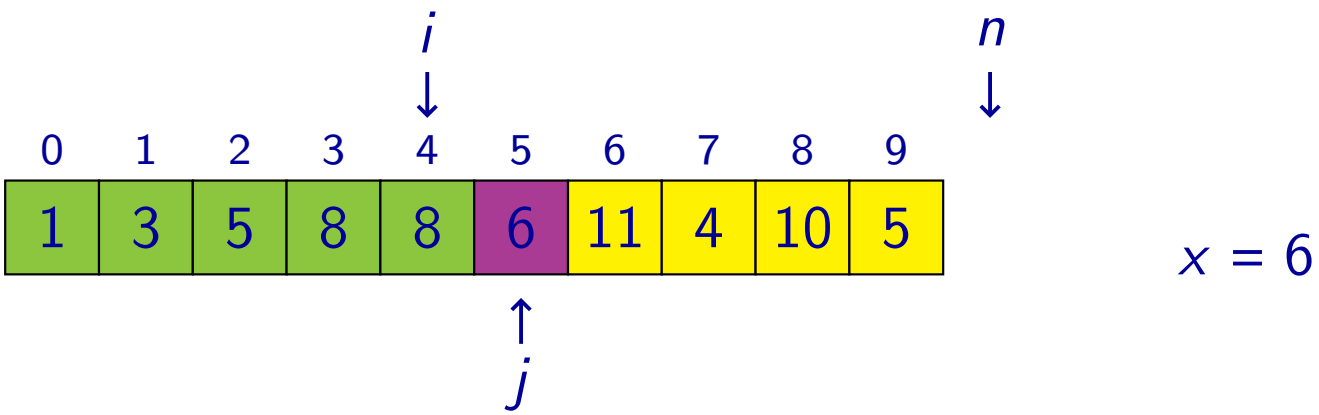


$x = 8$

Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

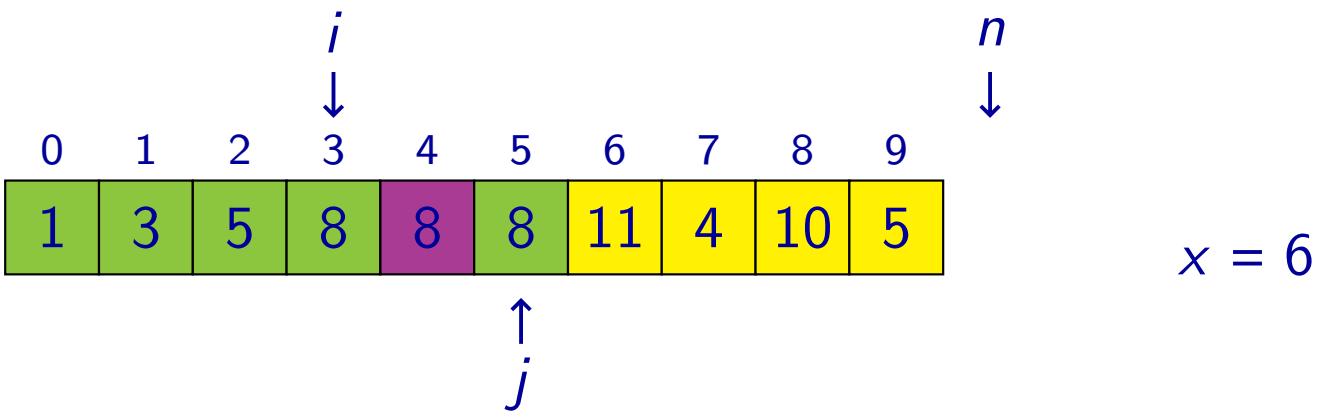
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

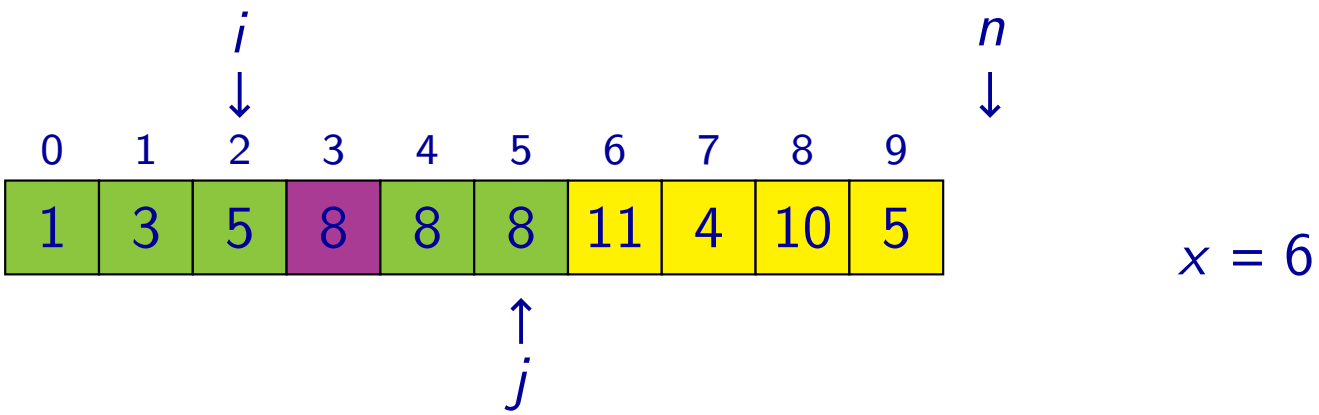
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

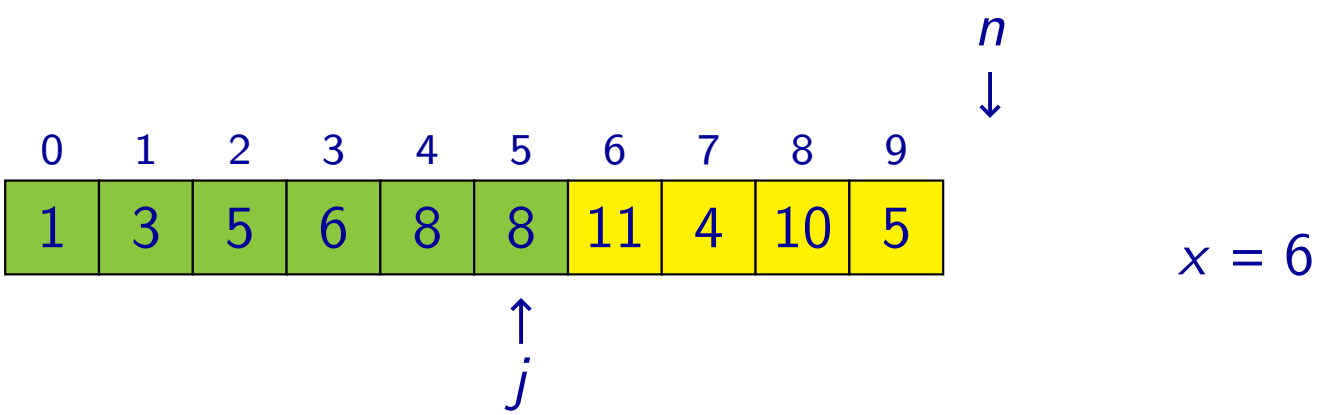
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

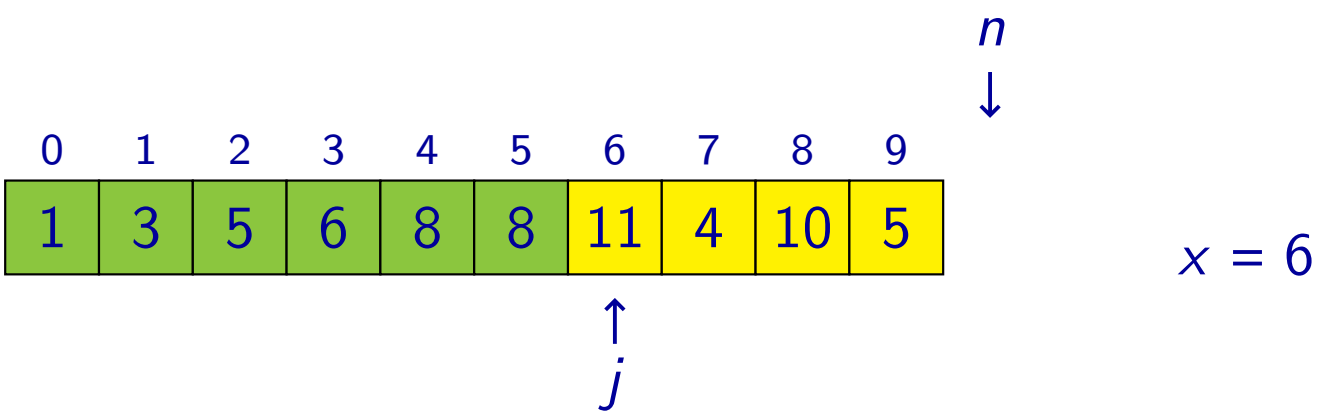
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

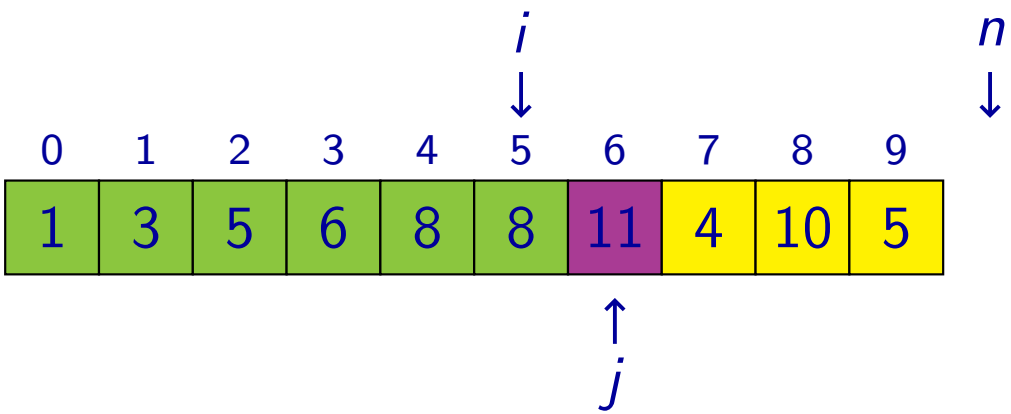
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$

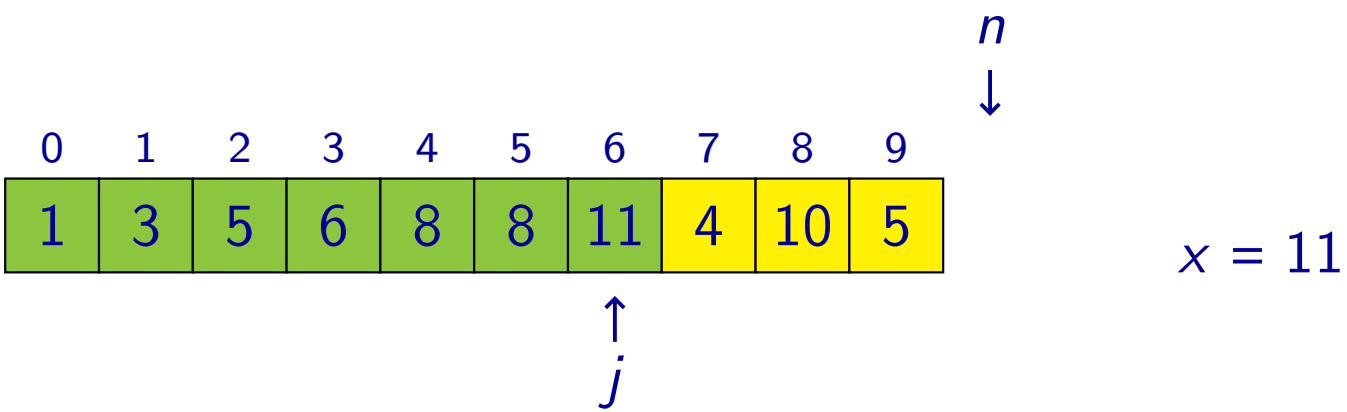


$x = 11$

Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$

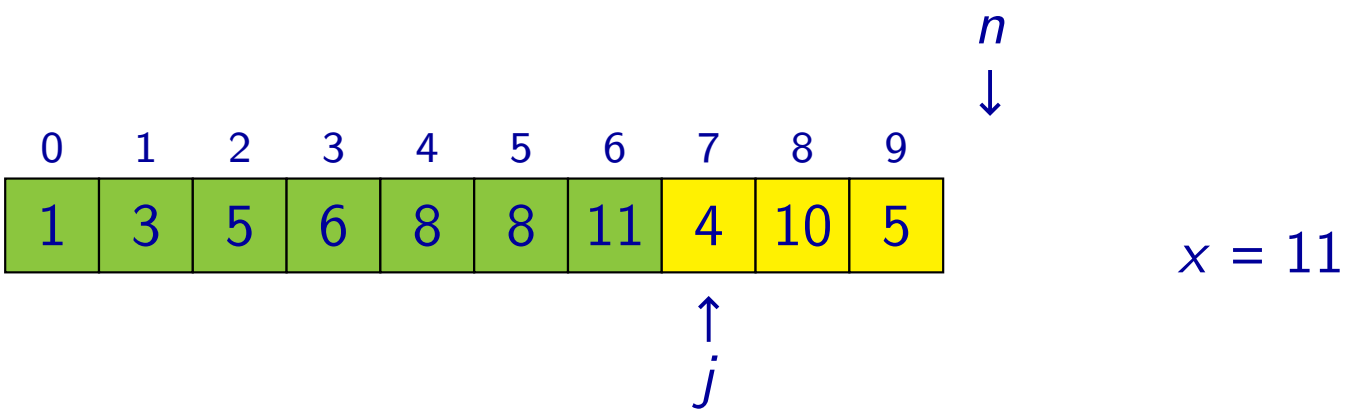


$x = 11$

Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

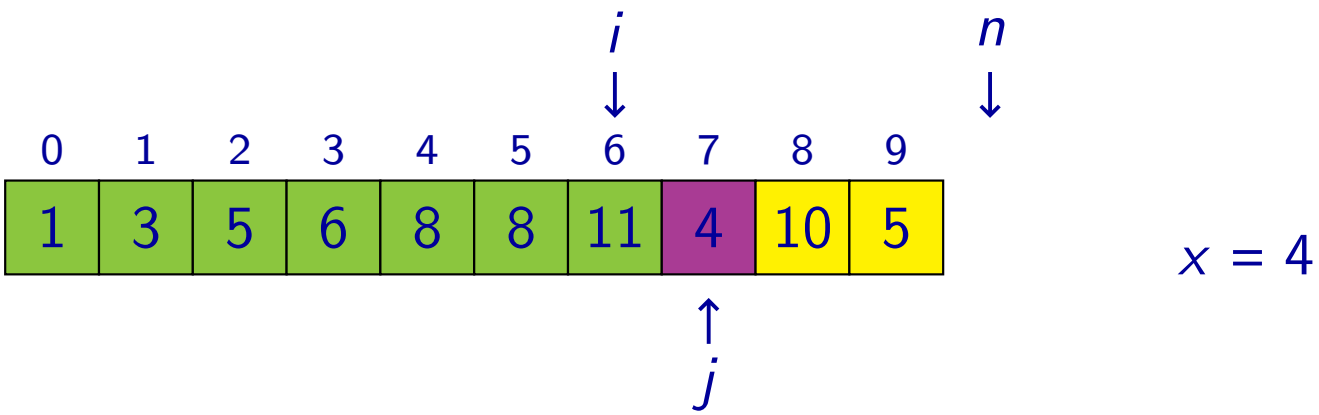
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

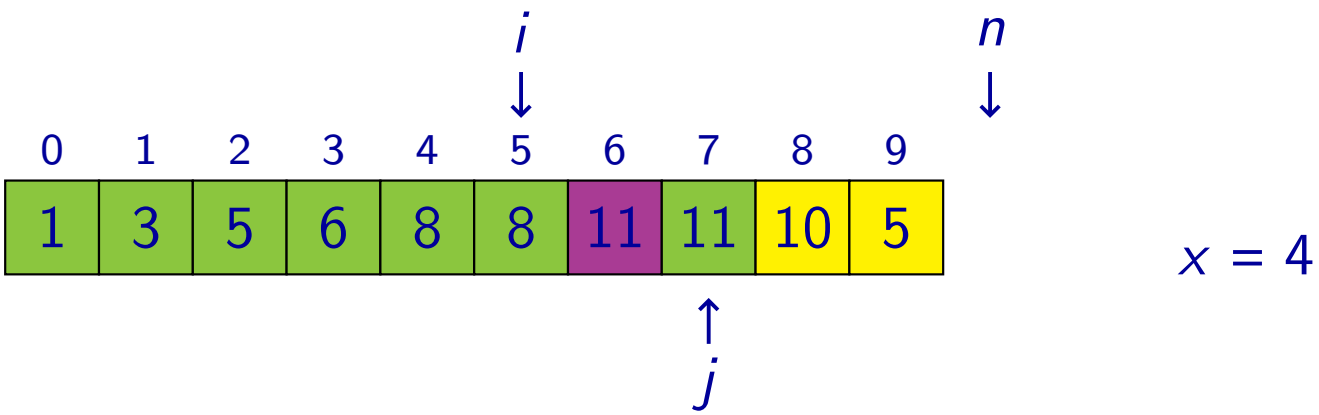
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

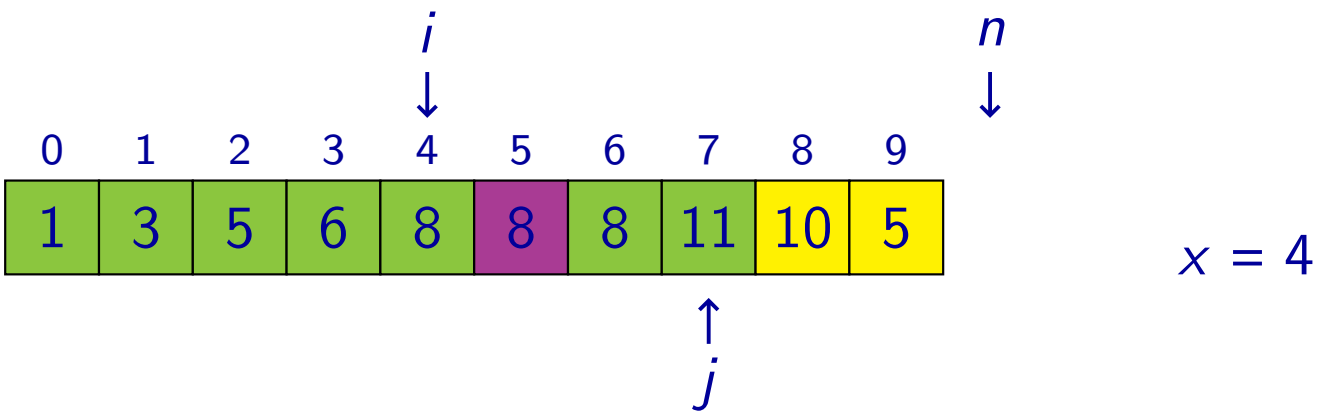
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

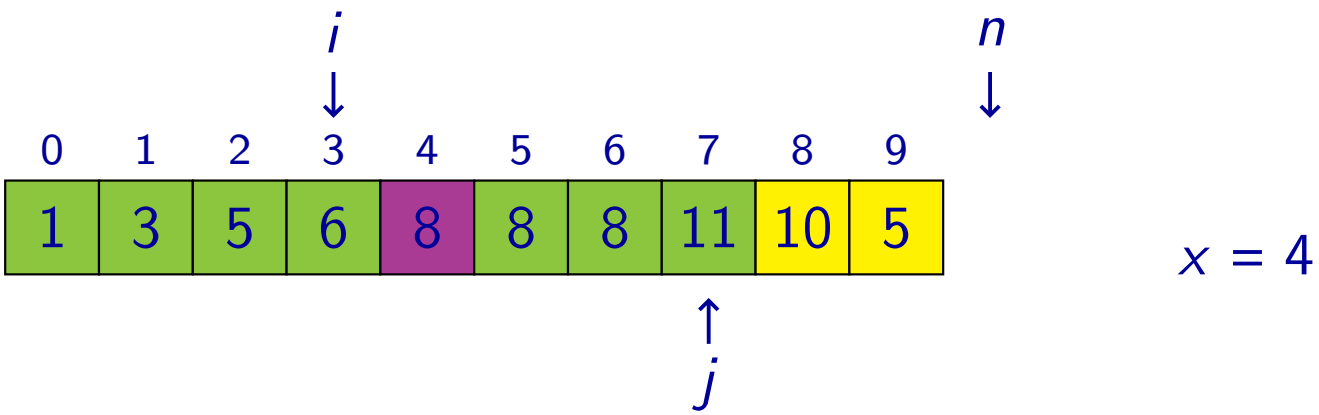
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

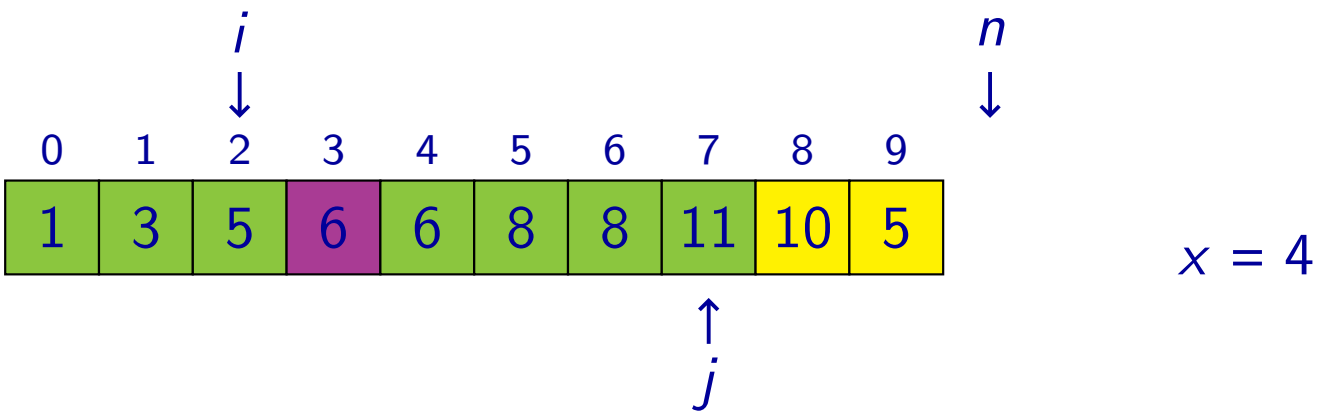
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

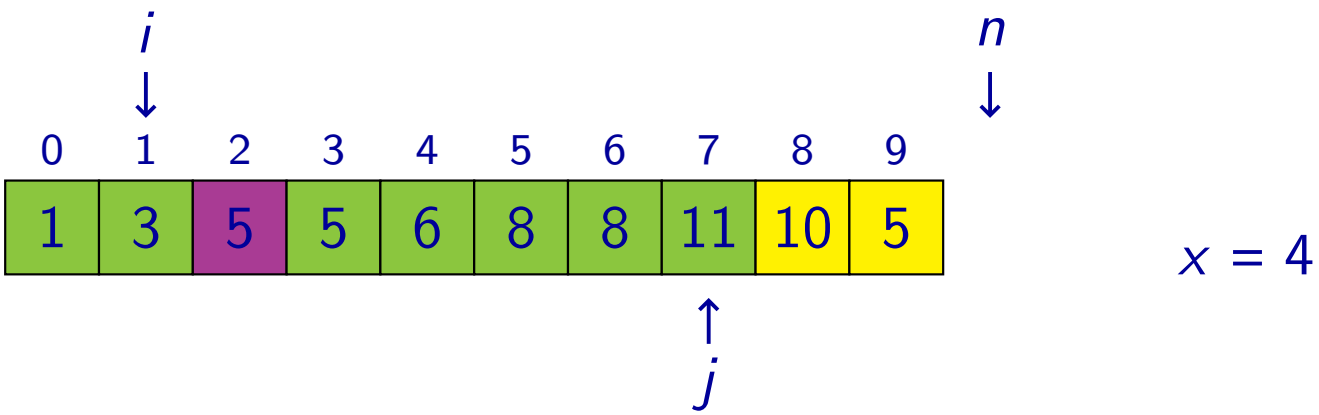
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

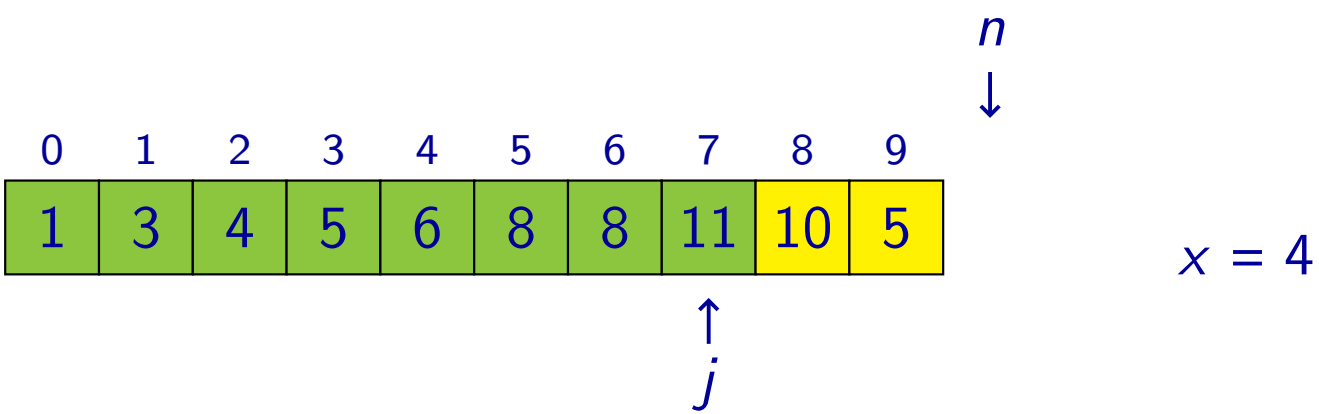
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

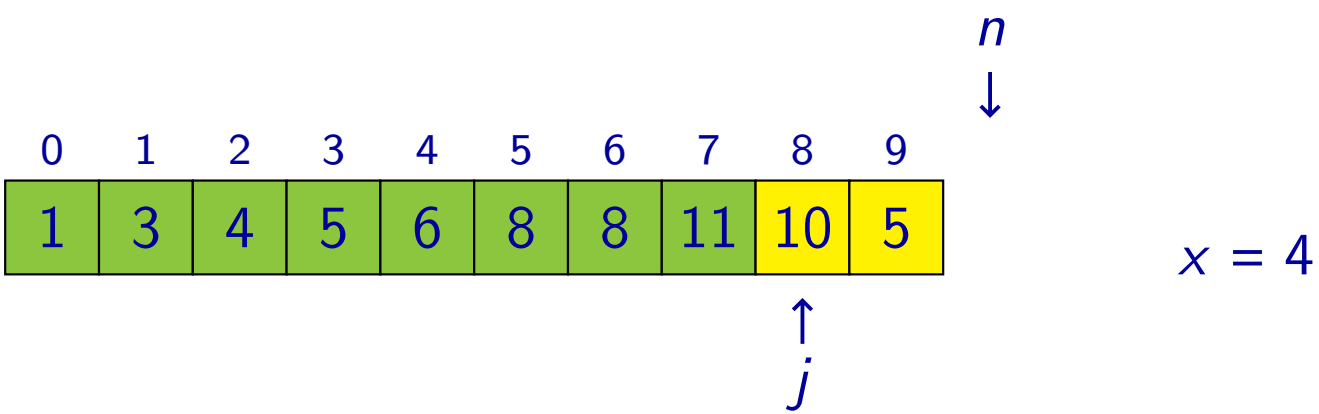
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

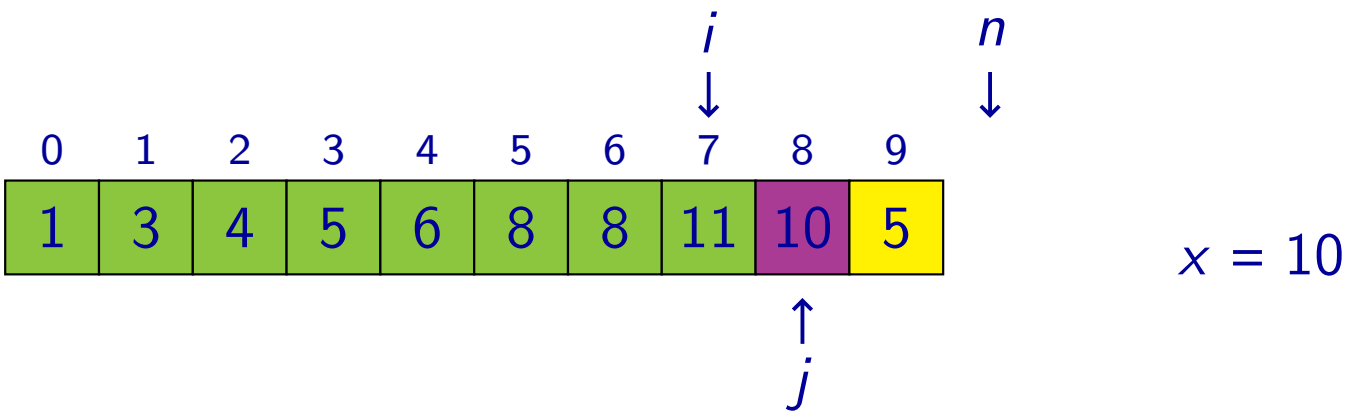
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

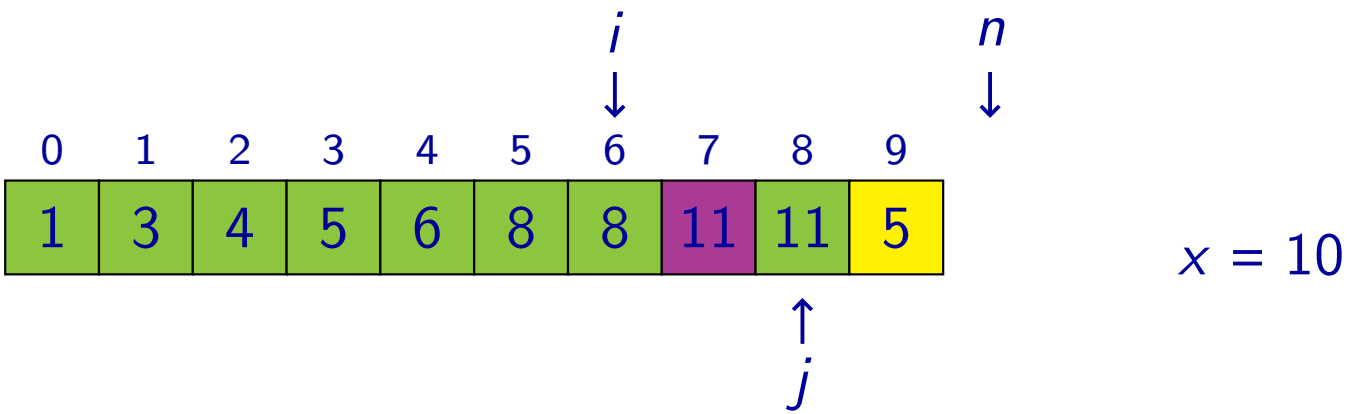
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$

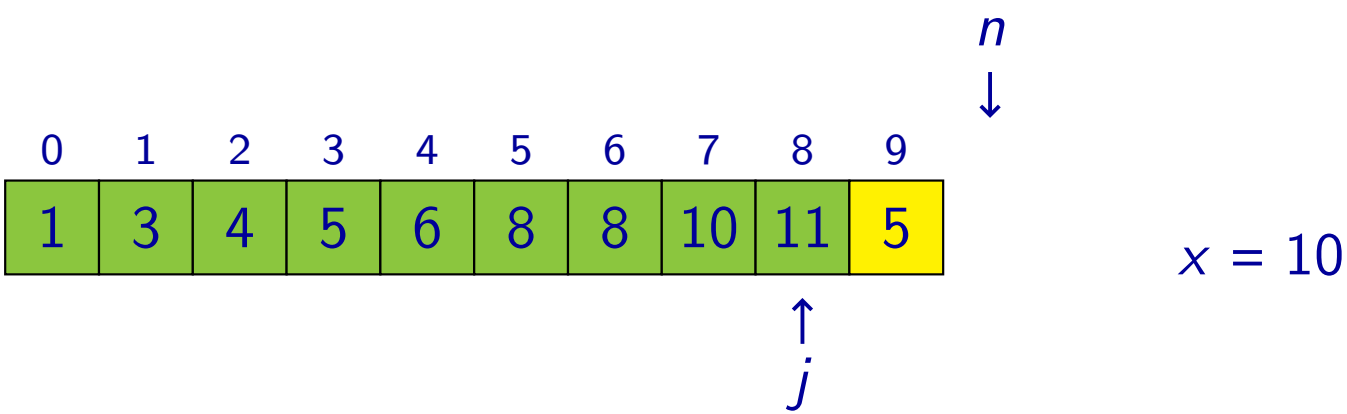


$x = 10$

Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

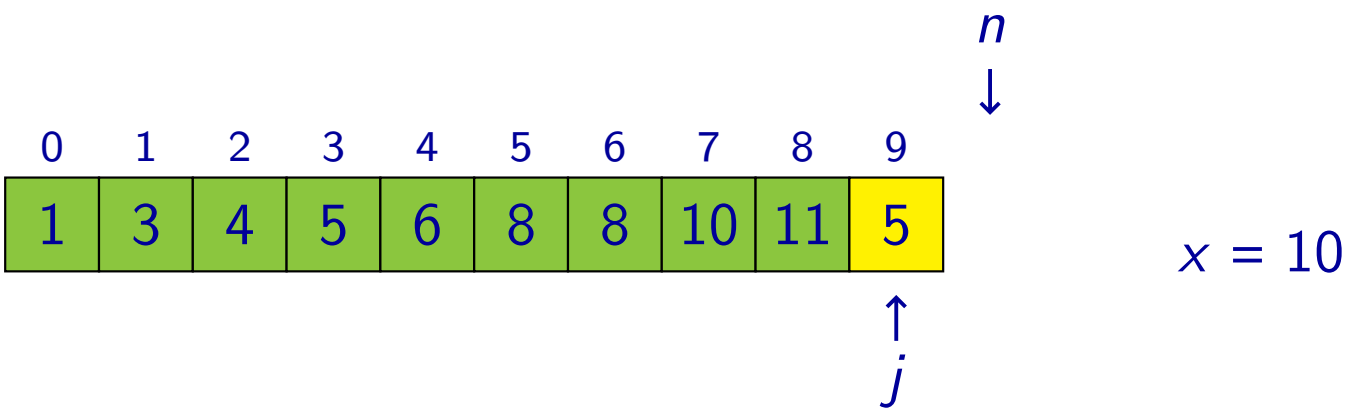
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

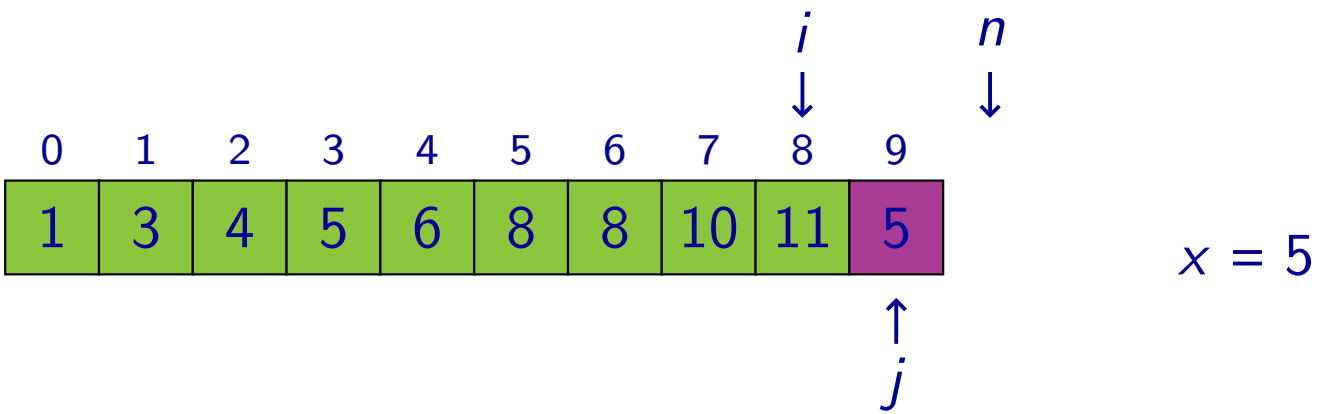
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

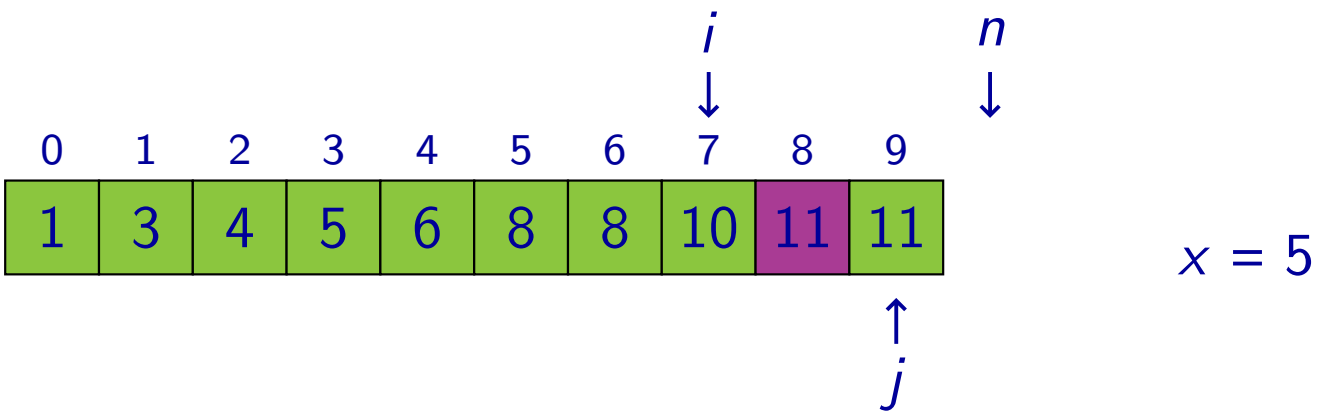
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

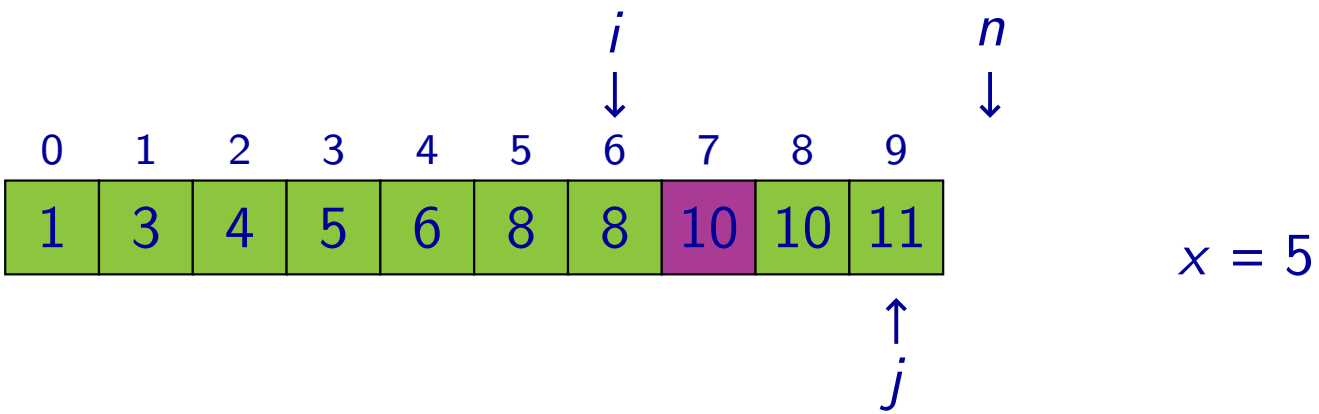
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

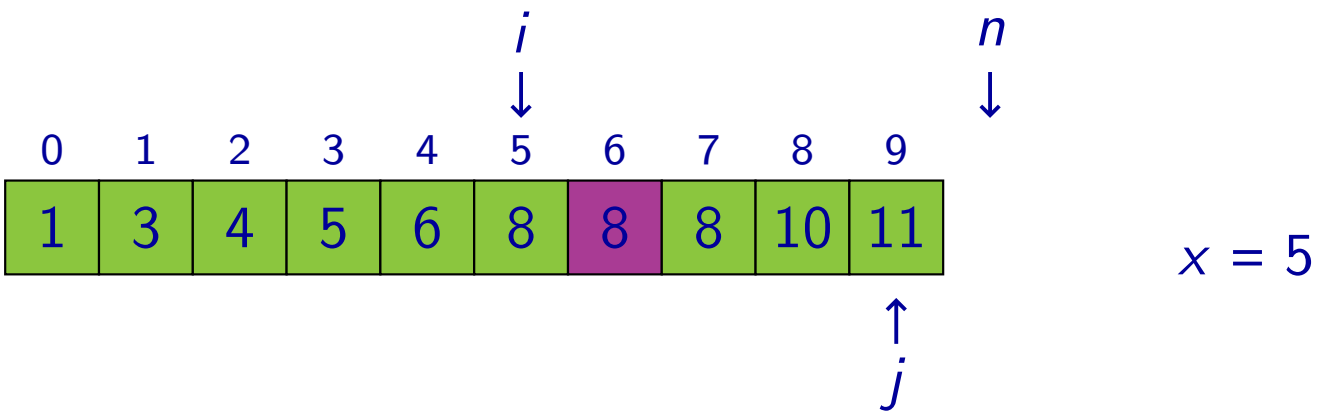
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

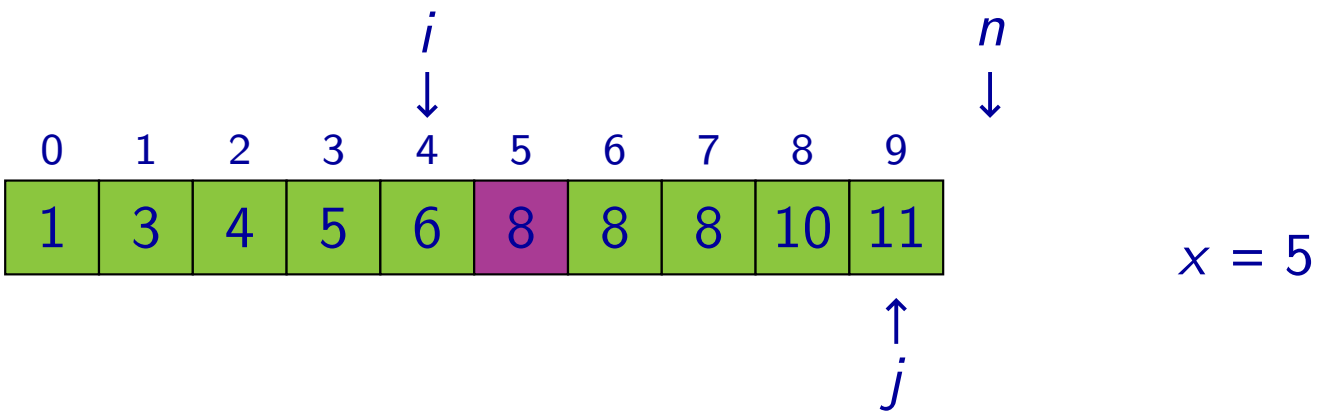
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

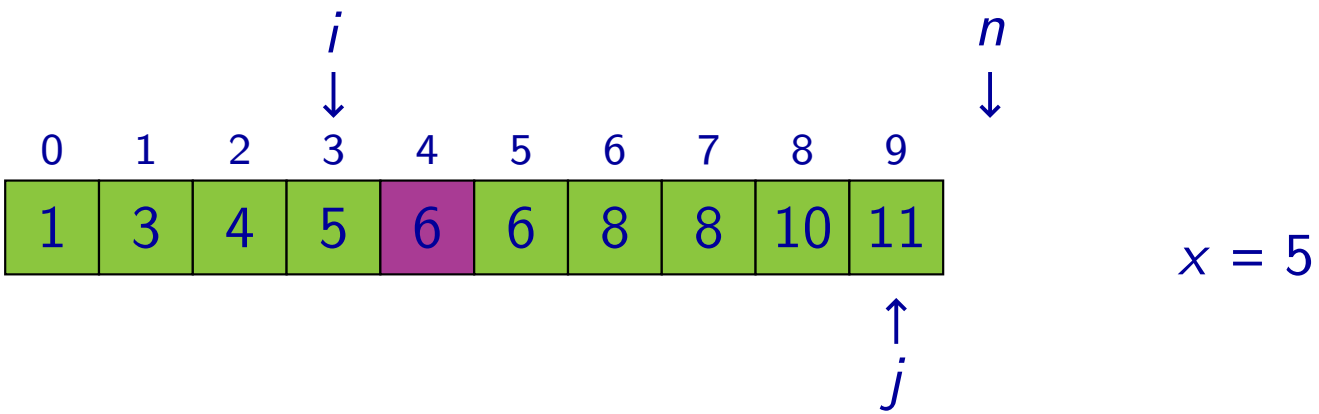
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

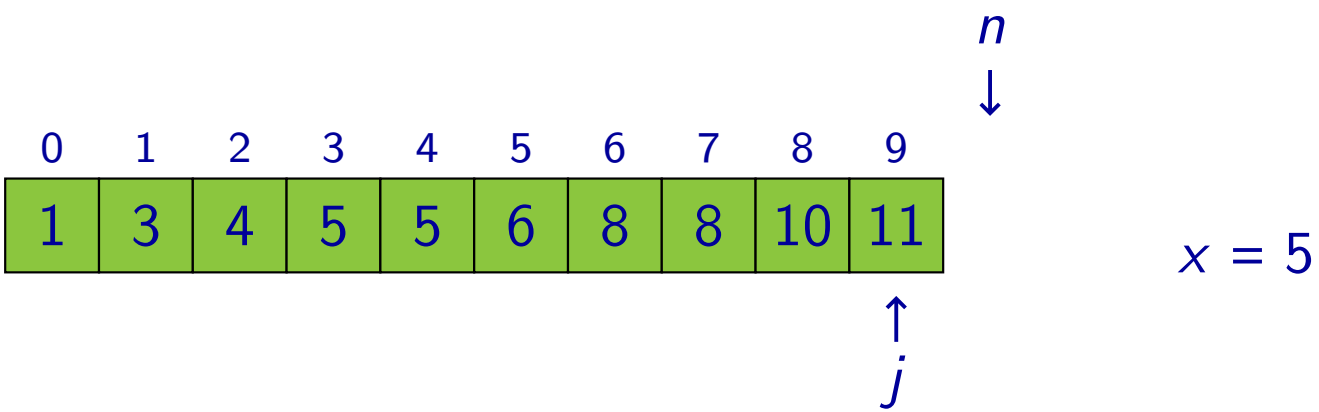
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

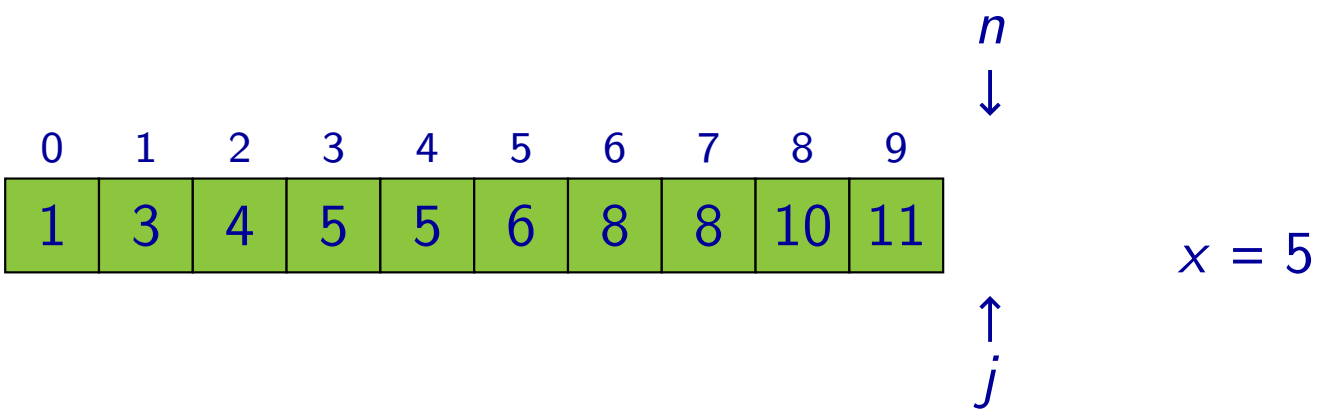
$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Příklad: Výpočet algoritmu INSERTION-SORT pro vstup

$$A = [\, 3, 8, 1, 5, 8, 6, 11, 4, 10, 5 \,],\ n = 10.$$



Invarianty

Předpokládejme, že vstupem je pole $A = [a_0, a_1, \ldots, a_{n-1}]$ a číslo n (kde $n \geq 1$) udávající délku tohoto pole, tj. že na začátku pro každé i, kde $0 \leq i < n$, platí $A[i] = a_i$.

- Na začátku cyklu **for** (tj. vždy před provedením testu $j < n$, resp. $j \leq n - 1$) platí následující invarianty:
  - $1 \leq j \leq n$
  - Prvky pole $A[0], A[1], \ldots, A[j-1]$ obsahují hodnoty $a_0, a_1, \ldots, a_{j-1}$ seřazené od nejmenší po největší, tj.

    $$A[0] \leq A[1] \leq \cdots \leq A[j-1]$$

  - Prvky pole $A[j], A[j+1], \ldots, A[n-1]$ obsahují hodnoty $a_j, a_{j+1}, \ldots, a_{n-1}$, tj.

    $$A[j] = a_j,\ A[j+1] = a_{j+1},\ \ldots,\ A[n-1] = a_{n-1}$$

Invarianty

- Na začátku cyklu **while** (tj. vždy před provedením testu $i \geq 0$) platí následující invarianty:
  - $1 \leq j < n$
  - $-1 \leq i < j$
  - Proměnná x obsahuje hodnotu a_j, tj. $x = a_j$.
  - Prvky pole $A[0], A[1], \ldots, A[i]$ a $A[i+2], A[i+3], \ldots, A[j]$ obsahují hodnoty $a_0, a_1, \ldots, a_{j-1}$ seřazené od nejmenší po největší, tj.
    $$A[0] \leq A[1] \leq \cdots \leq A[i] \leq A[i+2] \leq A[i+3] \leq \cdots \leq A[j]$$
  - Všechny prvky $A[i+2], A[i+3], \ldots, A[j]$ jsou ostře větší než x.
  - Prvky pole $A[j+1], A[j+2], \ldots, A[n-1]$ obsahují hodnoty $a_{j+1}, a_{j+2}, \ldots, a_{n-1}$, tj.
    $$A[j+1] = a_{j+1},\ A[j+2] = a_{j+2},\ \ldots,\ A[n-1] = a_{n-1}$$

Konečnost výpočtu

Dva možné případy, jak může vypadat nekonečný výpočet:

- nějaká konfigurace se zopakuje — následující konfigurace se opakují stále dokola
- objevují se stále nové a nové konfigurace

Konečnost výpočtu

Jeden z běžných způsobů dokazování toho, že se algoritmus zaručeně pro každý vstup po konečném počtu kroků zastaví:

- každé (dosažitelné) konfiguraci přiřadit hodnotu z nějaké vhodně zvolené množiny W
- na množině W definovat uspořádání $\leq$ takové, že ve W neexistují nekonečné (ostře) klesající posloupnosti
- ukázat, že s provedením každé instrukce se hodnota přiřazená konfiguraci zmenšuje, tj. pro $\alpha \xrightarrow{I} \alpha'$ je

$$f(\alpha) > f(\alpha')$$

($f(\alpha)$, $f(\alpha')$ jsou hodnoty z množiny W přiřazené konfiguracím α a α')

Konečnost výpočtu

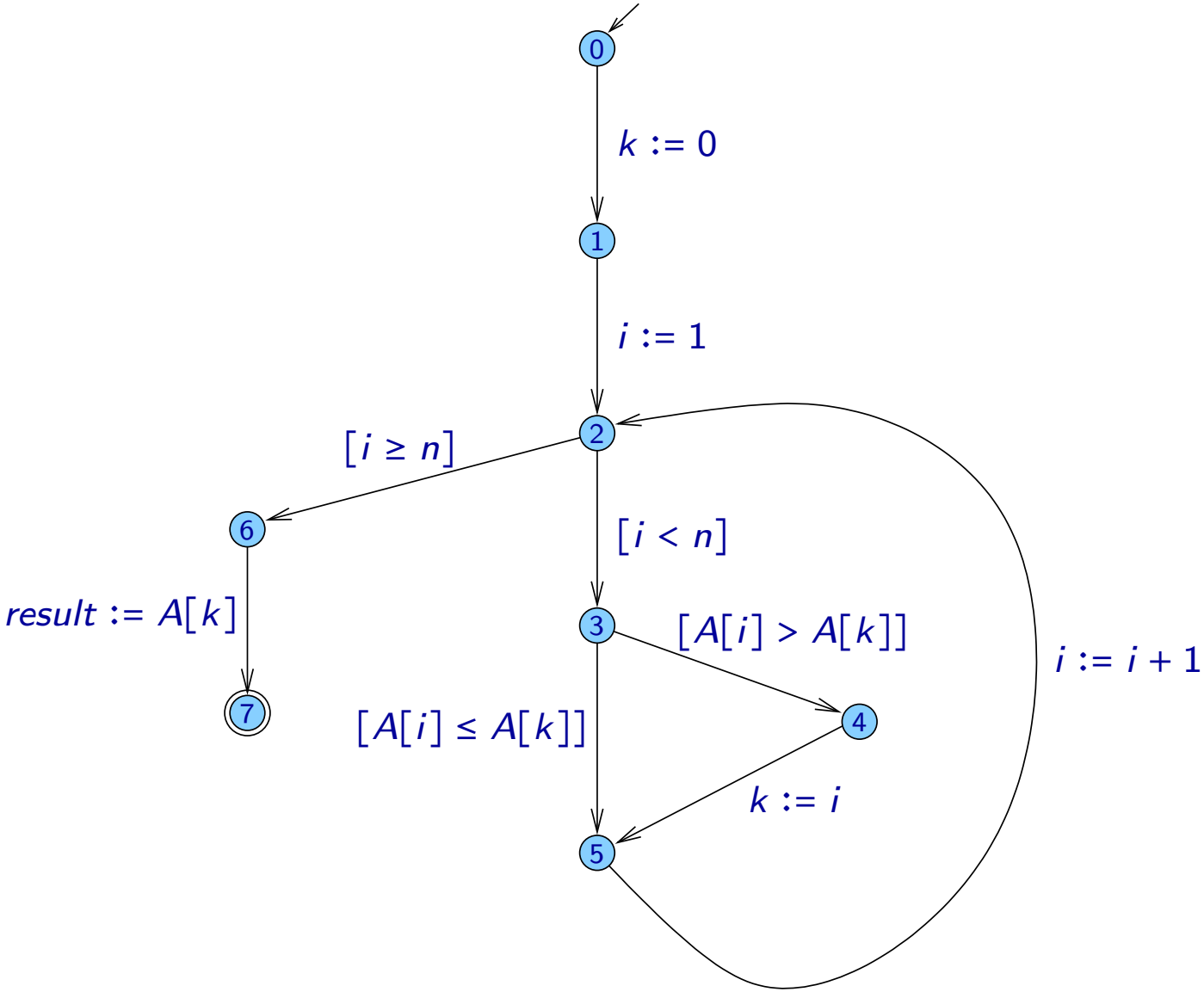
Jako množinu W je možno použít například:

- Množinu přirozených čísel $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ s uspořádáním $\leq$.
- Množinu vektorů přirozených čísel s lexikografickým uspořádáním, tj. s uspořádáním, kde vektor $(a_1, a_2, \dots, a_m)$ je menší než vektor $(b_1, b_2, \dots, b_n)$, jestliže
  - existuje i takové, že $1 \leq i \leq m$ a $i \leq n$, kde $a_i < b_i$ a pro všechna j taková, že $1 \leq j < i$, platí $a_j = b_j$, nebo
  - $m < n$ a pro všechna j taková, že $1 \leq j \leq m$, je $a_j = b_j$.

  Například $(5, 1, 3, 6, 4) < (5, 1, 4, 1)$ a $(4, 1, 1) < (4, 1, 1, 3)$.

  Poznámka: Počet prvků vektorů musí být omezen nějakou konstantou.

Konečnost výpočtu

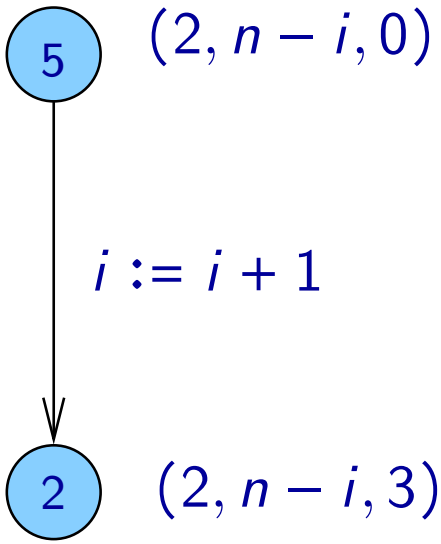


Konečnost výpočtu

Příklad: Vektory přiřazené jednotlivým konfiguracím:

- Stav 0: $f(\alpha) = (4)$
- Stav 1: $f(\alpha) = (3)$
- Stav 2: $f(\alpha) = (2, n - i, 3)$
- Stav 3: $f(\alpha) = (2, n - i, 2)$
- Stav 4: $f(\alpha) = (2, n - i, 1)$
- Stav 5: $f(\alpha) = (2, n - i, 0)$
- Stav 6: $f(\alpha) = (1)$
- Stav 7: $f(\alpha) = (0)$

Konečnost výpočtu



Je třeba brát v úvahu, že se touto instrukcí hodnota proměnné i mění.

Z konfigurace s přiřazeným vektorem $(2, n-i, 0)$ se přejde do konfigurace s přiřazeným vektorem $(2, n-i', 3)$, kde $i' = i + 1$.

Zjevně platí $n - i' < n - i$, neboť $n - (i + 1) < n - i$.