KATEDRA
INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

Paradigmata programování 1

4 Rekurzivní výpočetní proces

Michal Krupka

Obsah

Funkce `power`

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Rekurzivní funkce

- ukončovací podmínka: případy, kdy není třeba použít rekurzi
- rekurzivní aplikace: použití téže funkce pro jednodušší případ (bližší k ukončovací podmínce)

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Rekurzivní funkce

- ukončovací podmínka: případy, kdy není třeba použít rekurzi
- rekurzivní aplikace: použití téže funkce pro jednodušší případ (bližší k ukončovací podmínce)

Deklarativní přístup. Nezapomínejte také, že nad rekurzivním výpočtem je vhodné uvažovat deklarativně. Neuvažovat, jak k výsledku dojít, ale jaké má mít vlastnosti.

Obsah

Výpočetní proces

Výpočetní proces

je činnost, kterou vykonává počítač na základě nějakého programu nebo jeho části (funkce).
Výpočetní proces *generovaný* funkcí je výpočetní proces vykonávaný během její aplikace.

Rekurzivní výpočetní proces

Rekurzivní výpočetní proces

Výpočetní proces je *rekurzivní*, když v něm během aplikace funkce dochází znovu k aplikaci téže funkce.
(Obecněji: během vykonávání části programu dochází k vykonávání téže části programu.)

Rekurzivní výpočetní proces

Rekurzivní výpočetní proces

Výpočetní proces je *rekurzivní*, když v něm během aplikace funkce dochází znovu k aplikaci téže funkce.
(Obecněji: během vykonávání části programu dochází k vykonávání téže části programu.)

- je poznat, když program běží
- některá aplikace funkce by k aplikaci téže funkce vést neměla (*ukončovací podmínka*)

Rekurzivní výpočetní proces

Rekurzivní výpočetní proces

Výpočetní proces je *rekurzivní*, když v něm během aplikace funkce dochází znovu k aplikaci téže funkce.
(Obecněji: během vykonávání části programu dochází k vykonávání téže části programu.)

- je poznat, když program běží
- některá aplikace funkce by k aplikaci téže funkce vést neměla (*ukončovací podmínka*)

Rekurzivní aplikace funkce

Aplikace funkce, ke které dojde během aplikace téže funkce.

Rekurzivní výpočetní proces

Rekurzivní výpočetní proces

Výpočetní proces je *rekurzivní*, když v něm během aplikace funkce dochází znovu k aplikaci téže funkce.
(Obecněji: během vykonávání části programu dochází k vykonávání téže části programu.)

- je poznat, když program běží
- některá aplikace funkce by k aplikaci téže funkce vést neměla (*ukončovací podmínka*)

Rekurzivní aplikace funkce

Aplikace funkce, ke které dojde během aplikace téže funkce.

Iterativní výpočetní proces

Výpočetní proces je *iterativní*, když v něm po aplikaci funkce dochází opět k aplikaci téže funkce.
(Obecněji: část programu se vykonává opakovaně.)

Rekurzivní výpočetní proces

Rekurzivní výpočetní proces

Výpočetní proces je *rekurzivní*, když v něm během aplikace funkce dochází znovu k aplikaci téže funkce.
(Obecněji: během vykonávání části programu dochází k vykonávání téže části programu.)

- je poznat, když program běží
- některá aplikace funkce by k aplikaci téže funkce vést neměla (*ukončovací podmínka*)

Rekurzivní aplikace funkce

Aplikace funkce, ke které dojde během aplikace téže funkce.

Iterativní výpočetní proces

Výpočetní proces je *iterativní*, když v něm po aplikaci funkce dochází opět k aplikaci téže funkce.
(Obecněji: část programu se vykonává opakovaně.)

- bývá generován pomocí cyklů
- koncově rekurzivní funkce mohou generovat iterativní výpočetní proces

Funkce `power`

aplikace `(power 10 4)`
vyvolá aplikaci `(power 10 3)`
  aplikace `(power 10 3)`
  vyvolá aplikaci `(power 10 2)`
    aplikace `(power 10 2)`
    vyvolá aplikaci `(power 10 1)`
      aplikace `(power 10 1)`
      vyvolá aplikaci `(power 10 0)`
        aplikace `(power 10 0)`
        vrátí `1`
      pak vyvolá aplikaci `(* 10 1)`
      vrátí `10`
    pak vyvolá aplikaci `(* 10 10)`
    vrátí `100`
  pak vyvolá aplikaci `(* 10 100)`
  vrátí `1000`
pak vyvolá aplikaci `(* 10 1000)`
vrátí `10000`

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
  aplikace (power 10 3)
  vyvolá aplikaci (power 10 2)
    aplikace (power 10 2)
    vyvolá aplikaci (power 10 1)
      aplikace (power 10 1)
      vyvolá aplikaci (power 10 0)
        aplikace (power 10 0)
        vrátí 1
      pak vyvolá aplikaci (* 10 1)
      vrátí 10
    pak vyvolá aplikaci (* 10 10)
    vrátí 100
  pak vyvolá aplikaci (* 10 100)
  vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce `power`

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce `power`

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power`

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power`

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce power

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce power

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power`

aplikace (power 10 4)
vyvolá aplikaci (power 10 3)
aplikace (power 10 3)
vyvolá aplikaci (power 10 2)
aplikace (power 10 2)
vyvolá aplikaci (power 10 1)
aplikace (power 10 1)
vyvolá aplikaci (power 10 0)
aplikace (power 10 0)
vrátí 1
pak vyvolá aplikaci (* 10 1)
vrátí 10
pak vyvolá aplikaci (* 10 10)
vrátí 100
pak vyvolá aplikaci (* 10 100)
vrátí 1000
pak vyvolá aplikaci (* 10 1000)
vrátí 10000

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power`

aplikace `(power 10 4)`
vyvolá aplikaci `(power 10 3)`
aplikace `(power 10 3)`
vyvolá aplikaci `(power 10 2)`
aplikace `(power 10 2)`
vyvolá aplikaci `(power 10 1)`
aplikace `(power 10 1)`
vyvolá aplikaci `(power 10 0)`
aplikace `(power 10 0)`
vrátí `1`
pak vyvolá aplikaci `(* 10 1)`
vrátí `10`
pak vyvolá aplikaci `(* 10 10)`
vrátí `100`
pak vyvolá aplikaci `(* 10 100)`
vrátí `1000`
pak vyvolá aplikaci `(* 10 1000)`
vrátí `10000`

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power`

aplikace `(power 10 4)`
vyvolá aplikaci `(power 10 3)`
aplikace `(power 10 3)`
vyvolá aplikaci `(power 10 2)`
aplikace `(power 10 2)`
vyvolá aplikaci `(power 10 1)`
aplikace `(power 10 1)`
vyvolá aplikaci `(power 10 0)`
aplikace `(power 10 0)`
vrátí 1
pak vyvolá aplikaci `(* 10 1)`
vrátí 10
pak vyvolá aplikaci `(* 10 10)`
vrátí 100
pak vyvolá aplikaci `(* 10 100)`
vrátí 1000
pak vyvolá aplikaci `(* 10 1000)`
vrátí 10000

Funkce `power`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Prostředí při aplikaci `(power 10 4)`

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

Prostředí při aplikaci `(power 10 4)`

Globální prostředí

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

Prostředí při aplikaci `(power 10 4)`

Globální prostředí

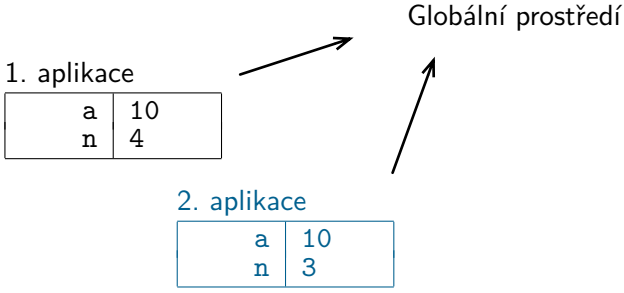
1. aplikace

a	10
n	4

```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

Prostředí při aplikaci `(power 10 4)`

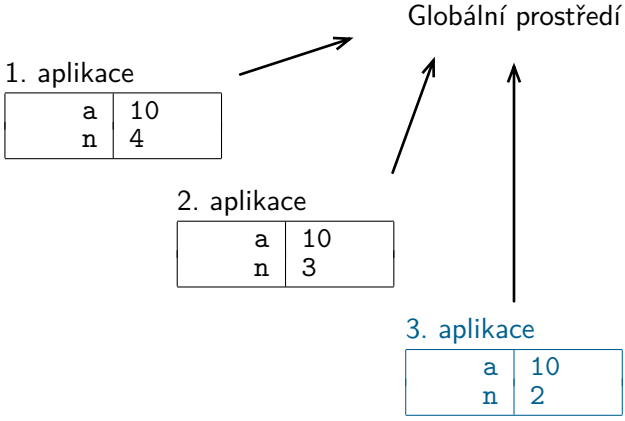



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci (power 10 4)

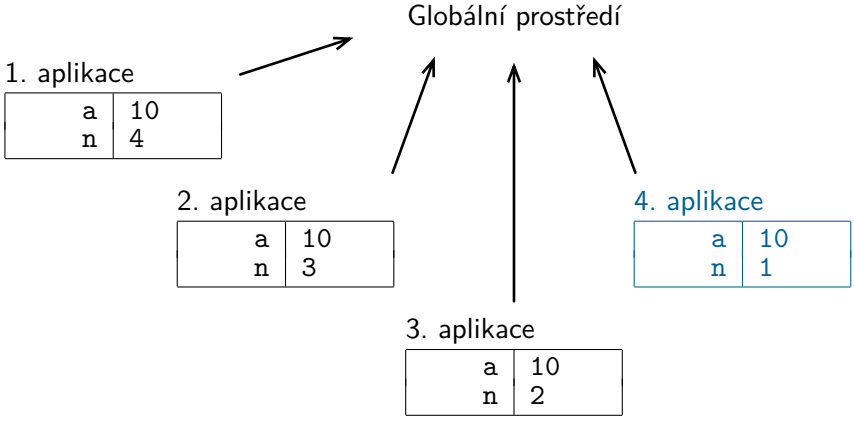



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci (power 10 4)

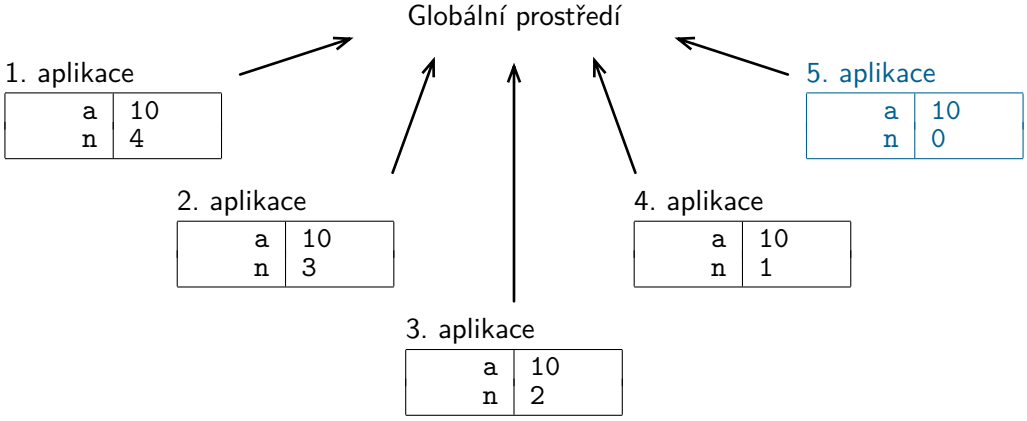



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci (power 10 4)

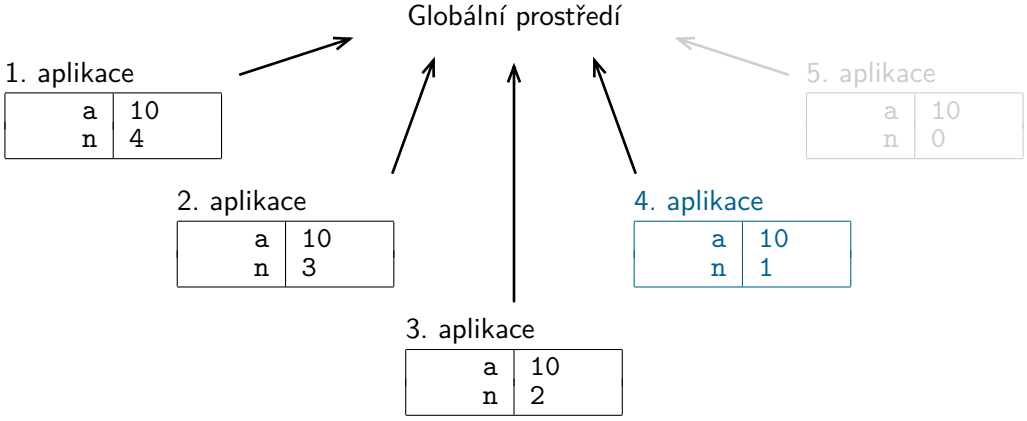



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci (power 10 4)

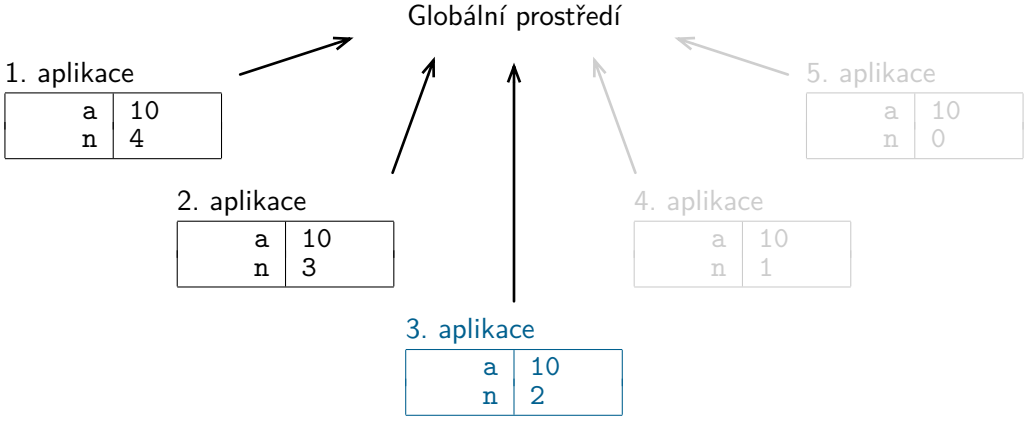



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci (power 10 4)

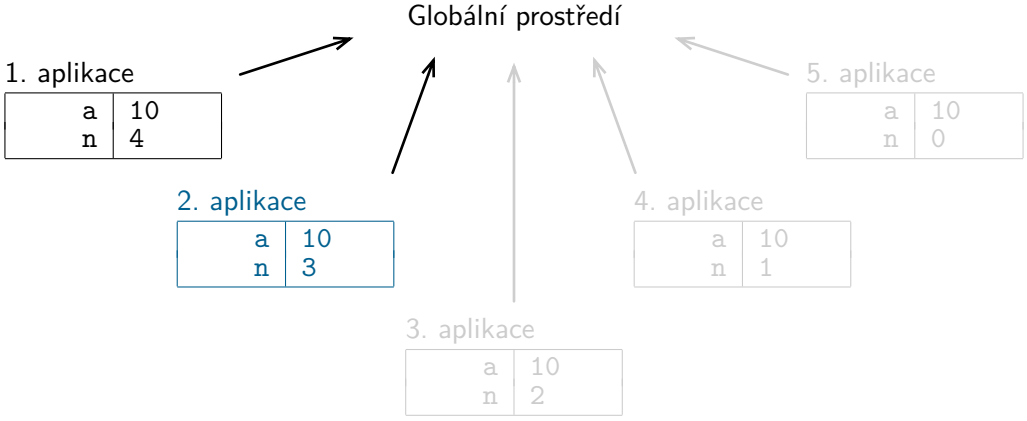



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci (power 10 4)

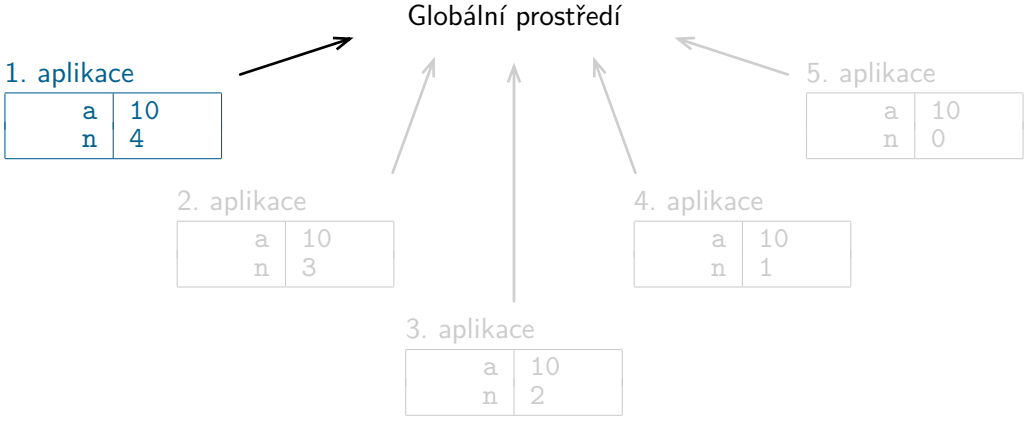



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci (power 10 4)

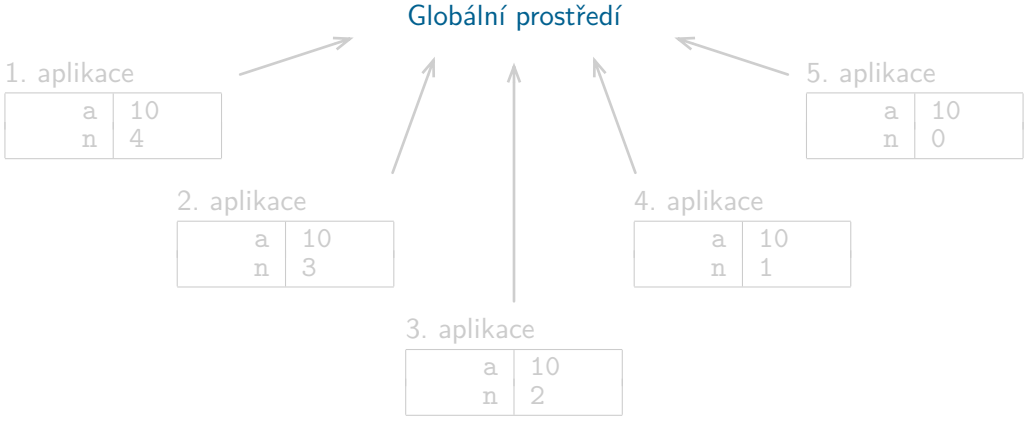



```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci `(power 10 4)`




```
(defun power (a n)
  (if (= n 0)
      1
    (* a (power a (- n 1)))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Verze funkce `power` s koncovou rekurzí (iterativní)

Verze funkce `power` s koncovou rekurzí (iterativní)

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))

(defun power (a n)
  (power-iter a n 1))
```

Funkce `power-iter`

aplikace `(power-iter 10 4 1)`
vyvolá aplikaci `(power-iter 10 3 10)`
aplikace `(power-iter 10 3 10)`
vyvolá aplikaci `(power-iter 10 2 100)`
aplikace `(power-iter 10 2 100)`
vyvolá aplikaci `(power-iter 10 1 1000)`
aplikace `(power-iter 10 1 1000)`
vyvolá aplikaci `(power-iter 10 0 10000)`
aplikace `(power-iter 10 0 10000)`
vrátí `10000`
vrátí `10000`
vrátí `10000`
vrátí `10000`
vrátí `10000`

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Funkce power-iter

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce power-iter

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power-iter`

aplikace (power-iter 10 4 1)
vyvolá aplikaci (power-iter 10 3 10)
aplikace (power-iter 10 3 10)
vyvolá aplikaci (power-iter 10 2 100)
aplikace (power-iter 10 2 100)
vyvolá aplikaci (power-iter 10 1 1000)
aplikace (power-iter 10 1 1000)
vyvolá aplikaci (power-iter 10 0 10000)
aplikace (power-iter 10 0 10000)
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Funkce `power-iter`

aplikace `(power-iter 10 4 1)`
vyvolá aplikaci `(power-iter 10 3 10)`
aplikace `(power-iter 10 3 10)`
vyvolá aplikaci `(power-iter 10 2 100)`
aplikace `(power-iter 10 2 100)`
vyvolá aplikaci `(power-iter 10 1 1000)`
aplikace `(power-iter 10 1 1000)`
vyvolá aplikaci `(power-iter 10 0 10000)`
aplikace `(power-iter 10 0 10000)`
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000
vrátí 10000

Funkce `power-iter`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a
                (- n 1)
                (* a ir))))
```

Začátek aplikace
Konec aplikace (vrácená hodnota)

Prostředí při aplikaci `(power-iter 10 4 1)`

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

Prostředí při aplikaci `(power-iter 10 4 1)`

Globální prostředí

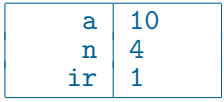
```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

aktuální prostředí

Prostředí při aplikaci `(power-iter 10 4 1)`

Globální prostředí

1. aplikace

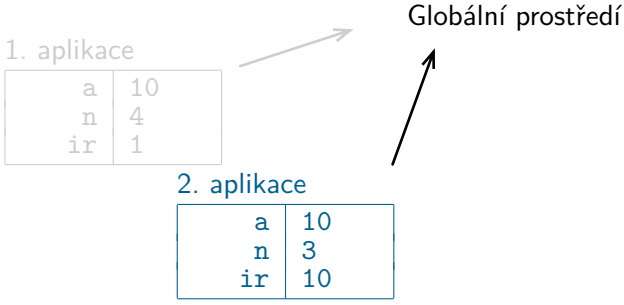


a	10
n	4
ir	1

```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

aktuální prostředí

Prostředí při aplikaci `(power-iter 10 4 1)`

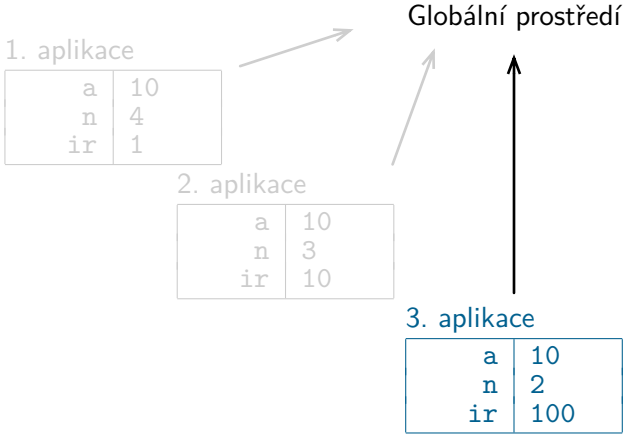



```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci `(power-iter 10 4 1)`

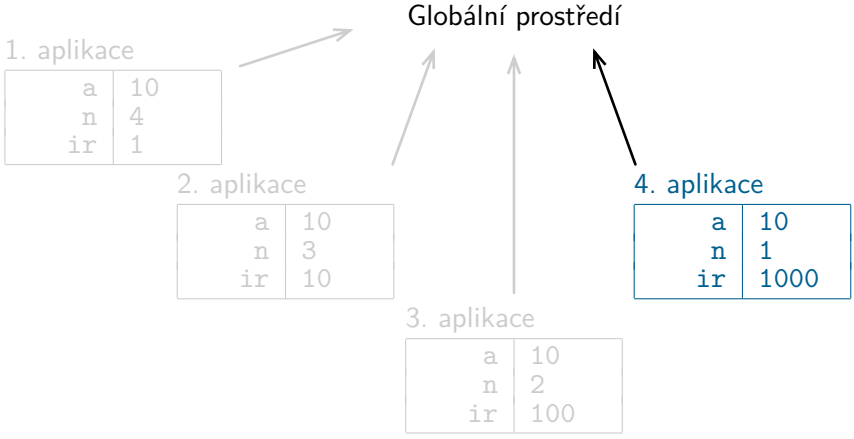



```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci `(power-iter 10 4 1)`

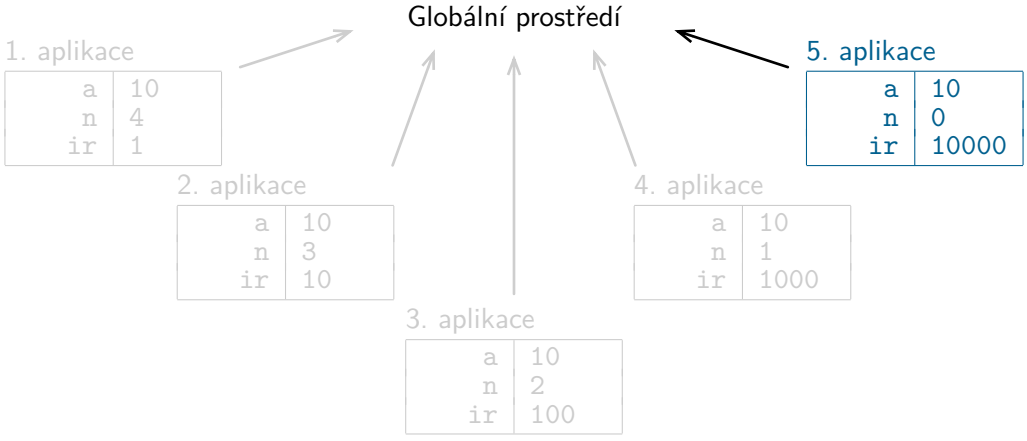



```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci `(power-iter 10 4 1)`

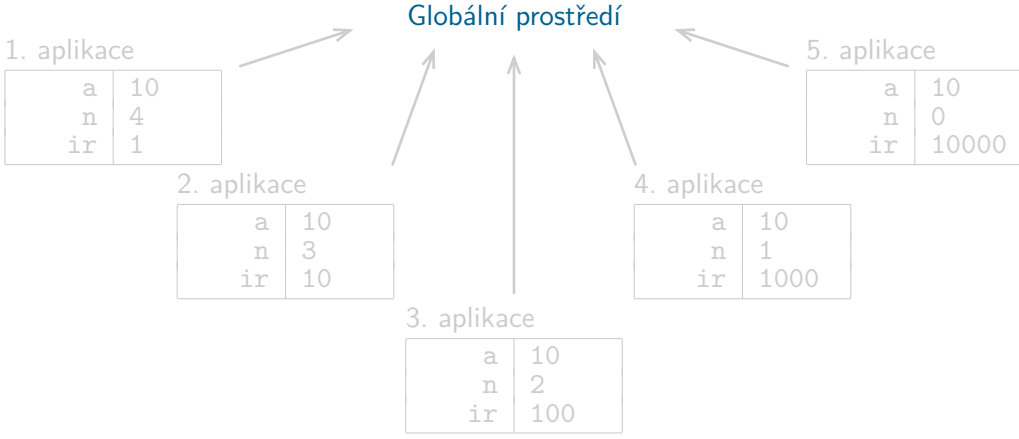



```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Prostředí při aplikaci `(power-iter 10 4 1)`




```
(defun power-iter (a n ir)
  (if (= n 0)
      ir
    (power-iter a (- n 1) (* a ir))))
```

aktuální prostředí

prostředí už nebude nikdy potřeba

Různé způsoby výpočtu a^9

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$a^9 = a(a^8)$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$\begin{aligned} a^9 &= a(a^8) \\ &= a(a(a^7)) \end{aligned}$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$\begin{aligned} a^9 &= a(a^8) \\ &= a(a(a^7)) \\ &= a(a(a(a^6))) \end{aligned}$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$\begin{aligned} a^9 &= a(a^8) \\ &= a(a(a^7)) \\ &= a(a(a(a^6))) \\ &= a(a(a(a(a^5)))) \end{aligned}$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3))))))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a \cdot a))))))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a \cdot a)))))) \\
&= a(a(a(a(a(a \cdot a^2))))))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a \cdot a)))))) \\
&= a(a(a(a(a(a \cdot a^2))))) \\
&= a(a(a(a(a \cdot a^3))))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a \cdot a)))))) \\
&= a(a(a(a(a(a \cdot a^2)))))) \\
&= a(a(a(a(a \cdot a^3)))) \\
&= a(a(a(a \cdot a^4)))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a\cdot a)))))) \\
&= a(a(a(a(a(a\cdot a^2))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a\cdot a)))))) \\
&= a(a(a(a(a(a\cdot a^2))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6))
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a \cdot a)))))) \\
&= a(a(a(a(a(a \cdot a^2)))))) \\
&= a(a(a(a(a \cdot a^3)))) \\
&= a(a(a(a \cdot a^4))) \\
&= a(a(a \cdot a^5)) \\
&= a(a(a \cdot a^6)) \\
&= a(a \cdot a^7)
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a\cdot a)))))) \\
&= a(a(a(a(a(a\cdot a^2))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a\cdot a)))))) \\
&= a(a(a(a(a(a\cdot a^2))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno) Iterativně

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a \cdot a))))))) \\
&= a(a(a(a(a(a(a \cdot a^2)))))) \\
&= a(a(a(a(a(a \cdot a^3))))) \\
&= a(a(a(a(a \cdot a^4)))) \\
&= a(a(a(a \cdot a^5))) \\
&= a(a(a \cdot a^6)) \\
&= a(a \cdot a^7) \\
&= a \cdot a^8 \\
&= a^9
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a \cdot a))))))) \\
&= a(a(a(a(a(a(a \cdot a^2)))))) \\
&= a(a(a(a(a(a \cdot a^3))))) \\
&= a(a(a(a(a \cdot a^4)))) \\
&= a(a(a(a \cdot a^5))) \\
&= a(a(a \cdot a^6)) \\
&= a(a \cdot a^7) \\
&= a \cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$aaaaaaaaa$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a(a\cdot a^4)))) \\
&= a(a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$aaaaaaaaa$

$(a\cdot a)(aaaaaaa)$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a(a\cdot a^4)))) \\
&= a(a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa)
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a(a\cdot a^4)))) \\
&= a(a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa)
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a(a\cdot a^4)))) \\
&= a(a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$aaaaaaaaa$

$(a\cdot a)(aaaaaaa)$

$(a^2\cdot a)(aaaaaa)$

$(a^3\cdot a)(aaaaa)$

$(a^4\cdot a)(aaaa)$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa)
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a\cdot a^4)))) \\
&= a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa)
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a\cdot a^4)))) \\
&= a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a\cdot a^4)))) \\
&= a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$aaaaaaaaa$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$
\begin{aligned}
&aaaaaaaaa \\
&a(aaaaaaaa)
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$
\begin{aligned}
&aaaaaaaaa \\
&a(aaaaaaaa) \\
&a(a\cdot a)^4
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$
\begin{aligned}
&aaaaaaaaa \\
&a(aaaaaaaa) \\
&a(a\cdot a)^4 \\
&a(a^2\cdot a^2)^2
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$
\begin{aligned}
&aaaaaaaaa \\
&a(aaaaaaaa) \\
&a(a\cdot a)^4 \\
&a(a^2\cdot a^2)^2 \\
&a(a^4\cdot a^4)
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$
\begin{aligned}
&aaaaaaaaa \\
&a(aaaaaaaa) \\
&a(a\cdot a)^4 \\
&a(a^2\cdot a^2)^2 \\
&a(a^4\cdot a^4) \\
&a\cdot a^8
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a\cdot a^3)))) \\
&= a(a(a(a\cdot a^4))) \\
&= a(a(a\cdot a^5)) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$
\begin{aligned}
&aaaaaaaaa \\
&a(aaaaaaaa) \\
&a(a\cdot a)^4 \\
&a(a^2\cdot a^2)^2 \\
&a(a^4\cdot a^4) \\
&a\cdot a^8 \\
&a^9
\end{aligned}
$$

Různé způsoby výpočtu a^9

Rekurzivně ($a^0 = 1$ vynecháno)

$$
\begin{aligned}
a^9 &= a(a^8) \\
&= a(a(a^7)) \\
&= a(a(a(a^6))) \\
&= a(a(a(a(a^5)))) \\
&= a(a(a(a(a(a^4))))) \\
&= a(a(a(a(a(a(a^3)))))) \\
&= a(a(a(a(a(a(a(a^2))))))) \\
&= a(a(a(a(a(a(a(a(a)))))))) \\
&= a(a(a(a(a(a(a(a\cdot a))))))) \\
&= a(a(a(a(a(a(a\cdot a^2)))))) \\
&= a(a(a(a(a(a\cdot a^3))))) \\
&= a(a(a(a(a\cdot a^4)))) \\
&= a(a(a(a\cdot a^5))) \\
&= a(a(a\cdot a^6)) \\
&= a(a\cdot a^7) \\
&= a\cdot a^8 \\
&= a^9
\end{aligned}
$$

Iterativně

$$
\begin{aligned}
&aaaaaaaaa \\
&(a\cdot a)(aaaaaaa) \\
&(a^2\cdot a)(aaaaaa) \\
&(a^3\cdot a)(aaaaa) \\
&(a^4\cdot a)(aaaa) \\
&(a^5\cdot a)(aaa) \\
&(a^6\cdot a)(aa) \\
&(a^7\cdot a)a \\
&a^8\cdot a \\
&a^9
\end{aligned}
$$

Zrychleně

$$
\begin{aligned}
&aaaaaaaaa \\
&a(aaaaaaaa) \\
&a(a\cdot a)^4 \\
&a(a^2\cdot a^2)^2 \\
&a(a^4\cdot a^4) \\
&a\cdot a^8 \\
&a^9
\end{aligned}
$$

Implementace zrychlené mocniny

```
(defun power2 (n)
  (* n n))

(defun fast-power (a n)
  (cond ((= n 0) 1)
        ((evenp n)
         (power2 (fast-power a (/ n 2))))
        (t (* a (fast-power a (- n 1))))))
```

Rychlost výpočtu mocniny

Počet násobení při výpočtu a^n:
(násobení jedničkou nepočítáme)

rekurzivně: $n - 1$

iterativně: $n - 1$

zrychleně: $\lceil \log_2 n \rceil$

($\lceil x \rceil$ je číslo x zaokrouhlené nahoru.)

Rychlost výpočtu mocniny

Počet násobení při výpočtu a^n:
(násobení jedničkou nepočítáme)

rekurzivně: $n - 1$

iterativně: $n - 1$

zrychleně: $\lceil \log_2 n \rceil$

($\lceil x \rceil$ je číslo x zaokrouhlené nahoru.)

$\mathbf{n}$	$\mathbf{n-1}$	$\lceil \log_2 \mathbf{n} \rceil$
1	0	0
2	1	1
3	2	2
4	3	2
5	4	3
6	5	3
7	6	3
8	7	3
9	8	4
100	99	7
1000	999	10
10000	9999	14

Rychlost výpočtu mocniny

Počet násobení při výpočtu a^n:
(násobení jedničkou nepočítáme)

rekurzivně: $n - 1$

iterativně: $n - 1$

zrychleně: $\lceil \log_2 n \rceil$

($\lceil x \rceil$ je číslo x zaokrouhlené nahoru.)

$\mathbf{n}$	$\mathbf{n-1}$	$\lceil \log_2 \mathbf{n} \rceil$
1	0	0
2	1	1
3	2	2
4	3	2
5	4	3
6	5	3
7	6	3
8	7	3
9	8	4
100	99	7
1000	999	10
10000	9999	14

Obsah

Lineárně a stromově rekurzivní výpočetní proces

Lineárně a stromově rekurzivní výpočetní proces

Lineárně rekurzivní výpočetní proces

Výpočetní proces je *lineárně rekurzivní*, když během aplikace funkce po skončení jedné její rekurzivní aplikace nenásleduje další její rekurzivní aplikace.

Lineárně a stromově rekurzivní výpočetní proces

Lineárně rekurzivní výpočetní proces

Výpočetní proces je *lineárně rekurzivní*, když během aplikace funkce po skončení jedné její rekurzivní aplikace nenásleduje další její rekurzivní aplikace.

Stromově rekurzivní výpočetní proces

Výpočetní proces je *stromově rekurzivní*, když alespoň jednou během aplikace funkce po skončení jedné její rekurzivní aplikace další její rekurzivní aplikace následuje.

Fibonacciho posloupnost

Fibonacciho posloupnost

$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$

Fibonacciho posloupnost

$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$

$$a_0 = 0$$
$$a_1 = 1$$
$$a_n = a_{n-2} + a_{n-1}$$

Fibonacciho posloupnost

$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$

$$a_0 = 0$$
$$a_1 = 1$$
$$a_n = a_{n-2} + a_{n-1}$$

```
(defun fib (n)
  (cond ((= n 0) 0)
        ((= n 1) 1)
        (t (+ (fib (- n 2)) (fib (- n 1))))))
```

Rozměňování

Rozměňování

```
(defun count-change (amount)
  (cc amount 6))
```

Rozměňování

```
(defun count-change (amount)
  (cc amount 6))

(defun cc (amount kinds)
  (cond ((= amount 0) 1)
        ((or (< amount 0) (= kinds 0)) 0)
        (t (+ (cc amount (- kinds 1))
              (cc (- amount (first-denom kinds)) kinds)))))
```

Rozměňování

```
(defun count-change (amount)
  (cc amount 6))

(defun cc (amount kinds)
  (cond ((= amount 0) 1)
        ((or (< amount 0) (= kinds 0)) 0)
        (t (+ (cc amount (- kinds 1))
              (cc (- amount (first-denom kinds)) kinds)))))

(defun first-denom (kinds)
  (cond ((= kinds 1) 1)
        ((= kinds 2) 2)
        ((= kinds 3) 5)
        ((= kinds 4) 10)
        ((= kinds 5) 20)
        ((= kinds 6) 50)))
```