

Principi programskih jezikov

Aritmetični izrazi

Cela števila: + x spremenljivke

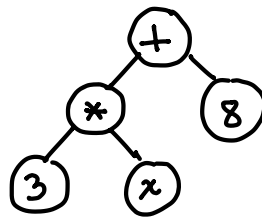
x ima vrednost 7

$$3 * x + 8 \rightsquigarrow 29$$

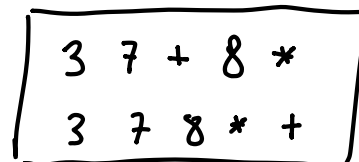
Sintaksa

Konkretna: "3*_x+_8" niz znakov

Abstraktna: drevo

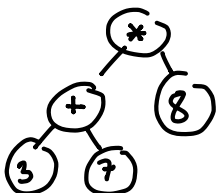


Konkretna RPN

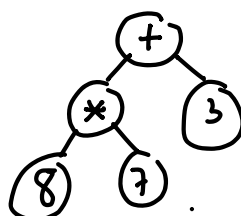
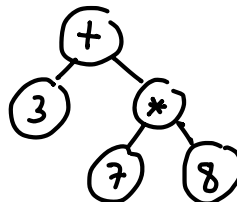


3 + 7 * 8

↙
(3 + 7) * 8

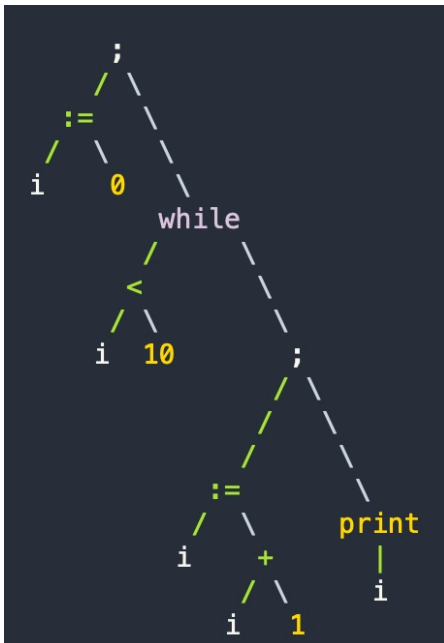


→ 3 + (7 * 8)



"8 * 7 + 3"

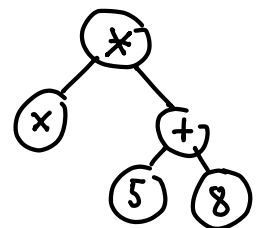
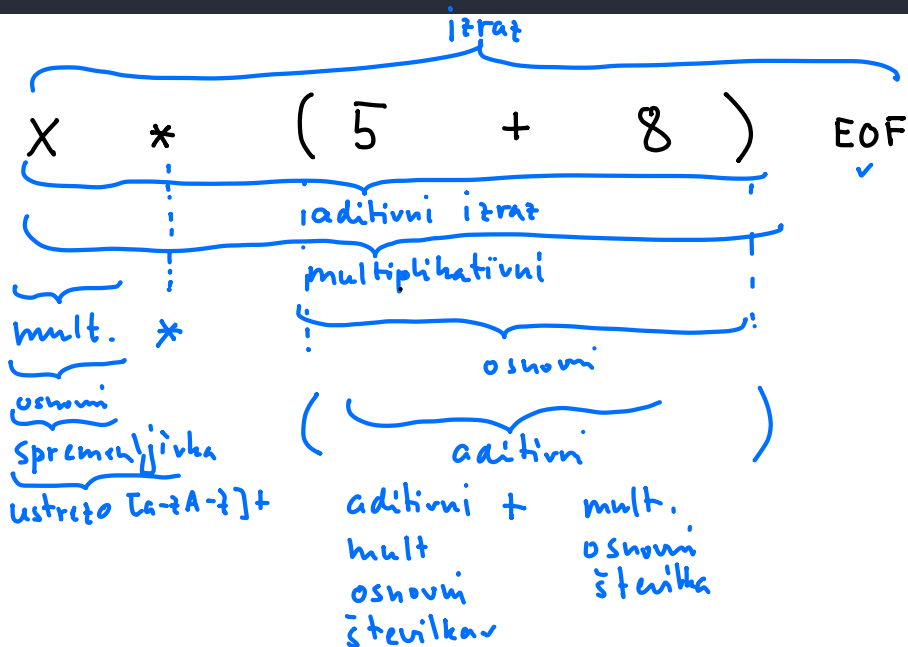
"59"



```
i := 0 ;
while i < 10:
    i := i + 1 ;
    print i
```

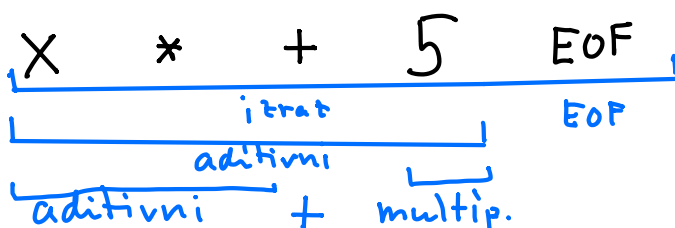
```
i = 0 ;
while (i < 10) {
    i = i + 1 ;
    System.out.println(i);
}
```

```
<izraz> ::= <aditivni-izraz> EOF
<aditivni-izraz> ::= <multiplikativni-izraz> | <aditivni-izraz> + <multiplikativni-izraz>
<multiplikativni-izraz> ::= <osnovni-izraz> | <multiplikativni-izraz> * <osnovni-izraz>
<osnovni-izraz> ::= <spremenljivka> | <številka> | ( <aditivni-izraz> )
<spremenljivka> ::= [a-zA-Z]+
<številka> ::= -? [0-9]+
```



$(X * (5 + 8))$
 $X * (5 + 8)$

```
<izraz> ::= <aditivni-izraz> EOF
<aditivni-izraz> ::= <multiplikativni-izraz> | <aditivni-izraz> + <multiplikativni-izraz>
<multiplikativni-izraz> ::= <osnovni-izraz> | <multiplikativni-izraz> * <osnovni-izraz>
<osnovni-izraz> ::= <spremenljivka> | <številka> | ( <aditivni-izraz> )
<spremenljivka> ::= [a-zA-Z]+
<številka> ::= -? [0-9]+
```



multipl.
osnovni
· ?!

osnovni
števila
✓

if <pogoj>
 <ukati> ?!
 <ukati>
else

Dinamična semantika

$\alpha \beta \gamma \delta \lambda$
 $\theta \eta \{ \xi \xi$
↑ ↑ ↑
theta zeta ksi

$$x + 3 * (y + 8)$$

Vrednosti spremenljivk:

$$\eta = [x \mapsto 2, y \mapsto 10] \quad \text{okolje}$$

$$[x \mapsto 2, z \mapsto 10]$$

↑ eta

$$\eta(x) = 2$$

$$\eta(y) = 10$$

$$\eta(z) \quad ?!$$

"v okolju η
se izraz e
evalvira / ovrednoti v
število n "

Relacija:

$$\eta \mid e \hookrightarrow n$$

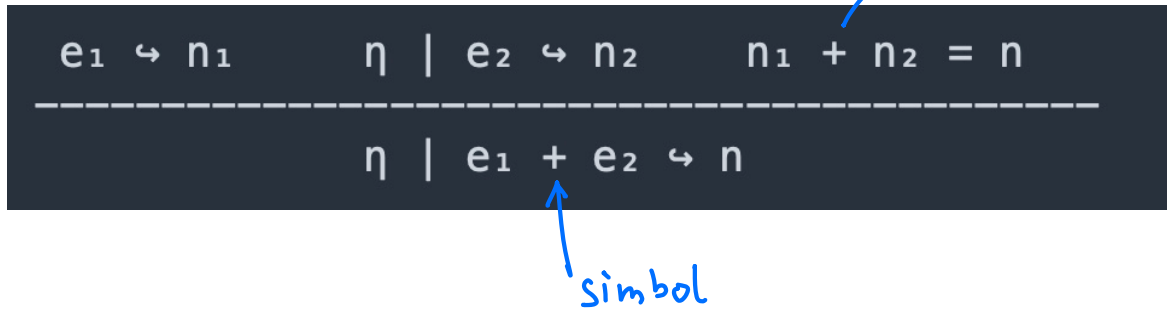
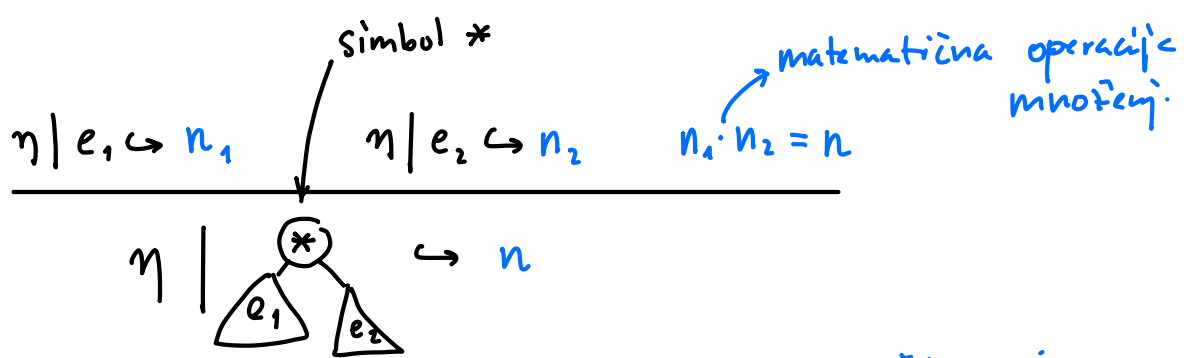
↑ ↑ ↑
okolje izraz vrednost

$$[x \mapsto 3, y \mapsto 5] \mid 3 + x \hookrightarrow 6 \quad \checkmark$$

$$[x \mapsto 3] \mid 3 + x \hookrightarrow 17 \quad \times$$

$$[] \mid 3 + 8 \hookrightarrow 22 \quad \times$$

$$[x \mapsto 3] \mid 3 + z \hookrightarrow 17 \quad \times$$



izpeljava

$\frac{[x \mapsto 3](x) = 3}{[x \mapsto 3] \mid x \hookrightarrow 3}$	$\frac{\frac{[x \mapsto 3] \mid 3 \hookrightarrow 3}{[x \mapsto 3] \mid 3 * 7 \hookrightarrow 21} \quad \frac{[x+3] \mid 7 \hookrightarrow 7}{3 * 7 = 21}}{[x \mapsto 3] \mid 3 + 21 \hookrightarrow 24}$
$\frac{[x \mapsto 3] \mid x \hookrightarrow 3 \quad [x \mapsto 3] \mid 3 * 7 \hookrightarrow 21}{[x \mapsto 3] \mid x + 3 * 7 \hookrightarrow 24}$	

Semantika malih korakov:

$$3 * 7 + 8 * 5 \mapsto$$

$$21 + 8 * 5 \mapsto$$

$$24 + 40 \rightarrow$$

$$64$$

Ukazni programski jezik

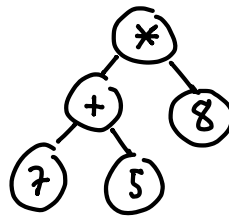
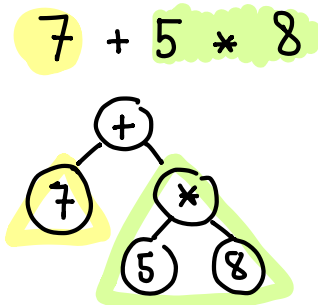
24.2.2025

- Jezik:
- cela števila + * = <
 - boolovi izrazi: true, false, ^, v, ¬
 - if-then-else, while, spremenljivke

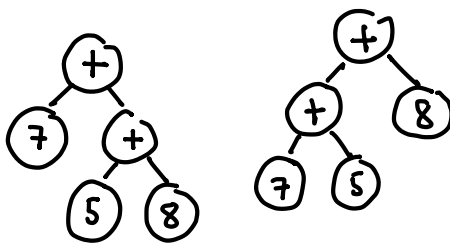
```
{aritmetični-izraz} ::=  
  {spremenljivka} |  
  {številka}  
  {aritmetični-izraz} + {aritmetični-izraz} |  
  → {aritmetični-izraz} * {aritmetični-izraz}
```

dvoumna

- Določimo
- prioriteto (kateri ima prednost)
 - asociativnost operatorjev:
levo, desno, nič



7 + 5 + 8



$$(x + y) + z \stackrel{?}{=} x + (y + z)$$

7 - 5 - 8

$$(7 - 5) - 8 = -6$$
$$7 - (5 - 8) = 10$$

levo * : $a * b * c = (a * b) * c$

desno * : $a * b * c = a * (b * c)$

nič * : $\Leftrightarrow$

$$(p \Leftrightarrow q) \Leftrightarrow r$$

+ - *

$$a^{b^c} = a^{(b^c)}$$

$$a^b \wedge c$$

$$x + 2y = 0 \Leftrightarrow x = -2y \Leftrightarrow \frac{x}{2} = -y$$

te tri izjave
so ekvivalentne

$$(x + 2y = 0 \Leftrightarrow x = -2y) \Leftrightarrow \frac{x}{2} = -y$$

$$\text{true} \Leftrightarrow \frac{x}{2} = -y$$

$$\frac{x}{2} = -y$$

Veliki koraki: $\eta \mid e \hookrightarrow v$
 $\uparrow$ $\uparrow$
 izraz vrednost

Mali koraki: $\eta \mid e \mapsto e'$
 $\uparrow$ $\uparrow$
 izraz izraz

Kratkostična konjunkcija

$$\eta \mid b_1 \hookrightarrow \text{false}$$

$$\eta \mid b_1 \text{ and } b_2 \hookrightarrow \text{false}$$

$$\eta \mid b_1 \hookrightarrow \text{true} \quad \eta \mid b_2 \hookrightarrow v_2$$

$$\eta \mid b_1 \text{ and } b_2 \hookrightarrow v_2$$

$$v_1, v_2 \in \{\text{false}, \text{true}\}$$

Dolgostična konjunkcija

$$\eta \mid b_1 \hookrightarrow v_1 \quad \eta \mid b_2 \hookrightarrow v_2$$

$$\eta \mid b_1 \text{ and } b_2 \hookrightarrow v_1 \wedge v_2$$

if (i < a.length && a[i] > 0) { }

if (f(7) && g(8)) { ... }



najprej ~~b~~₁ = f(7);

potem ~~b~~₂ = g(8);

if (b₁ && b₂) { }

while (i++ < n) { ... }

INC

Primer izvajanja programa:

$([x \mapsto 6], (\text{while } \underbrace{x < 7}_{\text{true}} \text{ do } x := x + 1 \text{ done})) \mapsto$

$([x \mapsto 6], (\underbrace{x := x + 1}_{\downarrow} ; \text{while } x < 7 \text{ do } x := x + 1 \text{ done})) \mapsto$
 $([x \mapsto 6], x := \overset{7}{x+1}) \mapsto ([x \mapsto 7], \text{skip})$

$([x \mapsto 7], (\text{skip} ; \text{while } x < 7 \text{ do } x := x + 1 \text{ done})) \mapsto$

$([x \mapsto 7], (\text{while } \underbrace{x < 7}_{\text{false}} \text{ do } x := x + 1 \text{ done})) \mapsto$

$([x \mapsto 7], \text{skip})$

$\llbracket 3 + 8 \rrbracket$

denotacija / pomen izraza $\underbrace{3 + 8}$



"3 + 8"

$\llbracket (x := 3 ; y := x + 8) \rrbracket$

$\llbracket x + 3 \rrbracket : \text{Env} \rightarrow \mathbb{Z}$

$\llbracket x \rrbracket \eta$

$\llbracket e_1 = e_2 \rrbracket(\eta) = (\llbracket e_1 \rrbracket(\eta) = \llbracket e_2 \rrbracket(\eta))$

$\llbracket e_1 < e_2 \rrbracket(\eta) = (\underbrace{\llbracket e_1 \rrbracket(\eta)}_{\text{celo št.}} < \underbrace{\llbracket e_2 \rrbracket(\eta)}_{\text{celo št.}})$

znak
manjše

mat. relacij.
manjše

$\perp$ ali T

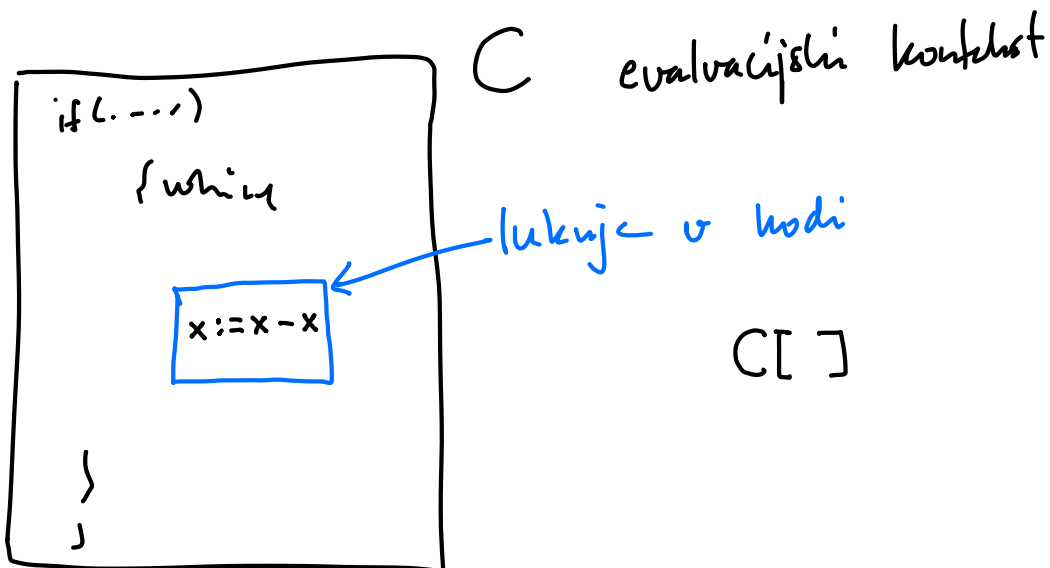
$$\rightarrow \llbracket e_1 < e_2 \rrbracket(\eta) = \begin{cases} T & \text{če } \llbracket e_1 \rrbracket(\eta) < \llbracket e_2 \rrbracket(\eta) \\ \perp & \text{sicer} \end{cases} \quad \text{NOOBSKO}$$

if (p) { return true; } else { return false; } NOOBSKO
OKROG RITI V ŽEP

$\Downarrow$
return p;

$$\llbracket \text{while } b \text{ do } c \text{ done} \rrbracket(\eta) = \begin{cases} \eta & \text{če } \llbracket b \rrbracket(\eta) = \perp \\ \llbracket c; \text{while } b \text{ do } c \text{ done} \rrbracket(\eta) & \text{sicer} \\ \text{"} \\ \llbracket \text{while } b \text{ do } c \text{ done} \rrbracket(\llbracket c \rrbracket(\eta)) \end{cases}$$

ciklična / rekurzivna definicija



Dokazovanje pravilnosti

Zapis lokalnih spremenljivk

Java:

```
{ int x = 42;  
  int z = 0;  
  if (x < 5) {  
    int y = 8;  
    ...  
  }  
  ...  
}
```

Comm:

```
new x := 42 in  
[ new y := x + 10 in  
  [ print x;  
    ...  
  ],  
  print 15  
]
```

Specifikacija: opis, kaj naj program počne

Implementacija: program

CompCert

$\{x > 10\} \quad x := x + 8 \quad \{x > 15\}$ ✓

$\{x > 10\} \quad x := x + 3 \quad \{x > 15\}$ ✗

C: $\{x < 10\}$
 $*p = 15;$
 $\{x < 10\}$

Težava: kaj, če p kaže na x?

"aliasing problem"

*p je "vzdručen" na x

```
{ b ≥ 0, i = 0, p = 1 }  
while i < b do  
    p := p * a;  
    i := i + 1  
done
```

i	p	$p = a^i$
0	1	✓
1	a	✓
2	a^2	✓
3	a^3	✓
4	a^4	✓

λ -račun

Funkcijski predpis (anonimna funkcija)

$$x \mapsto e$$

$$x \mapsto x^2 + 3x + 7 \quad "x \text{ se slika v } \dots"$$

Imenovana funkcija

$$f(x) := x^2 + 3x + 7$$

$$f := (x \mapsto x^2 + 3x + 7)$$

Imenovano število

$$a := 3 + \sqrt{5}$$

Uporaba ali aplikacija:

$\rightarrow$ argument

$$f(3)$$

$$(x \mapsto x^2 + 3x + 7)(3) = 3^2 + 3 \cdot 3 + 7$$

$$= 25 \quad \text{Pozor: to je že nadaljnje računanje!}$$

zamenjamo (substituiramo) x z izrazom 3

$$(x \mapsto x^2 + 3 \cdot x + 7)(a+6) = (a+6)^2 + 3 \cdot (a+6) + 7$$

Vezane in proste spremenljivke

$$\underline{x} \mapsto \underbrace{a \cdot x^2 + x + a}_{\text{območje veljavnosti}}$$

↑
deklariramo
(uvedemo) x

x je vezan (lokalen)

a je prost

$$a \mapsto a \cdot x^2 + x + a$$

a je vezan

x je prost

$$x \mapsto a + x$$

"prištej a "

$$a \mapsto a + x$$

"prištej x "

$$x \mapsto a + x^2$$

"kvadriraj in prištej a "

$$\int \underline{(x^2 + a \cdot x)} \, dx$$

$$\sum_{i=1}^n \underline{i \cdot (i+k)}$$

$$\lim_{\underline{x \rightarrow a}} \frac{x-a}{\underline{x+a}}$$

$$\exists \underline{x} \in \mathbb{R}. \quad \underline{x^3 = y}$$

"obstaja realno število,
katerega kub je enak y "

$$x^3 = y$$

x in y prosta

```

    ⋮
    for (i=0; i < n; i++) {
        S += i;
        print i;
    }

```

Zamenjava ali substitucija

izraz e , spremenljivka x , izraz e'

$e[e'/x]$ v e zamenjaj x z e'

Zapisi:

$e[x \mapsto e']$

$e[x := e']$

- $(x^2 + 3 \cdot x + 7)[3/x]$ je enako $3^2 + 3 \cdot 3 + 7$,
- $f(a + b)[(b + 1)/a]$ je enako $f((b + 1) + b)$,
- $f(a + b)[(x \mapsto x^2)/f]$ je enako $(x \mapsto x^2)(a + b)$.

$$\left(\int_0^1 (x^2 + b) dx \right) [(7+a)/b] = \int_0^1 (x^2 + (7+a)) dx$$

$$\left(\int_0^1 (x^2 + b) dx \right) [(7+x)/b] \stackrel{\text{NARJBE}}{=} \int_0^1 (x^2 + 7 + x) dx$$

$$\left(\frac{1}{3} + b \right) [(7+x)/b]$$

$$\frac{1}{3} + (7+x)$$

$$\frac{1}{3} + 7 + \frac{1}{2}$$

$$7 \frac{5}{6}$$

Pravilno : vezano spremenljivko preimenujemo

$$\left(\int_0^1 (x^2 + b) dx \right) [(7+x)/b] =$$

$$\left(\int_0^1 (z^2 + b) dz \right) [(7+x)/b] =$$

$$\int_0^1 (z^2 + (7+x)) dz$$

Gnezdjeni funkcijski predpisi

$$x \mapsto (y \mapsto x \cdot x + y)$$

x se slika v funkcijski predpis " y se slika v $x \cdot x + y$ "

$$(x \mapsto (y \mapsto x \cdot x + y)) (42) =$$
$$y \mapsto 42 \cdot 42 + y$$

Dvojna aplikacija

$$(x \mapsto (y \mapsto x \cdot x + y)) (42) (1)$$
$$(y \mapsto 42 \cdot 42 + y) (1)$$
$$42 \cdot 42 + 1$$

$$(f \mapsto f(f(3))) (n \mapsto n \cdot n + 1)$$

$$(n \mapsto n \cdot n + 1) ((n \mapsto n \cdot n + 1) (3)) =$$

$$(n \mapsto n \cdot n + 1) (3 \cdot 3 + 1) = (3 \cdot 3 + 1) \cdot (3 \cdot 3 + 1) + 1$$

$f(x)$ $\sin \alpha$ Ax $f \ 7$

$$\sin^2 \alpha = (\sin \alpha)^2$$

Zapis v λ -računu:

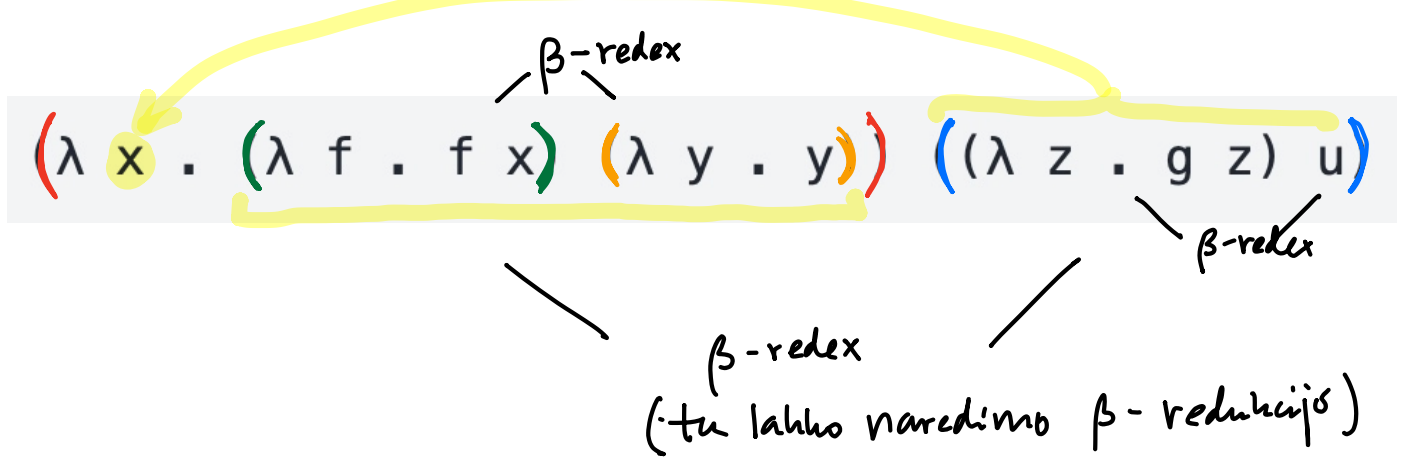
 $\lambda x. e_1 e_2 e_3$

- $((\lambda x. e_1) e_2) e_3$
- $(\lambda x. e_1 e_2) e_3$
- $\lambda x. ((e_1 e_2) e_3)$ ✓

 $\lambda x. ((\lambda y. (x y y)) x x)$ $\lambda x. (\lambda y. (\lambda z. (e)))$

Okrajšava: $\lambda x y z. e$

 $x \mapsto (y \mapsto x^2 + 7y)$ $\lambda x. (\lambda y. (x^2 + 7y))$ $\lambda x y. x^2 + 7$



$$(\lambda x . x x) (\lambda x . x x) =$$

$$(\lambda y . y y) (\lambda x . x x) =$$

$$(\lambda x . x x) (\lambda x . x x)$$

$$\lambda x . (\lambda y . y x)$$

Programiramo v λ -računu

Kompozicija: $(g \circ f)(x) := g(f x)$

$$(\text{compose } g \ f) \ x = g(f x)$$

$$\text{compose } g \ f = \lambda x . g(f x)$$

$$\text{compose } g = \lambda f . \lambda x . g(f x)$$

$$\text{compose} = \lambda g . \lambda f . \lambda x . g(f x)$$

$$= \lambda g \ f \ x . g(f x)$$

Const $c =$

$$(\lambda c . \lambda x . c) \ c =$$

$$(\lambda \tilde{z}nj. \lambda x. \tilde{z}nj) C = \lambda x. C$$

Pari: (a, b)
 $\lambda f. f a b$
↑ ↑
to "hranimo" v tem izrazu

$\underbrace{\hspace{10em}}_{\text{"objekt"}}$

$$\text{first } p = p(\lambda x y. x)$$

Naslednik:

n število

$$n f x = \underbrace{f (f (\dots f x))}_n$$

$$\begin{aligned}
 (\text{succ } n) f x &= \underbrace{f (f (f (\dots f x)))}_{n+1} \\
 &= f (\underbrace{f \dots f x}_n) \\
 &\quad f (n f x)
 \end{aligned}$$

$$\text{succ} = \lambda n f x. \underbrace{f (n f x)}_{\substack{\hookrightarrow n\text{-krat uporabi } f \text{ na } x \\ n+1 \text{ uporabe } f}}$$

$$+ := \lambda \underbrace{n \ m \ f \ x}_{\text{številci}} . \underbrace{(n \ f)}_{f \text{ --- } f \atop n} \underbrace{((m \ f) \ x)}_{(f(f(\dots f \ x))) \atop m}$$
$$\lambda \ m \ n \ f \ x \ . \ m \ (n \ f) \ x$$

m -krat uporabi $\underbrace{f \circ f \circ \dots \circ f}_n$

$$\underbrace{f_1 \cdots f_n}_{n_1} \underbrace{(f_1 \cdots f_{n_2} \cdots f_{n_3} \cdots x)}_{m}$$

Skupni m. n. uporab f

Deklarativno programiranje

Konstrukcije množic

Kartezični produkt / zmnožek

množici A in B $A \times B$ zmnožek A in B
 element (a, b) kjer $a \in A$ in $b \in B$

projekcija: $A \times B \xrightarrow{\pi_1} A$

$\pi_2 \downarrow$

B

računska pravila:

$$\pi_1(a, b) = a$$

$$\pi_2(a, b) = b$$

Primer: $\{1, 2, 3\} \times \{\square, \Delta\} = \{ (1, \square), (2, \square), (3, \square), (1, \Delta), (2, \Delta), (3, \Delta) \}$

Primer: $\{1, 2, 3\} \times \{3\} = \{3\}$
 $(2, ?)$ 

$$\{1, 2, 3\} \times \{*\} = \{(1, *), (2, *), (3, *)\}$$

Zmnožek večih množic: $A \times B \times C \times D$

(a, b, c, d)

$a \in A$

$b \in B$

$c \in C$

$d \in D$

Vsota ali disjunktna unija

Množici A in B ----- $A + B$ vsota (koprodukt)

injekciji "iota" $\longrightarrow$ $l_1(a)$ za $a \in A$
 $l_2(b)$ za $b \in B$

Primeri: $\{1, 2, 3\} + \{3, 4\} = \{l_1(1), l_1(2), l_1(3), l_2(3), l_2(4)\}$

Pogosto v matematiki: $A \overset{+}{\oplus} B = (\{0\} \times A) \cup (\{1\} \times B)$
kodiranje vsote s
pomočjo $\cup$ in $\times$

$$\{1, 2, 3\} \oplus \{3, 4\} = \{(0, 1), (0, 2), (0, 3), (1, 3), (1, 4)\}$$

Imamo $u \in A + B \Rightarrow$ dve možnosti

• $u = l_1(x)$ za neki $x \in A$
ali

• $u = l_2(y)$ za neki $y \in B$

$$\{\text{false}\} + \{\text{true}\} = \text{bool}$$

Obravnava primerov.

EkspONENT

Množici A in B - ... B^A eksponent
 $A \rightarrow B$

elementi preslikave z domeno A in kodomeno B
 $\lambda x. \dots$ $x \mapsto \dots$

→ je desno asociativna $A \rightarrow B \rightarrow C = A \rightarrow (B \rightarrow C)$

uporaba / aplikacija

$f: A \rightarrow B$ $a \in A$
 $fa \in B$

Računsko pravilo: $(\lambda x. e) a = e[a/x]$

Primeri: $\mathbb{R} \rightarrow \mathbb{R}$ $(\lambda x. x)$, $\sin$, $\cos$, $\exp$

→ $\mathbb{R}_x \times \mathbb{R} \rightarrow \mathbb{R}$ $\max$, $+$, $\times$, $-$

→ $\mathbb{R}_x \rightarrow (\mathbb{R}_y \rightarrow \mathbb{R})$ $\lambda x. \lambda y. x^2 + 3xy$

$\mathbb{R}_x \rightarrow \mathbb{R}_y \rightarrow \mathbb{R}$
funkcija, ki sprejme
število in vrne funkcijo,
ki sprejme število in
vrne število

$\lambda x. \lambda y. x^2 + 3xy$

$\lambda x. \lambda y. x + y$

$\mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$
funkcija, ki sprejme par števil
in vrne število

$\mathbb{R} \times \mathbb{R}$
 $\lambda u. (\pi_1 u)^2 + 3(\pi_1 u)(\pi_2 u)$

+

$$(\mathbb{R} \rightarrow \mathbb{R}) \rightarrow \mathbb{R}$$

funkcija, ki sprejme funkcij
in vrne število

$$\lambda f. 42$$

$$\lambda f. f'(0)$$

$$\lambda f. \int_0^1 f(x) dx$$

$$\int_0^1$$

$$\lambda f. f(0)$$

let $p = e_1$ in e_2

$\{ \top \mid p = e_1; \\ e_2 \}$

let $p = e_1$ in

let $q = e_2$ in

$\boxed{e_2}$

$$(\mathbb{N} \rightarrow \emptyset) = \emptyset !$$

void f(int n)

$\mathbb{Z} \rightarrow \text{void}$

if b then x else y . . .

match b with

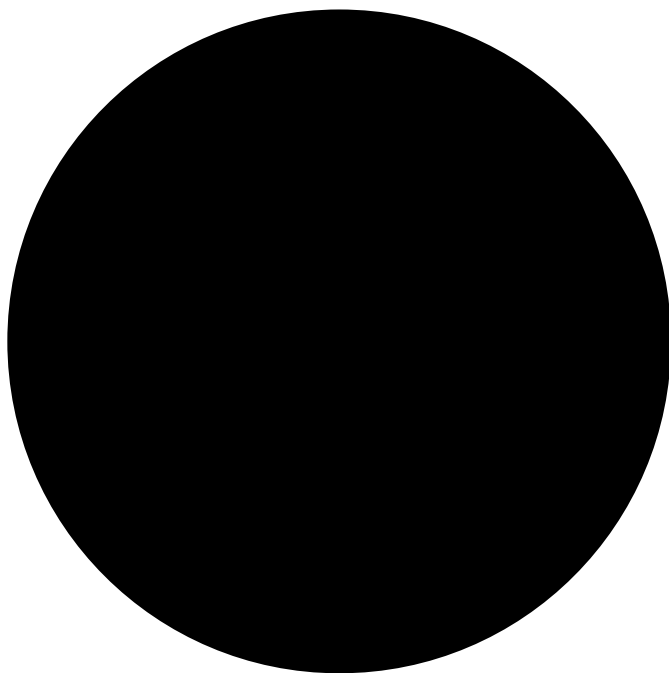
| true $\rightarrow$ x

| false $\rightarrow$ y

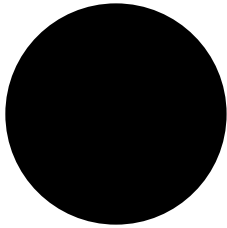
Fraktali so negibne točke

Preslikava f preslika množico S v unijo treh kopij S , skrčenih za faktor $1/2$:

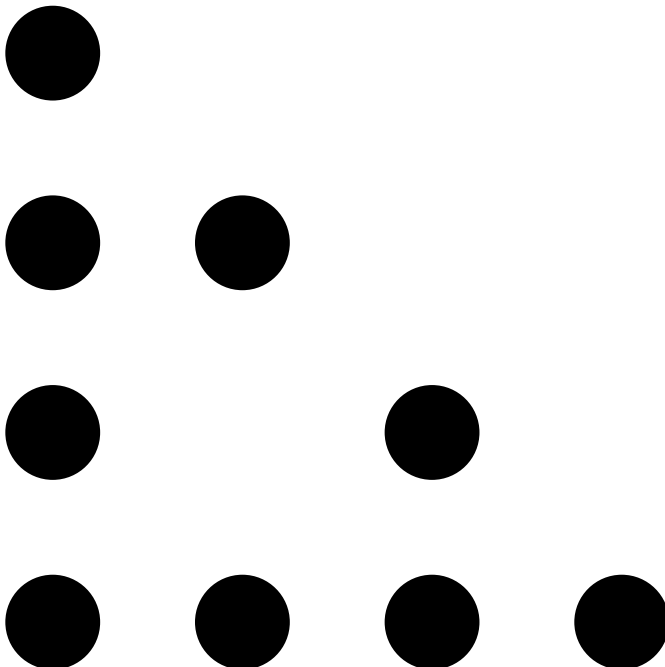
```
In[ ]:= f[s_] :=  
  {Scale[s, 0.5],  
   Translate[Scale[s, 0.5], {0.5, 0}],  
   Translate[Scale[s, 0.5], {0, 0.5}]  
  }  
  
In[ ]:= Graphics[Disk[{0.7, 0.3}, 0.25]]  
Out[ ]:=
```



```
In[ ]:= Graphics[f[Disk[{0.7, 0.3}, 0.25]]]  
Out[ ]:=
```

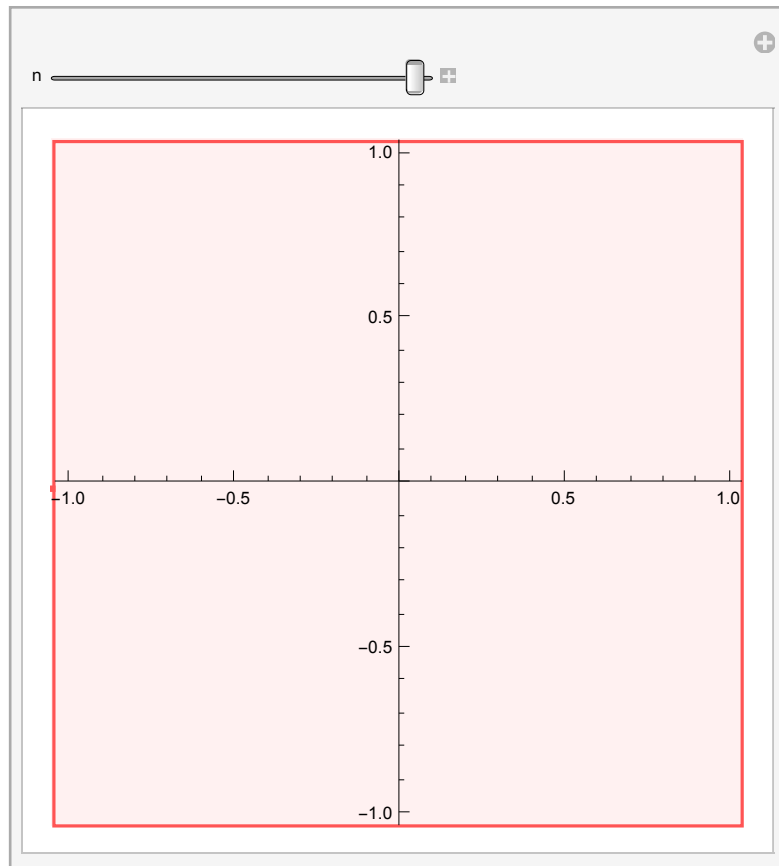


```
In[ ]:= Graphics[f[f[Disk[{0.7, 0.3}, 0.25]]]]  
Out[ ]:=
```



In[]:= Manipulate[Graphics[Nest[f, Disk[{0, 0}, 1 / 2], n], Axes → True], {n, 0, 6, 1}]

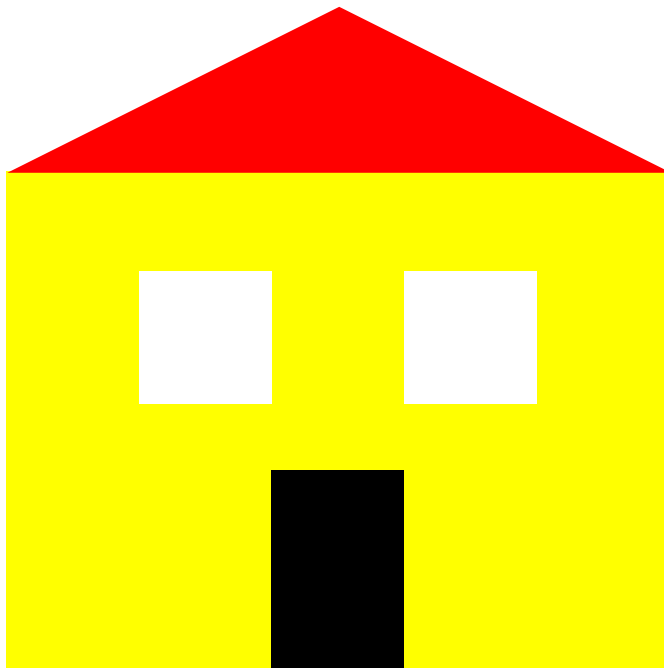
Out[]:=



```

In[ ]:= hisica = {{Yellow, Rectangle[{0, 0}, {1, 0.75}]},
  {Black, Rectangle[{0.4, 0}, {0.6, 0.3}]},
  {White, Rectangle[{0.2, 0.4}, {0.4, 0.6}]},
  {White, Rectangle[{0.6, 0.4}, {0.8, 0.6}]},
  {Red, Polygon[{0, 0.75}, {0.5, 1}, {1, 0.75}]]}
};
Graphics[hisica]
Out[ ]:=

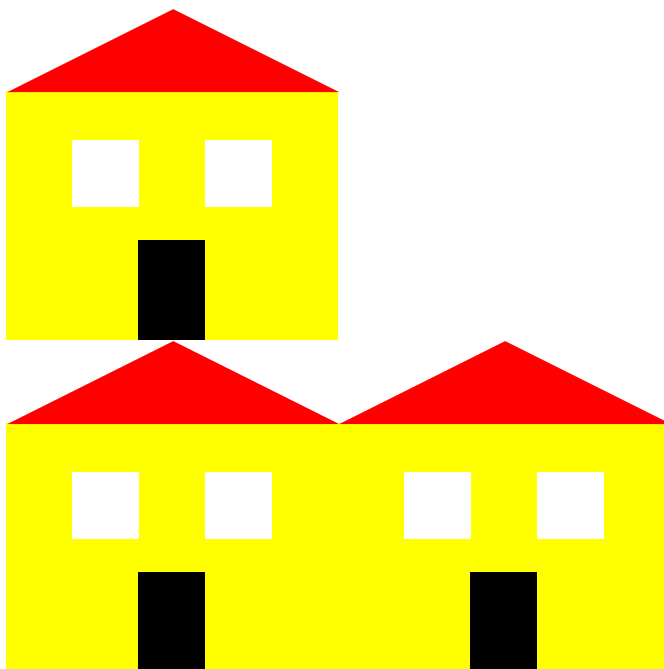
```



```

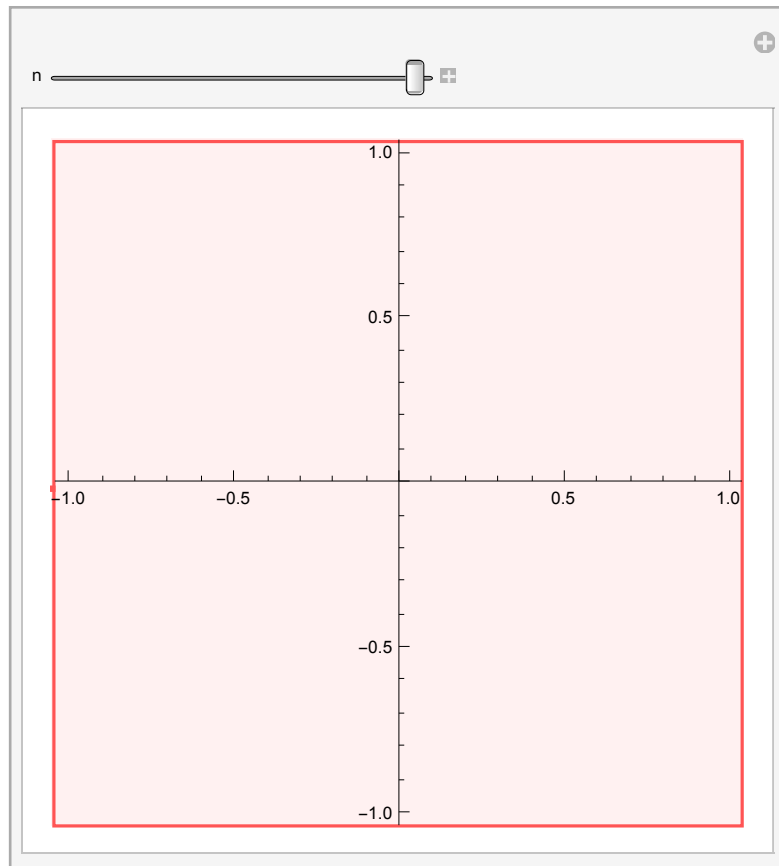
In[ ]:= Graphics[f[hisica]]
Out[ ]:=

```



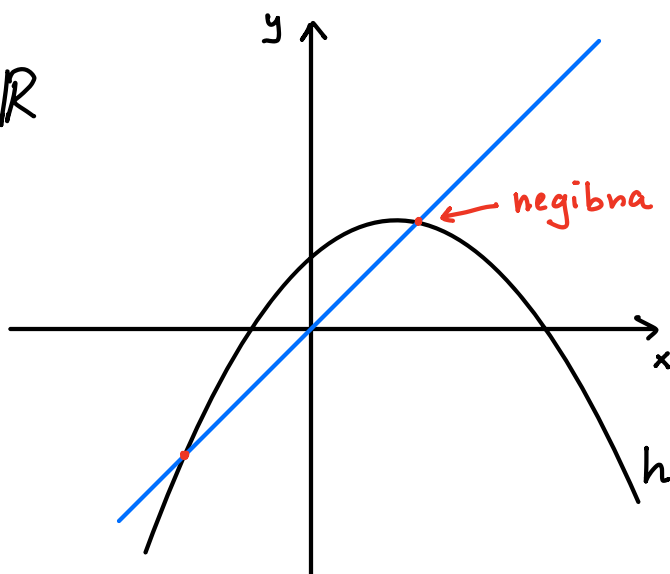
In[]:= Manipulate[Graphics[Nest[f, hisica, n], Axes → True], {n, 0, 6, 1}]

Out[]:=



Rekurzija

$$h: \mathbb{R} \rightarrow \mathbb{R}$$



$$x = h(x)$$

$$x^2 = \frac{1}{2}$$

$$x = \pm \sqrt{\frac{1}{2}} = 0.707...$$

$$x = h(x)$$

$$0 = \frac{1}{2} - x^2$$

$$x_0 = 1$$

$$x_1 = h(x_0) = \frac{1}{2}$$

$$x = \underbrace{\frac{1}{2} - x^2 + x}_{h(x)}$$

$$x_2 = h(x_1) \approx \frac{3}{4}$$

⋮

$\binom{n}{k}$ = št. podmnožic s k elem. množica z n elementi

$$\binom{n}{0} = 1$$

$$\binom{n}{n} = 1$$

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

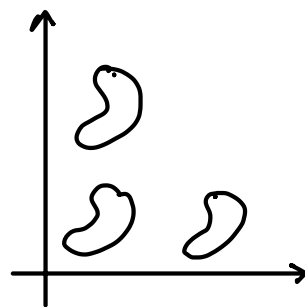
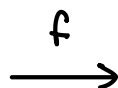
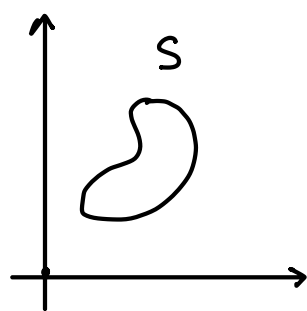
$$\{1, \dots, n\}$$

↑

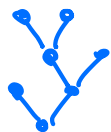
$$[1, 2, 1, 2, 1, 2, \dots]$$

Fraktali

$f(S) :=$



Induktivni tip : samo končni podatki
(seznam, drevo)



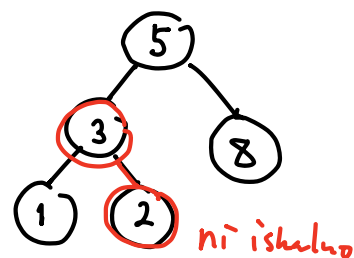
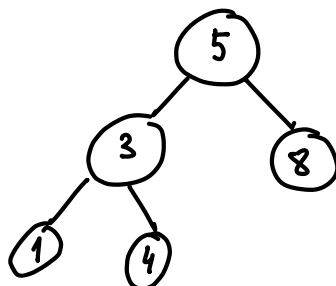
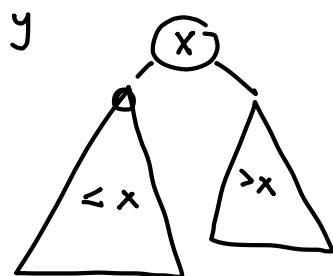
dobro-osnovano

[pravilno: dobro osnovani podatki,
well-founded]

Koinduktivni tip : končni & neskončni podatki

Strukturna rekurzija

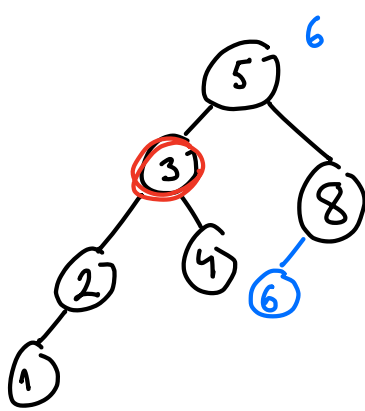
Iskalno drevo



velikost = število vozlišč

vr

y

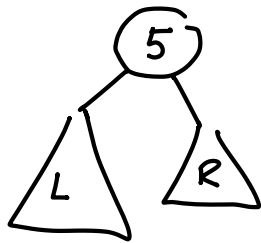


$\leadsto 1, 2, 3, 4, 5, 8$



Boolean blindness rant

y



primerjaj y in 5:

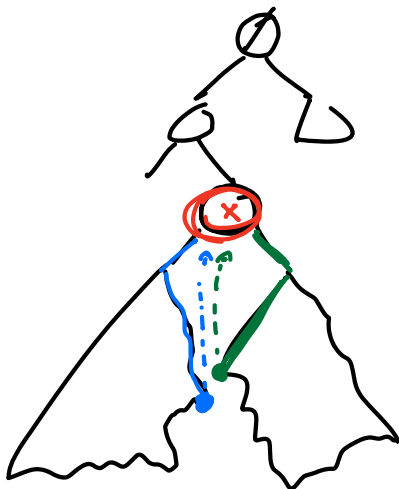
- $\rightarrow y = 5$, smo našli y
- $\rightarrow y < 5$, išči y v L
- $\rightarrow y > 5$, išči y v R

Podatkovna baza:

spol : BOOL

-- moški = TRUE
ženska = FALSE

type spol = Moški | Ženska | X



Koinduktivni tipi & izpeljava tipov

Induktivni: končni seznam, končna drevesa

Koinduktivni tipi:

- končne in neskončne strukture
- leni seznam: končni ali neskončni
- tok podatkov
- input/output

`type list = Nil | Cons of int * list`

OCaml

`data List = Nil | Cons Int List`

Haskell

Induktivni seznam so končni.

Koinduktivni seznam: končni in neskončni.

Tok podatkov tipa α :

- tok je urejeni par (d, t) , kjer je $d : \alpha$

podatek in t je tok

type stream = Stream of int * stream

↳ induktivno: končen tok Stream(3, Stream(4, ...))

↑
ne moremo
se ustaviti

končnega toka ni!

odgovor: ta tip je prazen

koinduktivno : ✓

Normalni ljude: List Int seznam celih števil

OCaml: int list

Normalni ljude: $\mathbb{R} \times \mathbb{R}$ (3, -5), List Int
int list

Haswell: $(\mathbb{R}, \mathbb{R})$ (3, -5)

$(\mathbb{R}, \mathbb{R}, \mathbb{R})$ urejene trojice

$[\mathbb{R}]$ tip seznamov realnih števil
 $[3, \sqrt{2}, \pi]$

Eratoštenovo sito

(2), (3), ~~4~~, (5), ~~6~~, (7), ~~8~~, ~~9~~, ~~10~~, //

Asistent pri Analizi
$$f x = \begin{cases} e_1 & \text{če } p_1 \\ e_2 & \text{če } p_2 \\ e_3 & \text{sicer} \end{cases}$$

Haswell
$$f x \mid p_1 = e_1$$
$$\mid p_2 = e_2$$
$$\mid \text{otherwise} = e_3$$

True

$f\ x = \text{if } p_1 \text{ then } e_1 \text{ else (if } p_2 \text{ then } e_2 \text{ else } e_3)$

$h\ x (f\ (g\ x\ y)) \Rightarrow h\ x \$ f \$ g\ x\ y$

○ : 1 : 2 : 3
↑

Zavlačevanje (thunking):

izraz $e : T$ ← recimo, da se zacikla

$\underbrace{\lambda().e}_{\text{unit} \rightarrow T}$ ← se ne zacikla

Zavlačevanje (thunk): $\underbrace{\lambda().e}_t$

Sprožimo (force): $t()$

Java:
(približno)

public static T t() { return e; }

Sprožimo s klicem t()

$$f(x) = f(x_0) + f'(x_0) \cdot (x - x_0) + \dots + \frac{f^{(n)}(x_0)}{n!} \cdot (x - x_0)^n + \dots$$

$$\sin x = x - \frac{1}{3!} x^3 + \frac{1}{5!} x^5 + \dots$$

↑

$\sin x$ lahko predstavimo s tohom koeficientov v potenčni vrsti

$$\begin{matrix} [0, 1, 0, -\frac{1}{3!}, 0, \frac{1}{5!}, 0, \dots] \\ x^0 \quad x^1 \quad x^2 \quad x^3 \end{matrix}$$

I/O

Program, ki izvaja operacijo na input/output :

- izračuna vrednost ^{→ tipa a} (brez I/O) in jo vrne
- pošlje podatek na output in nadaljuje
- prebere podatek z input in nadaljuje v odvisnosti od podatka

IO a =

| Return a

| Write String (IO a)

| Read (String → IO a)

Izpeljava tipov

Preverjanje tipov:

- programer zapiše tip izraza
- prog. jezik preveri, da izraz ima dani tip

Izpeljava:

- prog. jezik sam ugotovi tip izraza

```
if (i < 7) { ↗ preveri bool 3 + 8  
  int a = 3 + 8; podamo tip, se preveri  
  var b = 3 + 8 izpelji tip b
```

```
var c = 0.mojaMetoda(3);  
           ↘ vrne tip T
```

Primer:

```
fun x → x + 3  
  ↑      ↑  
  A      int  
  ↓      ↗  
  int?   int?  
  A = int
```

: A → int

A = int

odgovor;

int → int

if $\underbrace{3 < 5}_{\text{bool}}$ then $\underbrace{(\text{fun } x \rightarrow x)_{\alpha \rightarrow \alpha}}$ else $(\text{fun } y \rightarrow \underbrace{y + 3}_{\substack{\text{int?} \quad \text{int?} \checkmark}})$

$\downarrow$

$\alpha \rightarrow \alpha = \beta \rightarrow \text{int}$

$\swarrow$

$\beta \rightarrow \text{int}$

$\alpha \rightarrow \alpha = \text{int} \rightarrow \text{int}$

$\alpha = \text{int}$

odgovor: $\text{int} \rightarrow \text{int}$

1 Zpeljava tipov (še enkrat)

Združevanje (unification)

Imamo enačbe in neznanke $\alpha, \beta, \gamma, \delta, \dots$

(prolog: neznanke se začnejo z veliko začetnico)

Primer

$$f(\alpha) = f(g(x)) \Rightarrow \alpha = g(x)$$

Primer

$$\textcircled{1} \quad f(\alpha, g(\beta)) = f(a, \gamma)$$

$$\textcircled{2} \quad \gamma = g(g(\alpha))$$

Rešitev:

$$\gamma = g(g(\alpha))$$

Vstavimo γ v $\textcircled{1}$: $f(\alpha, g(\beta)) = f(a, g(g(\alpha)))$

$$\alpha = a$$

$$\gamma = g(g(a))$$

$$g(\beta) = g(g(\alpha))$$

$$g(\beta) = g(g(a))$$

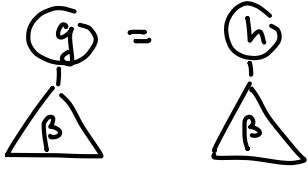
$$\beta = g(a)$$

Rešitev: $\alpha = a$
 $\beta = g(a)$
 $\gamma = g(g(a))$

Primer:

$$f(\underline{\alpha}, \underline{g(\beta)}) = f(\underline{a}, \underline{h(\beta)})$$

$\Downarrow$



• $\alpha = a \quad \checkmark$

• $\underline{g(\beta)} = \underline{h(\beta)}$ NI REŠITVE

Primer:

$$f(\underline{\alpha}, \underline{\alpha}) = f(\underline{a}, \underline{b})$$

$\Downarrow$

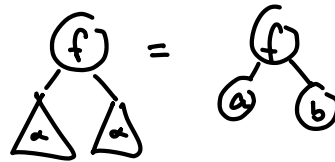
Rešitev:

$\alpha = a$

1. $\alpha = a$

2. $\alpha = b$

$\underline{a} = \underline{b}$



$$x^2 + x - 1 = 0$$

$$x = 1 - x^2$$

Primer:

$$f(\alpha) = f(g(\alpha))$$

$\Downarrow$

$$\underline{\alpha} = g(\underline{\alpha})$$

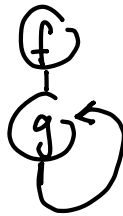
α se pojavi na obeh straneh

$$\alpha = g(g(\alpha))$$

$$\alpha = g(g(g(\alpha)))$$

$\vdots$

Ni rešitve



neskončna
rešitev

Postopek združevanja

Primer:

$$f(\alpha, \beta) = f(\alpha, \alpha)$$

$\Downarrow$

~~$\alpha = \alpha$~~

$\beta = \alpha$

Rešitev:

α poljuben

$\beta = \alpha$

Vprašanje: $\alpha = \alpha \vee \neg \alpha$ (rešitev: $\alpha = \text{true}$)

SIMBOLA $\vee$ IN $\neg$ IMATA POMEN!

Upoštevali smo neke dodatne enačbe:

$$\text{true} \vee \beta = \text{true}$$

$$\gamma \vee \delta = \delta \vee \gamma$$

$$\neg \text{true} = \text{false}$$

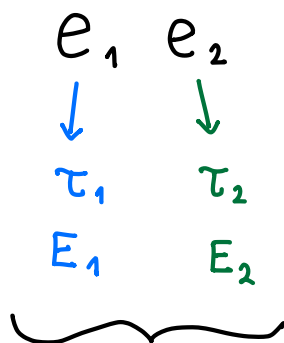
⋮

$$\neg \text{false} = \text{true}$$

Združevanje: Simboli, ki nastopajo, so neinterpretirani (o njih ne vemo nič)

$$\alpha = \text{or}(\alpha, \text{not}(\alpha))$$

Primer v prag. jeziku: izpeljava tipa



odgovor: Ψ

enačbe: $E_1, E_2, \tau_1 = \vartheta \rightarrow \Psi, \tau_2 = \vartheta$

Dodamo enačbo (naj bosta ϑ in Ψ **novi** **nemanki**)

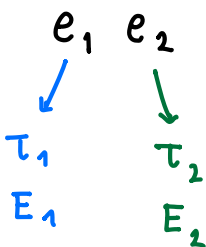
$$\tau_1 = \vartheta \rightarrow \Psi$$

$$\tau_2 = \vartheta$$

ϑ theta

Ψ psi

Izboljšava:



Naj bo Ψ nova nemanka:

odgovor: Ψ (tip aplikacij e_1, e_2)

enačbe: $E_1, E_2, \tau_1 = \tau_2 \rightarrow \Psi$

if e_1 then e_2 else e_3 } tip: τ_2
 enable:
 $\tau_2 = \tau_3$
 E_1, E_2, E_3
 $\tau_1 = \text{bool}$

Dodano: $\tau_1 = \text{bool}$

C/C++/Java: $e_1 ? e_2 : e_3$

$7 + (x < 5 ? x : 10) \checkmark$

Python: $e_2 \text{ if } e_1 \text{ else } e_3$

if e_1 then e_2 else e_3

$\tau_2 \neq \tau_3$

$\Rightarrow$ if e_1 then e_2 else e_3

$\sigma = \tau_1 + \tau_2$
 Type $\sigma = \begin{matrix} \text{Foo of } \tau_1 \\ \text{Bar of } \tau_2 \end{matrix}$

Java

class BlueBar {

class Baz

class Foo extends Baz

class Bar extends Baz

public Baz f() {

if (e_1) {

return e_2 ;

class Foo

} else {

return e_3 ;

class Bar

}

}

Rekurzija

Imamo rekurzivno definiciju

let rec $x = e$;
 $\swarrow$ x se pojavi u e

Običajno: let rec $f = \overbrace{\text{fun } y \rightarrow \dots}^{e \text{ je funkcija}}$;
 let rec $f \ y = \dots$

Rekurzivno: $\overset{\alpha}{x} = e$
 $\swarrow$ τ , enable E
 $\left. \begin{array}{l} x \text{ ima tip } \alpha \\ \text{enable:} \\ E, \alpha = \tau \end{array} \right\}$

① Uvedemo novu neznanu α i zabeležimo $x : \alpha$

Primer: $x = 1 :: 2 :: x$
 $\swarrow$ α
 int list
 $\swarrow$ α
 int list
 $\swarrow$ α
 int list

$\alpha = \text{int list}$ $\rightarrow \alpha \mapsto \text{int list}$
 $\text{int list} = \text{int list} \checkmark$
 $x = \text{int list}$
 $\text{int list} = \text{int list} \checkmark$

Seznam:

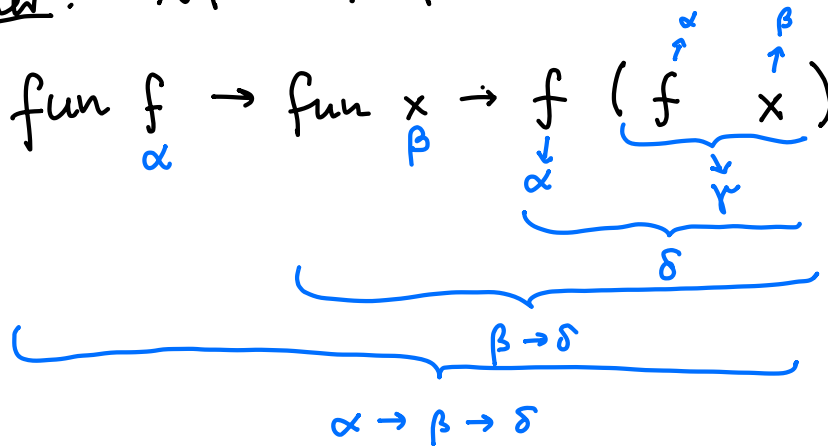
• $[] : \alpha \text{ list}$ (α nova neznanica)

• $e_1 :: e_2$
 $\swarrow$ $\swarrow$
 τ_1 τ_2
 E_1 E_2
 $\left. \begin{array}{l} \text{tip: } \tau_1 \text{ list} \\ \text{enable: } E_1, E_2, \tau_2 = \tau_1 \text{ list} \end{array} \right\}$

$2 :: [3; 4]$

Primer: $x = (3, x)$ odgovor: α
 $\alpha = \text{int} \times \alpha$
 NI REŠITVE

Primer: $\lambda f x. f(f x)$



1) $\alpha = \beta \rightarrow \gamma \checkmark$
 $\alpha = \gamma \rightarrow \delta \checkmark$
 $\alpha := (\beta \rightarrow \gamma)$
 $\beta \rightarrow \gamma = \gamma \rightarrow \delta \checkmark$
 $\hookrightarrow \beta = \gamma \checkmark \quad \boxed{\beta := \gamma}$
 $\hookrightarrow \gamma = \delta \quad \boxed{\gamma := \delta}$

Odgovor: $(\delta \rightarrow \delta) \rightarrow (\delta \rightarrow \delta)$

Rešitev:
 $\left[\begin{array}{l} \alpha := (\beta \rightarrow \gamma) \\ \beta := \gamma \\ \gamma := \delta \end{array} \right.$
 $\alpha := (\delta \rightarrow \delta)$
 $\beta := \delta$
 $\gamma := \delta$

IZREK O VARNOSTI

Program, ki ima tip, je varen.
 ?

Specifikacija, implementacija & abstrakcija

- Specifikacija:
 - opiše, zahteva, kaj naj program/knjžnica počne
- implementacija:
 - koda, ki nekaj počne

Primer iz matematike

Specifikacija

Def: Grupa je:

- množica G
- element $e \in G$
- preslikava $m: G \times G \rightarrow G$
- preslikava $i: G \rightarrow G$

opisuje sestavne dele grupe

signatura

ki zadoščajo:

1. $m(x, e) = x = m(e, x)$
2. $m(x, i(x)) = e = m(i(x), x)$
3. $m(x, m(y, z)) = m(m(x, y), z)$

kako se sestavni deli dovajajo
(logične izjave)

aksiomi za grupo

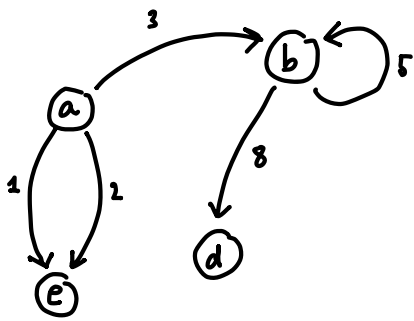
Implementacija

Primer:
 $G := \mathbb{Z}$
 $e := 0$
 $m := +$
 $i := -$

1. $x + 0 = x = 0 + x$ ✓

2. $x + (-x) = 0 = (-x) + x$ ✓

3. $x + (y + z) = (x + y) + z$ ✓



$$V = \{a, b, d, e\}$$

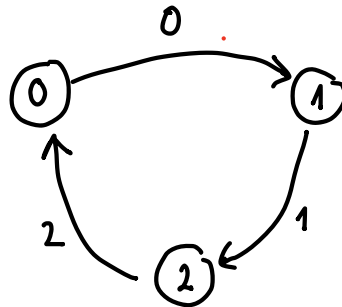
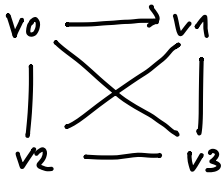
$$E = \{1, 2, 3, 5, 8\}$$

src: $E \rightarrow V$

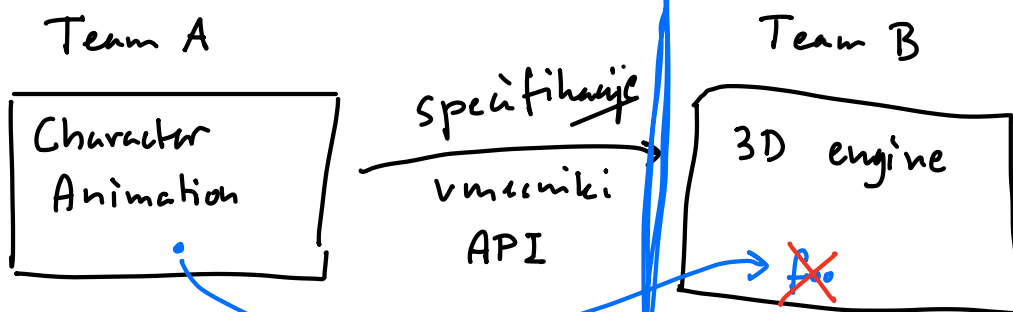
1	$\mapsto$	a
2	$\mapsto$	a
3	$\mapsto$	a
5	$\mapsto$	b
8	$\mapsto$	b

trg: $E \rightarrow V$

1	$\mapsto$	e
2	$\mapsto$	e
3	$\mapsto$	b
5	$\mapsto$	b
8	$\mapsto$	d



Abstrakcija



abstrakcija
(skrijemo podrobnosti implementacije)

Logično programiranje

program = logična pravila $P_1, \dots, P_n$
žagon = poižvedba / vprašanje Q
izvajanje = iskanje dokaza $P_1, \dots, P_n \Rightarrow Q$

Logične formule:

- izjavni račun:

- osnovne/primitivne formule ($P_1, P_2, \dots$)
- logični konstanti: $\perp$ neresnica, T resnica
- vezniki: $\wedge, \vee, \Rightarrow, \neg, \Leftrightarrow$

- predikatni račun

- osnovni predikati in relacije ($<, =, \text{"deli"}, \text{"-je mati od -"} \text{"})$
- termini, ki označujejo neke objekte/individue/entite (števila, točke, premice, stanja stroja/pomnilnika, ljudje, ...) - podatki
- kvantifikatorja:
 - $\forall$ univerzalni ("za vse")
 - $\exists$ eksistenčni ("obstaja")

Vse skupaj: logika prvega reda

Horn Hoareova logika

$$\Rightarrow \forall x_1, \dots, x_n. (P_1 \wedge P_2 \wedge \dots \wedge P_m \Rightarrow Q)$$

Za vse $x_1, x_2, \dots, x_n$, če velja P_1 in P_2 in $\dots$ in P_m , potem Q .

$\Rightarrow P_1, P_2, \dots, P_m, Q$ osnovni predikati in relacije

Primeri:

Termi / podatki :

- spremenljivke x, y , oseba, stanje
- konstante $0, \pi$, princHarry, začetnoStanje
- funkcijski simboli $+$, $-$, succ, višina($\dots$), naslednjeStanje($\dots$)

Posebni primeri: $\forall x_1, \dots, x_n. (P_1 \wedge P_2 \wedge \dots \wedge P_m \Rightarrow Q)$

① $m=0$: $\forall x_1, \dots, x_n. Q$ aksiom / dejstvo

Primer: $\forall x. \underbrace{x=x}_Q$

② $n=0$: $P_1 \wedge \dots \wedge P_m \Rightarrow Q$ ni spremenljivk

Primer: $\text{less}(4,3) \wedge \text{less}(2,3) \Rightarrow \text{less}(1,5)$

Kako s Hornovimi formulami podamo funkcije?

Seštevanje: $\forall x. 0 + x = x$
 $\forall x, y. \text{succ}(x) + y = \text{succ}(\underbrace{x + y}_z)$ } aksiomi za +

Tehnika: seštevanje + nadomestimo z relacijo
 $\text{plus}(x, y, z)$ "z je vsota x in y"

"Peanova Aksioma": $\forall x. \text{plus}(0, x, x)$
kot Hornovi formuli: $\forall x, y, z. \text{plus}(x, y, z) \Rightarrow \text{plus}(\text{succ}(x), y, \text{succ}(z))$
↑ on-line zapisnik: Vsota

Funkcija $f: X \times Y \rightarrow Z$

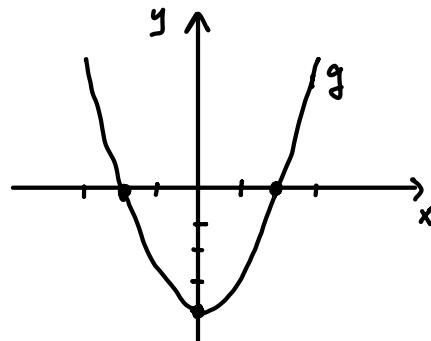
Relacija $R(x, y, z) \Leftrightarrow f(x, y) = z$

↳ Graf funkcije f

Zakaj "graf"?

$$g: \mathbb{R} \rightarrow \mathbb{R}$$
$$g(t) := t^2 - 4$$

$$R(x, y) \Leftrightarrow g(x) = y$$
$$x^2 - 4 = y$$



Formule, ki jih lahko pretvorimo v Hornovo obliko:

$P_1 \vee P_2 \Rightarrow Q$ je ekvivalentno $P_1 \Rightarrow Q$ in $P_2 \Rightarrow Q$

$(\exists x. P) \Rightarrow Q$ je ekvivalentno $\forall x. (P \Rightarrow Q)$

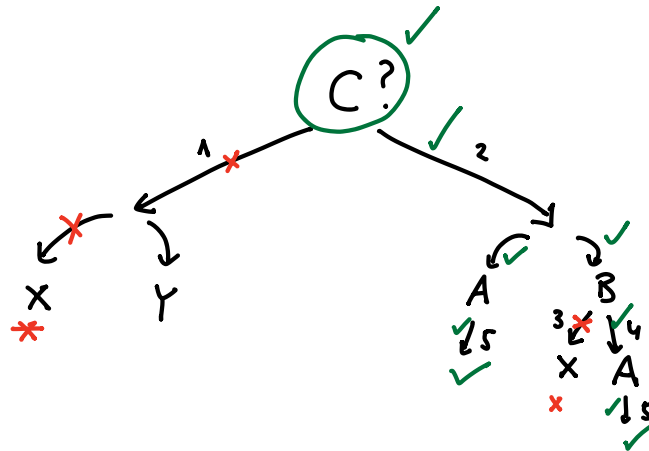
Formule, ki niso ekvivalentne Hornovim:

$\exists x. Q$

$\neg Q$

1. $X \wedge Y \Rightarrow C$
2. $A \wedge B \Rightarrow C$
3. $X \Rightarrow B$
4. $A \Rightarrow B$
5. A

sledi $C?$

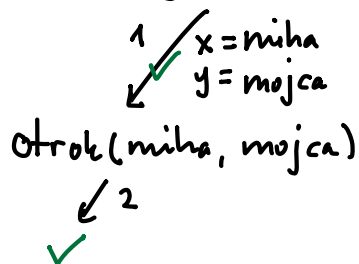


iskanje
dokaza
v globino

1. $\forall x y. \text{otrok}(x, y) \Rightarrow \text{mlajsi}(x, y)$
2. $\text{otrok}(\text{miha}, \text{mojca})$

$\text{mlajsi}(\text{miha}, \text{mojca})?$

✓



1. $\forall x. \text{sodo}(x) \Rightarrow \text{liho}(\text{succ}(x))$
2. $\forall y. \text{liho}(y) \Rightarrow \text{sodo}(\text{succ}(y))$
3. $\text{sodo}(\text{zero})$

$\text{sodo}(\text{succ}(\text{succ}(\text{succ}(\text{zero}))))$ X
 $\swarrow^2 \text{ } y = \text{succ}(\text{succ}(\text{zero}))$
 $\text{liho}(\text{succ}(\text{succ}(\text{zero})))$
 $\swarrow^1 \text{ } x = \text{succ}(\text{zero})$
 $\text{sodo}(\text{succ}(\text{zero}))$
 $\swarrow^2 \text{ } y = \text{zero}$
 $\text{liho}(\text{zero})$
X

Programiranje s Hornovimi formulami:

- program $H_1, \dots, H_n$ Hornove formule
- poizvedba $\exists x_1, \dots, x_k, G$?

$$\frac{(\eta, c_1) \mapsto (\eta', c_1')}{(\eta, (c_1 ; c_2)) \mapsto (\eta', (c_1' ; c_2))}$$

$\forall \eta, c_1, c_2, \eta', c_1'. (\eta, c_1) \mapsto (\eta', c_1') \Rightarrow (\eta, (c_1 ; c_2)) \mapsto (\eta', (c_1' ; c_2))$
 Hornova formula

OCaml $x :: y$

$x :: y :: z$

Prolog $[x | y]$

$[x, y | z]$

Programiranje z omejitvami

Prolog: program = seznam Hornovih formul
Aritmetika? + x -

① Hornove formule:

$n \in \mathbb{N}$ predstavimo z $\underbrace{\text{succ}(\text{succ}(\dots \text{succ}(\text{zero}) \dots))}_n$

$x + y = z$ predstavimo z vsota(x, y, z) + pravila

$x * y = z$ ---- zmožnek(x, y, z) + pravila

pomanjkljivosti: • eniški sistem

- počasno
- nepraktično

$2 \cdot 1 + 2 \cdot 2$?

Prologova aritmetika

- vgrajena števila
- operacije & primerjave + * < > ...
- računanje: is

X is (2 + 3) * 7

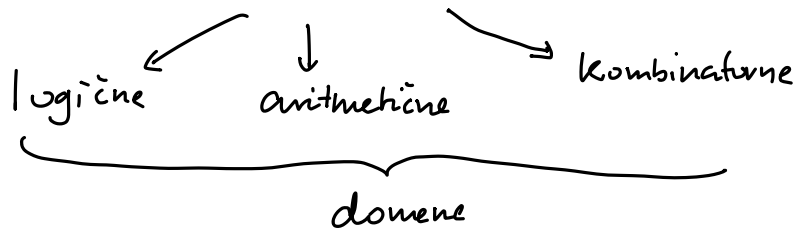
↑
spremenljivka

↑
izraz,
če vsebuje spremenljivke,
morajo imeti znane vrednosti

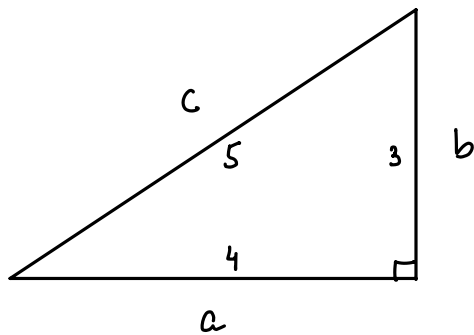
Prolog : program = seznam ^{logična} pravila
(omejitve)

Programiranje z omejitvami :

program = seznam omejitev



$$\text{profit} = (X^2 + \int -f + \sum \dots z)$$



$$a^2 + b^2 = c^2$$

permutacija $(\underset{N}{3}, P)$.

seznam L

dolžina $L = N$

P

1	2	3
1	3	2
2	1	3
2	3	1
3	1	2
3	2	1

6

5	3		7				
6			1	9	5		
	9	8				6	
8				6			3
4			8		3		1
7				2			6
	6					2	8
			4	1	9		5
				8			7
						7	9

← P
← Q
← R

Rows

all-distinct($[x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, \dots]$)

Podtipi & objekti

Polimorfizem $\leadsto$ izraz ima več tipov

$$\text{fun } x \rightarrow (\text{fun } y \rightarrow (x, y)) : \alpha \rightarrow \beta \rightarrow \alpha * \beta$$

poljubna

$$\begin{aligned} \text{int} &\rightarrow \text{int} \rightarrow \text{int} * \text{int} \\ \text{bool} &\rightarrow \text{int} \rightarrow \text{bool} * \text{int} \end{aligned}$$

$$42 \rightarrow \begin{cases} \text{int} \\ \text{float} \end{cases}$$

$$\begin{aligned} &\nearrow \text{int} \\ 42 + 7 &\quad \checkmark \\ \sin(42) &\quad \checkmark \\ &\searrow \text{float} \end{aligned}$$

$$\begin{aligned} &\text{int } x \\ &\sin(x) \\ &\quad \downarrow \\ &\text{pretvori int v float} \end{aligned}$$

Podtipi

Tip A je podtip tipa B, če lahko vrednosti tipa A uporabljamo, kot da bi imele tip B. (*)

$$A \leq B \quad \text{"A je podtip B"}$$

Primer: $\text{int} \leq \text{float}$ (v mnogih jezikih, ne v OCaml)

(*) kot pravilo:

$$\frac{e : A \quad A \leq B}{e : B}$$

Pozor: $A \leq B$ ni nujno "podmnožica"
(glej podtipe zapisov)

Splošna pravila za $\leq$

$$\begin{array}{c} \frac{}{A \leq A} \\ \text{refleksivnost} \end{array} \quad \frac{A \leq B \quad B \leq C}{A \leq C} \quad \text{transitivnost}$$

šibka urejenost

~~$$\frac{A \leq B \quad B \leq A}{A = B} \quad ?$$~~

antisimetričnost

ne velja v splošnem
(glej podtipe zapisov)

Pravilo za produkt:

$$\frac{A_1 \leq B_1 \quad A_2 \leq B_2}{A_1 \times A_2 \leq B_1 \times B_2}$$

Primer: $\text{int} \leq \text{float}$
 $(3, 1.7) : \text{float} \times \text{float}$
 $\uparrow$
 int

Pravilo za funkcije: !!

PAZI

$$\frac{B_1 \leq A_1 \quad A_2 \leq B_2}{(A_1 \rightarrow A_2) \leq (B_1 \rightarrow B_2)}$$

Primer:

$$\begin{array}{c} A_1 \quad A_2 \\ (\text{float} \rightarrow \text{float}) \end{array} \leq \begin{array}{c} B_1 \quad B_2 \\ (\text{int} \rightarrow \text{float}) \end{array} \quad \checkmark$$

$$\begin{array}{ll} B_1 \leq A_1 ? & \text{int} \leq \text{float} \quad \checkmark \\ A_2 \leq B_2 ? & \text{float} \leq \text{float} \quad \checkmark \end{array}$$

$$\begin{array}{l} \text{sin} : \text{float} \rightarrow \text{float} \\ \text{int} \rightarrow \text{float} ? \end{array}$$

$$\begin{array}{l} \text{int} \leq \text{float} \\ f : \text{int} \rightarrow \text{int} \\ f : \text{int} \rightarrow \text{float} ? \quad \checkmark \\ \quad \quad \quad 4 \quad \quad \quad 42 \\ f : \text{float} \rightarrow \text{int} ? \quad \times \\ \quad \quad \quad 3.7 \end{array}$$

Podtipi zapisov

Tip zapisa:
(reward)

{ ime :- string ; priimek :- string ; rojstvo :- datum }

↑
polja (field)

$$\{l_1: \tau_1, l_2: \tau_2, \dots, l_n: \tau_n\}$$
$$\{l_i : \tau_i\}_{i=1}^n$$

Primer :

kompleksno število :

```
tip { re: float, im: float }
```

Vrednost $\{ \text{re} = 3.1, \text{im} = -0.7 \}$

Podtipi zapisov:

SNEMAJ

→ po šlini

→ vrstni red

→ po globini

Podtipi v širino:

```
point = { x: float, y: float }
```

$c_{point} = \{ x: float, y: float, c: color \}$

$$c_{\text{point}} \leq \text{point}$$

Ali obravnavamo

$$\{x:\text{int}, y:\text{float}\} \text{ in } \{y:\text{float}, x:\text{int}\}$$

ket enaka tipa? DA.

$$\{ \underset{l_1}{\overset{\tau_1}{x}} : \text{float}, \underset{l_2}{\overset{\tau_2}{c}} : \text{color}, \underset{l_3}{\overset{\tau_3}{y}} : \text{int} \} \leq \{ \underset{k_1}{\overset{\sigma_1}{y}} : \text{int}, \underset{k_2}{\overset{\sigma_2}{x}} : \text{float} \} \quad \checkmark$$

$n=3 \quad m=2 \quad j=2$

Pravilo

$$\forall j \in \{1, \dots, m\}. \exists i \in \{1, \dots, n\}. k_j = l_i \wedge \sigma_j = \tau_i$$

$$\{ l_i : \tau_i \}_{i=1}^n \leq \{ k_j : \sigma_j \}_{j=1}^m$$

↑
imena polj (lejabli)

Primer :

$$\{ x : \text{float}, c : \text{color}, y : \text{int} \} \leq \{ \overset{\text{prepovedano}}{x} : \text{float}, \overset{\text{prepovedano}}{x} : \text{float}, y : \text{int} \}$$

Se ne zgodi

Primer: $\{ l_1 : \tau_1, \dots, l_n : \tau_n \} \leq \{ \}$

Podtipi v globino:

$$\frac{\tau_1 \leq \sigma_1 \quad \tau_2 \leq \sigma_2 \quad \dots \quad \tau_n \leq \sigma_n}{\{ l_1 : \tau_1, \dots, l_n : \tau_n \} \leq \{ l_1 : \sigma_1, \dots, l_n : \sigma_n \}}$$

Primer: $\{ x = 3; y = 5 \} : \{ x : \text{int}, y : \text{int} \}$
 $\wedge$
 $\{ x : \text{float}, y : \text{float} \}$ ✓

Podtipi v širino & globino:

= spremenimo $\leq$

za vsak $j \leq m$ obstaja $i \leq n$, da je $l_i = k_j$ in $A_i \leq B_j$

$$\{ l_1 : A_1; \dots; l_n : A_n \} \leq \{ k_1 : B_1; \dots; k_m : B_m \}$$

```
A = { mutable x : int; mutable y : int }
```

in

```
B = { mutable x : float; mutable y : float }
```

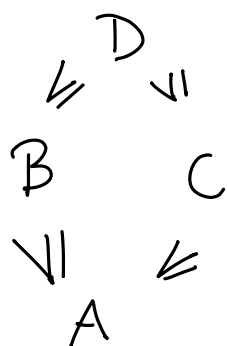
$a = \{ x=3 ; y=5 \} : A$

$a.x \leftarrow 7$

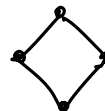
$a = \{ x=7 ; y=5 \}$

Če $a : B$ potem

$a.x \leftarrow 3.6$ **NE GRE**



"diamond"



class A extends ~~B, C~~

Podtipi na nivoju vmesnikov (API)

signatura

Sig

type t

val size : t → int

Objektno programiranje

ukazno programiranje (C, strojna koda, Fortran, Algol, Simula, Oberon)

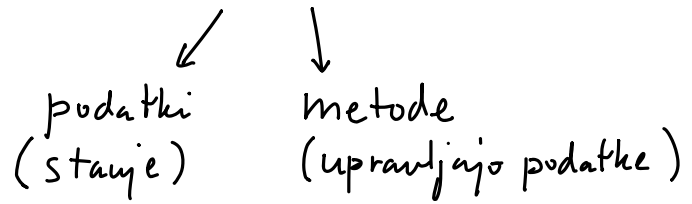
ukazi manipulirajo stanje

pomnilnik

registri

tabeli, spremenljivke

"Paketni" stanja $\rightsquigarrow$ OBJEKTI



Zapis : polja s podatki

objekt : atributi s podatki + metode
this/self objekt se lahko sklicuje sam nase

Objekt = rekurzivni zapis

- Strukturni podtipi (duck subtyping) 

$A \leq B$ ker to dopušča struktura tipov A in B
(pogledamo, kako sta A in B namejena)

- Nominalni podtipi
podtipi so določeni s hierarhijo,
ki jo določi programer

Java: podrazredi

$A \leq B$

public class A extends B {
...}

Java:

class A { int x; }

→ int x

class B extends A { int y; } → int x, int y

class C { int x; int y } → int x, int y

Nominalno

$B \leq A$

~~$C \leq A$~~

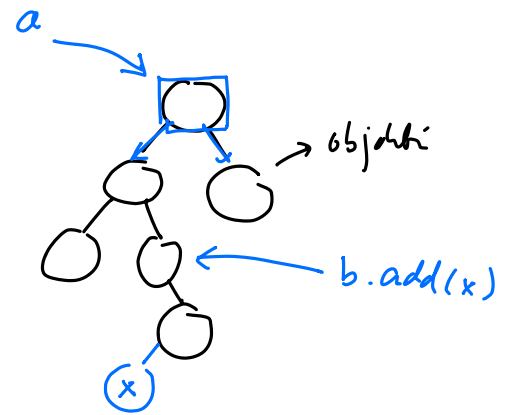
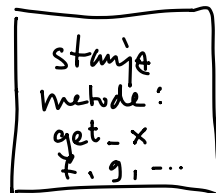
Strukturalno

$B \leq A$

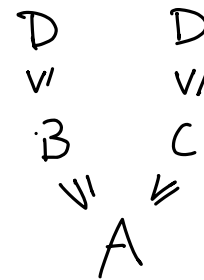
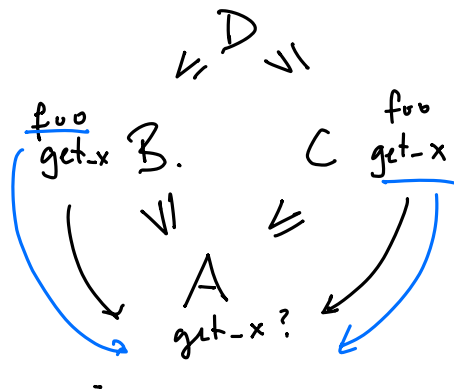
$C \leq A$

(Objekti & razredi) & razredi tipov

Objekt = enkapsulirano stanje



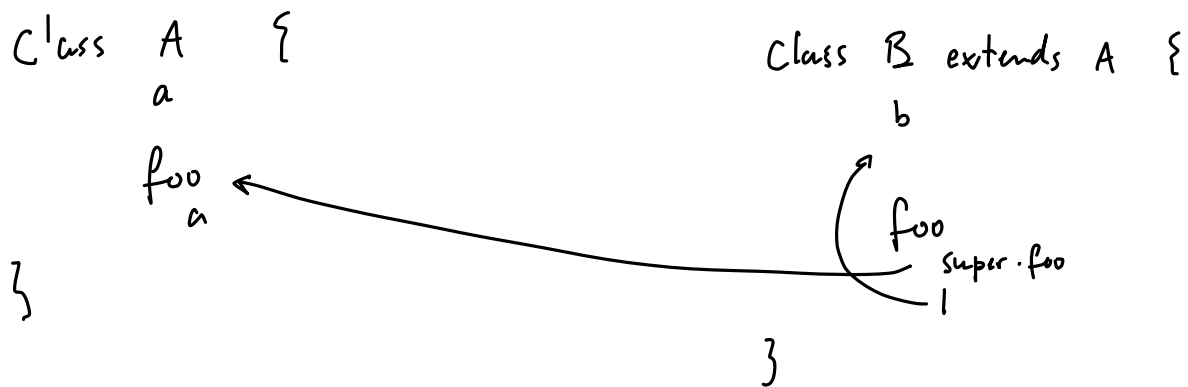
Koherenca



class A extends B, C
↑ ↑ ↑
this - super1 super2

~~get-x dobinno te pravega~~

class A extends B
↑ ↑
this super

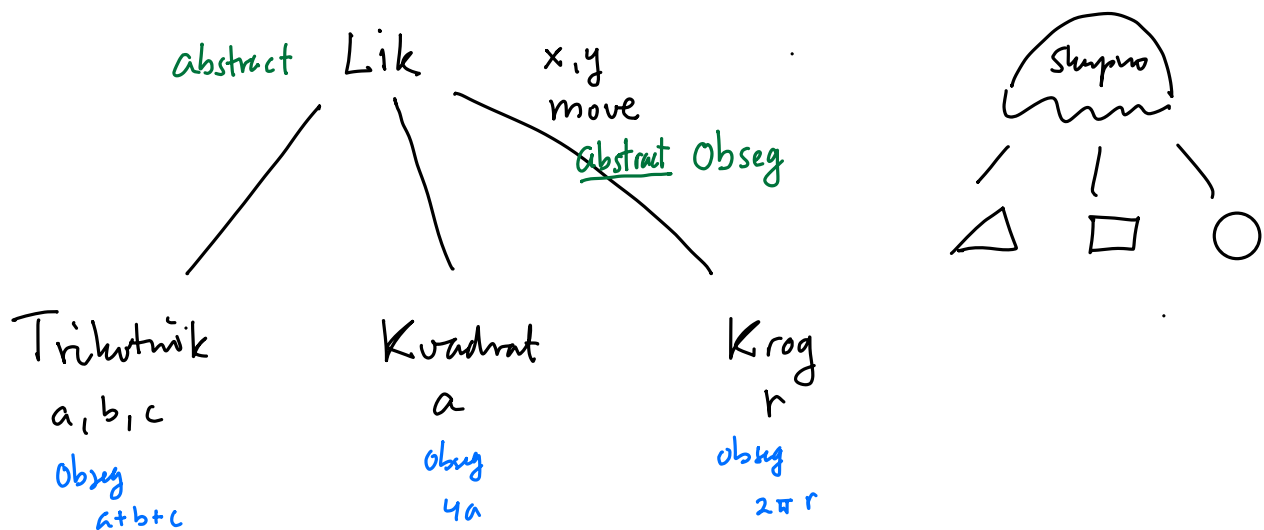


Class A {

f(int x)

f(bool b, int x)

Abstraktni razred:



interface I {
 f, g
 }

abstract class A {
 abstract f, g
 }

class C implements I

class C extends A

I, J, K

A, A', A''

Razredi tipov

Haskell, Lean

(type classes)

- Signature & structure (moduli)
- Razredi & objekti
- Razredi tipov

- "numerični tipi": int, float, int64, complex
skupno funkcionalst: +, *, -, 0, 1, ...
- tipi, ki "hranijo" podatke, ki jih lahko pregledamo / obdelamo
(iteriramo)

seznam, drevo, slovar

type class: opišemo, kaj lahko počnemo z
nekim tipom

- opis funkcionalnosti → type class
(vmesnik / specifikacija / API)
- implementacija → instance

"Sterilo bitov, ki jih zasede podatki"

Java: interface Sized {
int size();
}

✎

Tipi: Int, $a \rightarrow b$, Tree ...

class Cow a where
(opisano funkcionalnost a)

[a] seznam a-jev

data Tree a = Empty | Node a (Tree a) (Tree a)

Tree Int drevesa Int-ov

Tree: preslikava iz tipov
v tipov (type constructor)

Tree : $* \rightarrow *$
↳ "supertip tipov"

[-] : $* \rightarrow *$

Monade

Računski učinki

- interakcija programa z "zunanjim svetom":
 - input/output, komunikacijski tokovi,
 - stanje (avtomat, pomnilnik)
 - verjetnostno računanje (zunanji vir naključja)
- kontrolne strukture v programu:
 - izjema / napaka / manjkajoči podatek
 - nedeterminizem (možnih je več rezultatov)
 - stanje (spremenljivke)
 - divergenca (program se začne)

Monade

- posplošitev algebre: posebni primeri grupe, vektorski prostori, monoidi

Sestavni deli:

- čiste vrednosti (pure values) in izračune (computation), ki zajemajo računske učinke
- Vrednosti lahko pretvorimo/uložimo v izračune
- izračune lahko kombiniramo/komponiramo

Haskell

- monade $\rightarrow$ programski konstrukt, ki omogoča simulacijo / stil programiranja z učinki
- monade opišejo interakcijo z zunanjim svetom } good try
IO monada (hack)

Definicija monade:

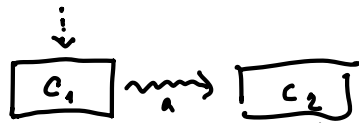
- 1) Konstruktor tipov $m : * \rightarrow *$
 $m : \text{Type} \rightarrow \text{Type}$

tip a — čistih vrednosti

tip ma — izračuni, ki vnejo (ali pa ne) vrednost tipa a

- 2) return : $a \rightarrow ma$ pretvori vrednost v izračun
pure

- 3) bind : $ma \rightarrow (a \rightarrow mb) \rightarrow mb$



čista aplikacija

apply : $a \rightarrow (a \rightarrow b) \rightarrow b$

apply ($\frac{\pi}{2}$) sin = 1

Primer: nedeterminizem

računski učinek: možnih je več rezultatov
(končno mnogo)

- $\text{m a} = \underbrace{[a]}$ seznam a-jev = veliko a-jev
tip: seznam elementov tipa a
- $\text{pure} : a \rightarrow [a]$
 $\text{pure } x = [x]$ seznam z enim elementom

- $\text{bind} : [a] \rightarrow (a \rightarrow [b]) \rightarrow [b]$

$$\text{bind } xs \quad f \quad = \text{concat} (\text{map } f \text{ } xs)$$

$$\begin{array}{ccccc} [x_1, \dots, x_n] & & f x_i & & f x_j \\ & & \parallel & & \downarrow \\ & & [y_1, \dots, y_n] & & [z_1, \dots, z_m] \end{array}$$

Imamo še zakone monade:

$$\text{bind } c \text{ } f \quad \text{se piše tudi kot} \quad c \gg= f$$

$\uparrow$ izraz $\searrow$ fja iz čistih vrednosti v izrazu

① $(\text{return } x) \gg= f = f \ x$ leva enota

② $c \gg= \text{return} = c$ desna eno

③ $(c \gg= f) \gg= g = c \gg= (\lambda x. (f \ x \gg= g))$ asociativnost

"A monad is just a monoid in the category of endofunctors"

Primer:

① $[x] \gg= f = \text{concat} (\text{map } f \ [x]) = \text{concat} \ [f \ x] = f \ x$

② $c \gg= \text{return} = \text{concat} (\text{map } \text{return} \ c) = c \quad \checkmark$

$$c = [x_1, \dots, x_n]$$

$$\text{map } \text{return} \ c = [[x_1], \dots, [x_n]]$$

③ ✓

Intermezzo:

Java:

$$C + C \stackrel{?}{=} 2 * C$$

$$7 + 7 \quad \checkmark \quad 2 * 7$$

$$\text{obj.killChicken()} + \text{obj.killChicken()} \quad \times$$

$$2 * \text{obj.killChicken()}$$

$$C_1 + C_2 \stackrel{?}{=} C_2 + C_1$$

Output:

$$\text{bind} : A \times \text{String} \rightarrow (A \rightarrow B \times \text{String}) \rightarrow B \times \text{String}$$

$$\begin{array}{ccc} c & f & \\ (42, \text{"foo"}) & \underbrace{f \ 42}_{(13, \text{"bar"})} & (13, \text{"foobar"}) \end{array}$$

$$\cdot \text{bind } C_1 (\lambda x. C_2)$$

$$\cdot C_1 \gg (\lambda x. C_2)$$

$$\cdot \text{do } x \leftarrow C_1 \\ C_2$$

$$C_1 \gg (\lambda x_1. C_2 \gg (\lambda x_2. \dots))$$

$$\text{do } x_1 \leftarrow C_1 \\ x_2 \leftarrow C_2 \\ \dots$$

$$\overline{C} \leftarrow C$$