

Ime in priimek

--	--	--	--	--	--	--	--

Vpisna številka

1	
2	
3	
Σ	

NAVODILA

- **Ne odpirajte te pole**, dokler ne dobite dovoljenja.
- **Preden začnete reševati test:**
 - Vpišite svoje podatke na testno polo z velikimi tiskanimi črkami.
 - Na vidno mesto položite osebni dokument s sliko in študentsko izkaznico.
 - Preverite, da imate mobilni telefon izklopljen in spravljen v torbi.
 - Prijavite se na spletno učilnico, kamor boste oddajali nekatere odgovore.
- Dovoljeni pripomočki: pisalo, brisalo, USB ključ, in poljubno pisno gradivo.
- Rešitve vpisujte v polo ali jih oddajte preko spletne učilnice. Pri odgovorih, ki ste jih oddali preko spletne učilnice, na izpitno nalogo napišite "glej spletno učilnico – datoteka <ime_datoteke>".
- Če kaj potrebujete, prosite asistenta, ne sosedov.
- **Med izpitom ne zapuščajte svojega mesta** brez dovoljenja.
- Testna pola vam bo odvzeta **brez nadaljnjih opozoril**, če:
 - komunicirate s komerkoli, razen z asistentom,
 - komu podate kak predmet ali list papirja,
 - odrinete svoje gradivo, da ga lahko vidi kdo drug,
 - na kak drug način prepisujete ali pomagате komu prepisovati,
 - imate na vidnem mestu mobilni telefon ali druge elektronske naprave.
- **Ob koncu izpita:**
 - Ko asistent razglasi konec izpita, **takoj** nehajte in zaprite testno polo.
 - **Ne vstajajte**, ampak počakajte, da asistent pobere **vse** testne pole.
 - **Testno polo morate nujno oddati.**
- Čas pisanja je 120 minut. Na vidnem mestu je zapisano, do kdaj imate čas.
- Predvideni ocenjevalni kriterij:
 1. ≥ 90 točk, ocena 10
 2. ≥ 80 točk, ocena 9
 3. ≥ 70 točk, ocena 8
 4. ≥ 60 točk, ocena 7
 5. ≥ 50 točk, ocena 6

Veliko uspeha!

1. naloga (35 točk)

a) (7 točk) Ker so v Elboniji počitnice, tokrat ne bo naloge iz elbonske slovnice. Definirajmo slovnico Boolovih izrazov, ki jih lahko sestavimo iz spremenljivk, konstant `true` in `false`, disjunkcije in konjunkcije:

$$\begin{aligned}\langle \text{izraz} \rangle &::= \langle \text{konjunktivni} \rangle \mid \langle \text{konjunktivni} \rangle \text{ or } \langle \text{izraz} \rangle \\ \langle \text{konjunktivni} \rangle &::= \langle \text{osnovni} \rangle \mid \langle \text{konjunktivni} \rangle \text{ and } \langle \text{osnovni} \rangle \\ \langle \text{osnovni} \rangle &::= \text{true} \mid \text{false} \mid \langle \text{spremenljivka} \rangle \mid (\langle \text{izraz} \rangle) \\ \langle \text{spremenljivka} \rangle &::= [\text{A-Z}]^+\end{aligned}$$

Narišite sintaktično drevo, ki ustreza izrazu `X and Y and Z or true or false`.

b) (7 točk) Za zgornje boolove izraze definiramo semantiko velikih korakov:

$$\begin{array}{c} \frac{}{\text{true} \hookrightarrow \text{true}} \qquad \frac{}{\text{false} \hookrightarrow \text{false}} \qquad \frac{P \hookrightarrow \text{false}}{P \text{ and } Q \hookrightarrow \text{false}} \qquad \frac{P \hookrightarrow \text{true} \quad Q \hookrightarrow B}{P \text{ and } Q \hookrightarrow B} \\[10pt] \frac{P \hookrightarrow \text{true}}{P \text{ or } Q \hookrightarrow \text{true}} \qquad \frac{P \hookrightarrow \text{false} \quad Q \hookrightarrow B}{P \text{ or } Q \hookrightarrow B} \end{array}$$

Klemen je v prologu zapisal prva štiri pravila takole:

```
eval(true, true).
eval(false, false).
eval(and(P, Q), B) :- eval(P, true), eval(Q, B).
eval(and(P, _Q), false) :- eval(P, false).
```

Dopolnite program in zapišite še preostali dve pravili:

c) (7 točk) Andreja je zanimalo, ali lahko nastavi vrednost spremenljivke `P` tako, da se bo izraz `P or true and P` evalviral v `false`. Zapišite poizvedbo v prologu, s katero ugotovite, ali je to možno in za katero vrednost `P`:

d) (7 točk) V funkcijskem programskem jeziku s parametričnim polimorfizmom izračunajte glavni tip izraza

```
fun h x y -> x :: (y :: h y)
```

Operator `::` stakne glavo in rep seznama. Njegov tip je $\alpha \rightarrow \alpha \text{ list} \rightarrow \alpha \text{ list}$.

e) (7 točk) V λ -računu predstavimo naravno število $n \in \mathbb{N}$ s Churchevim numeralom

$$\bar{n} = \lambda f . \lambda x . \underbrace{f(f(\cdots f x) \cdots)}_n$$

Pravimo, da λ -izraz e predstavlja funkcijo $h : \mathbb{N} \rightarrow \mathbb{N}$, če za vse $n \in \mathbb{N}$ velja $e \bar{n} = \overline{h(n)}$. Na primer, izraz $\lambda n . \lambda f . \lambda x . f (f (n f x))$ predstavlja funkcijo $n \mapsto n + 2$.

Katero funkcijo predstavlja izraz $\lambda n . \lambda f . \lambda x . (n (n f)) (f x)$?

2. naloga (40 točk)

Z $D(x, y)$ označimo največji skupni delitelj celih števil x in y , pri čemer vzamemo $D(0, 0) = 0$. Na primer, $D(-12, 8) = 4$ in $D(-7, -7) = 7$. Pri reševanju naloge si lahko pomagata z naslednjimi lastnostmi, ki veljajo za vse $x, y \in \mathbb{Z}$:

- $D(x, y) = D(y, x)$
- $D(x, y) = D(x - y, x)$
- $D(x, x) = |x|$

a) (30 točk) Dokazite *delno* pravilnost programa za $x, y \in \mathbb{Z}$:

```
{ x > 0 ∧ y > 0 }  
a := x ;  
b := y ;  
while not (a = b) do  
  if a > b then  
    a := a - b  
  else  
    b := b - a  
  end  
end  
{ a = D(x, y) }
```

Pišite čitljivo!

b) (10 točk) Utemeljite, zakaj se zanka `while`, pri navedenih pogojih vedno ustavi: podajte ustrezno količino, ki se zmanjša, vsakič ko se izvede telo zanke, a se ne more zmanjševati v nedogled.

3. naloga (30 točk)

Sezname običajno definiramo tako, da so vsi elementi istega tipa. Lahko pa bi imeli *mešane* sezname, ki vsebujejo elemente *dveh* tipov. V Haskellu bi tip definirali kot

```
data List2 a b = Nil | Cons1 a (List2 a b) | Cons2 b (List2 a b)
```

in v OCamlu kot

```
type ('a, 'b) list2 =  
  | Nil  
  | Cons1 of 'a * ('a, 'b) list2  
  | Cons2 of 'b * ('a, 'b) list2
```

V nadaljevanju bomo uporabili Haskell, lahko pa nalogo rešujete tudi v OCamlu. Na primer, mešani seznam celih števil in nizov [42, cow, 10, capibara, rabbit, 10] predstavimo kot

```
capibara = Cons1 42 (Cons2 "cow"  
  (Cons2 "capibara" (Cons2 "rabbit" (Cons1 10 Nil))))
```

a) (10 točk) Sestavite funkcijo

```
length2 :: List2 a b -> Int
```

ki vrne dolžino seznama. Primer:

```
> length2 capibara  
5
```

b) (10 točk) Sestavite funkcijo

```
map2 :: (a -> b) -> (c -> d) -> List2 a b -> List2 c d
```

ki sprejme funkciji f in g ter ju uporabi na elementih mešanega seznama. Primer (funkcija `toUpper` je v modulu `Data.Char`):

```
> map2 (\x -> x < 20) (map toUpper) capibara  
Cons1 False (Cons2 "COW" (Cons2 "CAPIBARA" (Cons2 "RABBIT" (Cons1 True Nil))))
```

c) (10 točk) Sestavite funkcijo

```
split :: List2 a b -> ([a], [b])
```

ki sprejme mešani seznam in ga razdeli na dva običajna seznama. Primeri:

```
> split (Cons1 "capibara" Nil)  
(["capibara"], [])  
> split (Cons2 "capibara" Nil)  
([], ["capibara"])  
> split capibara  
([42, 10], ["cow", "capibara", "rabbit"])
```