Programiranje I: 3. izpit

24. avgust 2022

Čas reševanja je 120 minut. Veliko uspeha!

1. naloga

a) Napišite funkcijo `zamenjaj : ('a * 'b) * ('c * 'd) -> ('a * 'c) * ('b * 'd)`, ki sprejme dva para in vrne par parov, kjer je druga komponenta prvega para zamenjana s prvo komponento drugega.

b) Napišite funkcijo `modus : int * int * int -> int option`, ki vrne modus (najpogostejšo vrednost) podane trojice, ali pa `None`, če so vse vrednosti različne.

c) Napišite funkcijo `uncons : 'a list -> ('a * 'a list) option`, ki seznam razdeli na potencialno glavo in rep.

d) Napišite funkcijo `vstavljaj : 'a -> 'a list -> 'a list`, ki vstavi prvi argument med vse ostale elemente seznama.

```
# vstavljaj 0 [1;2;3;4]
- : int list
[1;0;2;0;3;0;4]
```

e) Napišite funkcijo `popolnoma_obrni : 'a list list -> 'a list list`, ki popolnoma obrne gnezden seznam (obrne vrstni red seznamov in elemente v podseznamih). Funkcija mora biti repno rekurzivna in ne sme uporabljati funkcij iz standardne knjižnice.

```
# popolnoma_obrni [[1;2;3]; [4;5]; [7;8;9]]
- : int list list = [[9; 8; 7]; [5; 4]; [3; 2; 1]]
```

2. naloga

Neprazen označen trak predstavimo s tipom `tape`, ukaze glavi, ki je nad označenim delom traku, pa s tipom `command`.

```
type 'a tape = Tape of { left : 'a list; head : 'a; right : 'a list }

type 'a command = Left | Do of ('a -> 'a) | Right

let example = Tape { left = [ 3; 2; 1 ]; head = 4; right = [ 5; 6 ] }
```

Spremenljivka `example` predstavlja spodnjo konfiguracijo

```
        V
1 2 3   4   5 6
```

kjer je glava nad poljem z vrednostjo 4 (in indeksom 3), levo od glave so elementi `3, 2, 1` in desno elementa `5, 6`.

a) Napišite funkcijo `map : 'a tape -> ('a -> 'b) -> 'b tape`, ki sprejme trak in funkcijo, ter preslika elemente v traku. Ob koncu mora biti glava na enakem mestu (čeprav se vrednost pod glavo lahko spremeni)

b) Napišite funkcijo `izvedi : 'a tape -> 'a command -> 'a tape option`, ki sprejme trak in ukaz ter vrne nov trak, kjer je glava ustrezno premaknjena glede na ukaz. Če je ukaz premik `Left` ali `Right`, se glava ustrezno premakne levo ali desno. Če je ukaz `Do`‘, potem aplicira funkcijo na element pod glavo, glava pa ostane na enakem mestu. Če je premik neveljaven (npr. premik prek roba traku), potem funkcija vrne `None`.

```
# izvedi example Left
- : tape option
Some (Tape { left = [ 2; 1 ]; head = 3; right = [ 4; 5; 6 ] })
```

c) Napišite funkcijo `izvedi_ukaze : 'a tape -> 'a command list -> 'a tape`, ki sprejme trak in seznam ukazov in vrne nov posodobljen trak. Posodabljanje traku premika glavo glede na ukaze (levo ali desno) ali pa aplicira funkcijo podano v `Do` ukazu. Če je kak ukaz neveljaven naj funkcija preneha z izvajanjem in vrne kar je bilo spremenjeno do sedaj.

```
# izvedi_ukaze example [Do ((+) 1); Left; Do ((+) 4); Right; Do ((+) 1); Right]
- : int tape =
Tape {left = [6; 7; 2; 1]; head = 5; right = [6]}
```

d) Napišite funkcijo
`naberi_in_pretvori : 'a tape -> 'a command list -> ('a * 'a) list * 'a tape`, ki sprejme trak in seznam ukazov. Funkcija ukaze izvaja enega za drugim, kjer najprej izvede ukaz, in nato v seznam parov nabira vrednosti, **ki so se pojavile pod ukazom** `Do`. Poleg seznama nabranih parov naj funkcija vrne tudi končno stanje traku. Če tekom izvajanja pride do napake, naj funkcije vrne vrednosti nabrane do sedaj in zadnje veljavno stanje traku.

```
# naberi example [Do ((+) 1); Left; Do (( * ) 2); Right; Right; Right; Do ((-) 1);
Right; Do (fun x -> x / 2 )]
- : (int * int) list * int tape =
([(4, 5); (3, 6); (6, -5)],
Tape {left = [5; 5; 6; 2; 1]; head = -5; right = []})
```

e) Napišite funkcijo `pripravi_ukaze: 'a tape -> ('a -> 'a) -> ('a -> 'a) command list`, ki sprejme trak in funkcijo in vrne tak seznam ukazov, da za pravilno funkcijo `map` velja `pripravi_ukaze tape f = lst => map tape f == izvedi_ukaze tape lst`.

3. naloga

Nalogo lahko rešujete v Pythonu ali OCamlu.

Miha bo izdal svojo kolekcijo NeZamenljivih Žetonov (NZŽ-jev), ki bodo upodabljali abstraktne slike. Vsak žeton predstavlja lastništvo slike, ki za osnovo vzame bel pas dolžine N pikslov. Miha lahko na ta pas drugega za drugim riše vzorce celoštevilske dolžine vsaj 1 in največ k pikslov, kjer je med dvema zaporednima vzorcema razdalja vsaj l pikslov.

Ker mu programiranje ne gre, ga zanima, koliko različnih NZŽ-jev lahko naredi za podano dolžino pasu N ter parametra k in l. Napišite funkcijo `stevilo_nzz : int -> int -> int -> int` ali `def nzz(n: int, k: int, l: int) -> int`, ki sprejme dolžino pasu N in parametra k in l ter vrne število različnih slik, ki jih Miha lahko naredi.

Spodaj je narisanih vseh 11 možnih slik dolžine 4 in parametroma $k = 3$ in $l = 2$.

```
_ _ _ _

X _ _ _
_ X _ _
_ _ X _
_ _ _ X

X X _ _
_ X X _
_ _ X X

X X X _
_ X X X

X _ _ X
```