Programiranje I: mini izpit

26. november 2018

Čas reševanja je 60 minut. Veliko uspeha!

V nalogah si lahko pomagate z rešitvami prejšnjih nalog.

1. naloga

Napišite *repno rekurzivno* funkcijo, ki sprejme seznam celih števil in izračuna njihovo vsoto. Na primer za seznam `[0; 1; 2; 3; 4]` naj funkcija vrne `10`, v primeru seznama vseh naravnih števil do milijon (pridobimo z izračunom `List.init 1000001 (fun x -> x)`) pa naj vrne `500000500000`

2. naloga

Napišite funkcijo, ki preveri, ali je dani seznam urejen naraščajoče. Pri tem smatramo, da je seznam urejen naraščajoče, če je prvi element seznama manjši kot drugi element, in je podseznam, ki ne vsebuje prvega elementa, prav tako urejen. Na primer `[0; 1; 1; 42]` in `[-1]` sta urejena, `[2; -2]` pa ne.

3. naloga

Napišite funkcijo, ki vstavi celo število v urejen seznam celih števil. Na primer vstavljanje `4` v `[0; 1; 1; 42]` vrne `[0; 1; 1; 4; 42]`.

Če na tak način zaporedno vstavljamo elemente v prazen seznam, je rezultat prav tako urejen seznam. S pomočjo tega dejstva napišite funkcijo, ki sprejme seznam celih števil in vrne urejen seznam, ki vsebuje enaka števila.

4. naloga

Urejanje z vstavljanjem, ki smo ga definirali v prejšnji nalogi, ni odvisno od dejstva, da je vhod seznam celih števil. Napišite novo funkcijo za urejanje, ki kot argument poleg seznama elementov dobi tudi funkcijo `cmp`. Funkcija `cmp` sprejme `x` in `y` in pove, ali je `x` manjši kot `y`. Na primer funkcija `fun j k -> not (j < k)` obrne vrstni red ureditve celih števil. Če jo skupaj s seznamom `[0; 1; 1; 42]` podamo kot vhod naše funkcije za urejanje, nam ta vrne `[42; 1; 1; 0]`.

5. naloga

Želimo modelirati vkrcavanje potnikov na letalo. Vsak od potnikov je predstavljen kot element tipa

```
type flyer = { status : status ; name : string }
```

kjer je `status` lahko osebje *(staff)* ali navaden potnik *(passenger)*. Vsak navaden potnik ima v statusu dodatno zabeleženo prioriteto, ki je lahko vrhovna *(top)* ali pa spada v neko skupino *(group)* s celoštevilsko prioriteto.

Primer seznama potnikov je:

```
let flyers = [ {status = Staff; name = "Quinn"}
             ; {status = Passenger (Group 0); name = "Xiao"}
             ; {status = Passenger Top; name = "Jaina"}
             ; {status = Passenger (Group 1000); name = "Aleks"}
             ; {status = Passenger (Group 1000); name = "Robin"}
             ; {status = Staff; name = "Alan"}
             ]
```

Definirajte tip `priority` za predstavitev prioritet, in tip `status`.

6. naloga

Napišite funkcijo, ki uredi seznam potnikov v zaporedje za vkrcavanje tako, da je prvo na vrsti osebje, nato navadni potniki vrhovne prioritete, sledijo pa navadni potniki razporejeni padajoče glede na njihovo prioritetno skupino (znotraj skupin vrstni red ni pomemben). Zaporedje predstavimo s seznamom.

Primer zaporedja za vkrcavanje konstruiran iz zgornjega primera je:

```
[{status = Staff; name = "Quinn"};
 {status = Staff; name = "Alan"};
 {status = Passenger Top; name = "Jaina"};
 {status = Passenger (Group 1000); name = "Robin"};
 {status = Passenger (Group 1000); name = "Aleks"};
 {status = Passenger (Group 0); name = "Xiao"}]
```

7. naloga

Napišite funkcijo, ki sprejme seznam potnikov in ga razdeli v bloke, ki vsebujejo potnike enake prioritete (prioritete lahko primerjate z OCamlovo vgrajeno enakostjo =). Bloke vrnemo v obliki seznama seznamov, kjer naj bodo bloki urejeni glede na prioriteto.

Za zgornji primer torej dobimo:

```
[[ {status = Staff; name = "Alan"};
   {status = Staff; name = "Quinn"} ];
 [ {status = Passenger Top; name = "Jaina"} ];
 [ {status = Passenger (Group 1000); name = "Aleks"};
   {status = Passenger (Group 1000); name = "Robin"} ];
 [ {status = Passenger (Group 0); name = "Xiao"} ]]
```