**7.8 savarjiSoebi**

1. mocemulia siis [*f x | x ← xs,p x*] konstruqtori. rogor SeiZleba misi xelaxla warmodgena maRali rigis *map* da *filter* funqciebis gamoyenebiT?

2. gansazRvreT maRali rigis *all*, *any*, *takeWhile* da *dropWhile* funqciebi standartul prelude failSi mocemuli maTi aRwerebis gamouyeneblad.

3. xelaxla gansazRvreT *map f* da *filter p* funqciebi *foldr*-is gamoyenebiT.

4. *foldl* funqciis saSualebiT gansazRvreT *dec2int* :: [*Int*] *→ Int* funqcia, romelic gardasaxavs aTobiT ricxvs erT mTel ricxvad. magaliTad:

*> dec2int* [2*,* 3*,* 4*,* 5]

2345

5. axseniT, ratom aris Semdegi gansazRvreba dauSvebeli:

*sumsqreven* = *compose* [*sum, map* (*↑*2)*, filter even* ]

6. standartuli prelude failis gamouyeneblad gansazRvreT maRali rigis sabiblioTeko *curry* funqcia, romelic gardaqmnis funqcias wyvilebze karirebul funqciad da, piriqiT, gansazRvreT *uncurry* funqcia, romelic orargumentian karirebul funqcias aqcevs funqciad wyvilebze.

rCeva: upirveles yovlisa CawereT orive funqciis tipi.

7. maRali rigis *unfold* funqcia, romelic siis mosamzadeblad rekursiis martivi Sablonis inkafsulirebas axorcielebs, Semdegi saxiT SeiZleba ganisazRvros:

*unfold p h t x | p x* = []

*| otherwise* = *h x* : *unfold p h t* (*t x*)

amrigad, *unfold p h t* funqcia qmnis cariel sias, Tu *p* predikati WeSmaritia argumentisaTvis, xolo winaaRmdeg SemTxvevaSi qmnis aracariel sias *h* funqciis gamoyenebiT, romelic Tavs (ingl. *h*ead) iZleva, da *t* funqcias meore argumentis Sesaqmnelad, romelic rekursiulad muSavdeba imave gziT siis kudis (ingl. tail) sawarmoeblad. magaliTad, *int2bin* funqcia SeiZleba ufro kompaqturadac Caiwe­ros *unfold* funqciis gamoyenebiT Semdegnairad:

*int2bin* = *unfold* (*==* 0) (`*mod*`2) (`*div*`2)

gansazRvreT xelaxla *chop8*, *map f* da *iterate f* funqciebi *unfold* funqciis gamoyene­biT.

8. ganaxorcieleT striqonis gadamcemi programis modificireba gadacemis martivi Secdomebis gamosavlineblad luwobis bitebis gamoyenebiT. luwobis bitebis arsi aseTia. yoveli rvaTanrigiani orobiTi ricxvi kodirebis procesSi grZeldeba luwobis erTi bitiT, sadac Tavsdeba erTiani, Tu ricxvi Seicavs erTianebis kent raodenobas, xolo winaaRmdeg SemTxvevaSi ganuldeba. Tavis mxriv, yoveli cxraTanrigiani orobiTi ricxvi, romelic dekodirebisas gamoiyeneba, mowmdeba. Semowmebis arsi martivia: Tu orobiTi ricxvis pirvel rva TanrigSi (e.w. *baitSi*) aRmoCndeba, rom erTeulovani bitebis raodenoba luwia, xolo luwobis damatebiTi mecxre biti Seicavs erTians, es imas niSnavs, rom gaCnda Secdoma.

rCeva: sabiblioTeko *error* :: *String → a* funqcia amTavrebs gamoTvlas da ga­moaqvs ekranze mocemuli striqoni rogorc Setyobineba Secdomis Se­sa­xeb.

9. gamocadeT striqonis gadamcemi Tqveni axali programa wina savar­ji­So­dan, Tu gadamcemi iyenebs dazianebul sakomunikacio arxs, romelic ig­no­­rirebas ukeTebs pirvel bits. amis modelireba SesaZlebelia *tail* funq­ci­iT bitebis siebze.

pasuxebi:

**savarjiSo 1**

*map f* (*filter p xs*)

**savarjiSo 2**

*all p* = *and ◦ map p*

*any p* = *or ◦ map p*

*takeWhile \_*[] = []

*takeWhile p* (*x* : *xs*)

*| p x* = *x* : *takeWhile p xs*

*| otherwise* = []

*dropWhile \_*[] = []

*dropWhile p* (*x* : *xs*)

*| p x* = *dropWhile p xs*

*| otherwise* = *x* : *xs*

**savarjiSo 3**

*map f* = *foldr* (*λx xs → f x* : *xs*) []

*filter p* = *foldr* (*λx xs →* **if** *p x* **then** *x* : *xs* **else** *xs*) []

**savarjiSo 4**

*dec2nat* = *foldl* (*λx y →* 10 *∗ x* + *y*) 0

**savarjiSo 5**

yvela Semadgenel funqcias erTi da igive tipi ar aqvs. magaliTad:

*sum* :: [*Int*] *→ Int*

*map* (*↑*2) :: [*Int*] *→* [*Int*]

*filter even* :: [*Int* ] *→* [*Int*]

**let sums= sum.map(^2).filter even**

**savarjiSo 6**

*curry* :: ((*a, b*) *→ c*) *→* (*a → b → c*)

*curry f* = *λx y → f* (*x , y*)

*uncurry* :: (*a → b → c*) *→* ((*a, b*) *→ c*)

*uncurry f* = *λ*(*x , y*) *→ f x y*

**savarjiSo 7**

*chop8* = *unfold null* (*take* 8) (*drop* 8)

*map f* = *unfold null* (*f ◦ head*) *tail*

*iterate f* = *unfold* (*const False*) *id f*

**savarjiSo 8**

*encode* :: *String →* [Bit]

*encode* = *concat ◦ map* (*addparity ◦ make8 ◦ int2bin ◦ ord*)

*decode* :: [Bit] *→ String*

*decode* = *map* (*chr ◦ bin2int ◦ checkparity*) *◦ chop9*

*addparity* :: [*Bit*] *→* [*Bit*]

*addparity bs* = (*parity bs*) : *bs*

*parity* :: [Bit] *→ Bit*

*parity bs | odd* (*sum bs*) = 1

*| otherwise* = 0

*chop9* :: [Bit] *→* [[*Bit*]]

*chop9* [] = []

*chop9 bits* = *take* 9 *bits* : *chop9* (*drop* 9 *bits*)

*checkparity* :: [Bit] *→* [Bit]

*checkparity* (*b* : *bs*)

*| b ==* *parity bs* = *bs*

*| otherwise* = *error* "parity mismatch"[[1]](#footnote-1)

**savarjiSo 9**

amosaxsnelia mkiTxvelis mier!

1. ingl. *parity mismatch* \_ luwobis Seusabamoba [↑](#footnote-ref-1)