Programiranje II

2019/20

3. izpit, FAMNIT

Izpit rešujete posamično. Naloge so enakovredne. Pri reševanju je dovoljena uporaba literature. Čas pisanja izpita je 90 minut.

Veliko uspeha!

IME IN PRIIMEK:	
VPISNA ŠTEVILKA:	
ŠTUD. POGRAM:	
PODPIS:	

Naloga 1 (25%)

Dano imamo funkcijo `inc : int -> int`, s katero povečamo parameter funkcije za eno. Predpostavimo, da nimamo funkcije za seštevanje celih števil.

Implementiraj funkcijo

`is_sum int -> int -> int -> bool`,

ki preveri, če je tretji parameter vsota prvih dveh parametrov.

Primer:

```
is_sum 5 3 8  ==>  true
is_sum 10 1 8  ==>  false
```

Naloga 2 (25%)

DNA sekvence vsebujejo simbole `C`, `G`, `A`, in `T`.

a) Definiraj tip `dna_syn` z uporabo unije tipov. Predstavi simbole (`C`,`A`,`G`,`T`) s konstruktorji unije tipov.

b) Definiraj tip `dna_array` kot eno-dimenzionalno polje vrednosti tipa `dna_syn`.

c) Napiši funkcijo

```
longest_subseq : dna_array -> dna_syn -> int,
```

ki vrne dolžino najdaljše sekvence ponovitve simbola podanega kot drugi parameter v sekvenci, ki je podana kot prvi parameter funkcije.

Primer:

```
longest_subseq [C;G;C;A;A;T;C;C;A;A;A;T] A ==> 3
```

Naloga 3 (25%)

Binarno drevo je `btree` je definirano na sledeč način.

```
type 'a node = {left: 'a btree; key: int; right: 'a btree}
and  'a btree = Nil | Node of 'a node ;;
```

a) Kreiraj binarno drevo s korenom, ki vsebuje ključ 7, levim otrokom s ključem 3 in desnim otrokom s ključem 5. Drevo nima drugih vozlišč.

b) Napiši funkcijo

`is_heap 'a btree -> bool`,

ki preveri ali je drevo podano kot prvi parameter funkcije kopica. V kopici velja za poljubno vozlišče, da so ključi iz poddrevesa vedno večji od ključa danega vozlišča.

Naloga 4 (25%)

Definiraj modul KVS za upravljanje shrambe parov sestavljenih iz ključa in vrednosti. Ključi so tipa `string` in vrednosti so tipa `'a`.

a) Definiraj tip KVS.t kot seznam parov sestavljenih iz ključa in vrednosti.

b) Definiraj **strukturo** (implementacijo) modula KVS, ki naj vsebuje naslednji funkciji.

`add : KVS.t -> string*'a -> unit`, in
`get : KVS.t -> string -> 'a`.

c) Definiraj **signaturo** (vmesnik) modula KVS, ki skrije definicijo abstraktnega podatkovnega tipa KVS in omogoča dostop le do funkcije `get`.