Programiranje I: 1. izpit

25. januar 2024

Čas reševanja je 120 minut. Veliko uspeha!

1. naloga

a) Napišite funkcijo `je_sodo : int -> bool`, ki preveri, ali je število sodo.

b) Napišite funkcijo `seznam_sodih : int list -> int list`, in vrne nov seznam, ki vsebuje samo soda števila.

c) Definirajte tip `oznaceno`, ki ima dve varianti, eno s konstruktorjem `Sodo` in drugo s konstruktorjem `Liho`. Napišite še funkcijo `oznaci : int list -> oznaceno list`, ki sprejme seznam števil in vrne nov seznam, kjer so števila označena z ustreznimi konstruktorji.

```
# oznaci [1; 2; 3; 4; 5; 6; 7; 8; 9; 10];;
- : oznaceno list =
[Liho 1; Sodo 2; Liho 3; Sodo 4; Liho 5; Sodo 6; Liho 7; Sodo 8; Liho 9; Sodo 10]
```

d) Definirajte funkcijo `vsoti_kvadratov : oznaceno list -> int * int`, ki sprejme seznam označenih števil in vrne par, ki vsebuje vsoto kvadratov lihih števil in vsoto kvadratov sodih števil. Za vse točke naj bo funkcija *repno rekurzivna*.

2. naloga

Če predstavimo slovarje z asociativnimi seznami

```
type 'a dict = (string * 'a) list
```

lahko podatke v zapisu JSON predstavimo s tipom `json`, podanim z

```
type primitive = Bool of bool | Int of int | String of string | Null
type json = Primitive of primitive | Object of json dict | Array of json list
```

Na primer objekt (tako v JSON-u pravimo slovarjem)

```
{
  "name": "Matija",
  "age": 20,
  "friends": [1, 2, "Nemo"],
  "is_student": true,
  "is_professor": false,
  "is_ta": null
}
```

bi predstavili z vrednostjo

```
let json_example =
  Object
    [
      ("name", Primitive (String "Matija"));
      ("age", Primitive (Int 20));
      ("friends", Array [
        Primitive (Int 1);
        Primitive (Int 2);
        Primitive (String "Nemo")
      ]);
      ("is_student", Primitive (Bool true));
      ("is_professor", Primitive (Bool false));
      ("is_ta", Primitive Null);
    ]
```

a) Napišite funkcijo `prestej_stevila : json -> int`, ki prešteje število vseh celih števil v danem JSON-u.

b) Napišite funkcijo `izloci_nize : int -> json -> string list`, ki vrne seznam vseh nizov (v poljubnem vrstnem redu), ki se nahajajo vsaj `globina` globoko (gnezdeno v objektih ali seznamih).

```
# izloci_nize 2 json_example;;
- : string list =
["Nemo"]
# izloci_nize 1 json_example;;
- : string list =
["Matija"; "Nemo"]
```

c) Napišite funkcijo `dodaj_predpono : string -> json -> json`, ki vsem ključem v vseh objektih doda podano predpono.

d) Napišite funkcijo `izpisi : json -> unit`, ki ***izpiše*** JSON na berljiv način (ni treba, da točno tako, kot v zgornjem primeru). Lahko predpostavite, da so vsi nizi vsebujejo samo alfanumerične znake.

e) ChatGPT za navodilo

> Napišite funkcijo `je_konsistenten : json -> bool`, ki preveri, da za vsak seznam v JSON-u velja sledeče: elementi so zgolj primitivni elementi in vsi elementi imajo enak tip.

predlaga rešitev:

```
let rec same_type_elements lst =
  match lst with
  | [] -> true
  | [x] -> true
  | hd :: tl ->
      List.for_all (fun el -> (type_of_primitive hd) = (type_of_primitive el)) tl

(* Helper function to get the type of a primitive value *)
and type_of_primitive = function
  | Bool _ -> "bool"
  | Int _ -> "int"
  | String _ -> "string"
  | Null -> "null"

(* Main function to check consistency of an object *)
let rec je_konsistenten objekt =
  match objekt with
  | Primitive _ -> true (* Primitives are always consistent *)
  | Object dict -> List.for_all (fun (_, v) -> je_konsistenten v) dict
  | Array lst -> same_type_elements lst
```

Popravite rešitev in v komentarju pojasnite napake, ki jih je ChatGPT naredil.

3. naloga

Nalogo lahko rešujete v Pythonu ali OCamlu.

Taja se poroči in na šrangi mora pokazati (vseeno smo enakopravna družba), da lahko poskrbi za svojo novo družino. Pred njo stoji podrt mlaj (smreka), ki ga mora prerezati na koščke tako, da bodo skupaj dosegli kar največjo vrednost glede na dani cenik, vendar lahko opravi največ `k` rezov. Ker so jo šrangarji pri tem opravilu že videli, ker nočejo, da si umaže obleko, in ker ni dobro preveč nadlegovati neveste z motorko v rokah, je dovolj, da nadležnim obiskovalcem samo pove, na katerih mestih bi zarezala.

Napišite funkcijo `sranga : int list -> int -> int -> int`, ki sprejme cenilni seznam `c`, dolžino mlaja `m` in maksimalno število rezov `k`, ki jih lahko opravi. Funkcija naj vrne največjo vrednost, ki jo lahko pridobi. Velja, da je na `i`-tem mestu seznama `c` cena, ki bi jo dobili za kos dolžine `i - 1` (kos dolžine 0 pač ni smiselen). Prav tako Taja ne more odrezati kosa, katerega cene ni v seznamu `c`. Lahko predpostavite, da cene v `c` niso negativne in vedno strogo naraščajo, ter da je za dane podatke vedno mogoče najti rešitev.

Za mlaj dolžine 8 in cenilni sistem `[3;5;8;9;10;17;17;20]`, kjer lahko naredi največ 3 reze, lahko celoten mlaj proda za 23 enot.