Programiranje I: 2. izpit

06. July 2018

Čas reševanja je 150 minut. Veliko uspeha!

1. naloga

a) Write a function

```
apply : ('a -> 'b) -> 'a -> 'b
```

that applies a function to its argument.

b) Write a function

```
revapply : 'a -> ('a -> 'b) -> 'b
```

that performs reverse application. For instance, `revapply (revapply x f) g` should be equivalent to $g(f(x))$.

c) Write a function

```
take : int -> 'a list -> ('a list) option
```

such that `take n xs` returns the first `n` elements of `xs`, or `None` if `xs` has fewer than `n` elements.
For full credit, the function should be *tail-recursive*.

2. naloga

The type `'a list` is the collection of 0 or more elements of type `'a`. We can modify this idea to model collections of 1 or more elements of type `'a` by introducing a new type of non-empty lists.

a) Define a new type `'a nelist` of non-empty lists.

b) Define a function `head : 'a nelist -> 'a`.

c) Define a function `length : 'a nelist -> int` that computes the length of a non-empty list.

d) Define a function `list_of_nelist : 'a nelist -> 'a list`.

e) Define a function `fold : ('a -> 'b -> 'b) -> 'b -> 'a nelist -> 'b`.

3. naloga

Dr Hannah Habibah is a mathematician with a great love for symmetries. After returning from her recent trip to Hajjah, Yemen, she is looking for symmetries in her holiday photos.

The photos are represented as lists of bit-strings. Here's a sample 8x8 picture:

```
["00101011",
 "01001100",
 "11000111",
 "01100111",
 "01110110",
 "00100111",
 "01010001",
 "01001000"]
```

You can generate random photos with the following line of code:

```
Python:
import random
m = ["".join( [ str(random.randint(0, 1)) for i in range(0, 8) ] )
          for j in range(0, 8)]

OCaml:
let m = List.init 8 (fun _ -> List.fold_left (^) ""
               (List.init 8 (fun _ -> string_of_int (Random.int 2)))) ;;
```

Dr Habibah wants to divide each row into blocks such that each block is symmetric, in a suitable way. Her goal is to find the least number of blocks for each row.

There are different kinds of symmetries she is considering:

- a block is *p-symmetric* if it is a palindrome
- a block B of length `n` is *sum-symmetric* if the sum of the first `int(n/2)`[1] bits of B and the last `int(n/2)` bits of B are equal

a) Define a boolean predicate `is_palindrome` that checks if a block is p-symmetric.

```
# Example:
>>> is_palindrome("01010")
True
```

b) Write a function `number_of_blocks` that computes the least number of blocks that a row has to be split into so that each block is symmetric.

```
# Example:
>>> number_of_blocks(m[0])
3
```

c) Write a function `blocks` that not only returns the minimum number of blocks for a row, but also indicates how to split the row into those blocks. There may be several ways to split a row into k blocks; it suffices to indicate one way of obtaining k blocks.

```
# Example:
>>> [blocks(l) for l in m][0]
(3, ['0', '01010', '11'])
```

[1] OCaml: `int_of_float ((float_of_int n) /. 2.)`

d) Write a boolean predicate `sum_symmetric`.
Hint: to convert a bit-string *b* to a list of integers, write `l = [int(c) for c in b]`[2]

```
# Example:
>>> sum_symmetric(m[-1])
True
>>> sum_symmetric('1011')
False
```

e) Generalise your functions `number_of_blocks` and `blocks` to take a boolean predicate `is_symmetric` as additional argument.

```
# Example:
>>> [blocks(l, sum_symmetric) for l in m][0]
(2, ['00', '101011'])
```

[2]OCaml: Load the "Str" module with `#load "str.cma" ;;`
then use `l = List.map int_of_string (Str.split (Str.regexp "") b)`