Programming Paradigms Fall 2022 — Problem Sets

by Nikolai Kudasov

November 10, 2022

1 Problem set №11

Consider the following knowledge base:

1. Implement the following predicates on lists:

   (a) Implement predicate `subseq/2` that checks whether the first list is a subsequence of the second list. The predicate should allow to use variables in the first argument and for elements in the second argument:

   ```
   ?- subseq([2,4], [1,2,3,4,5])
   true

   ?- L=[1,X,Y], subseq([1,2], L)
   L = [1, 2, _1656]
   L = [1, _1322, 2]
   L = [1, 1, 2]

   ?- sublist(X, [1,2,3])
   X = []
   X = [1]
   X = [1, 2]
   X = [1, 2, 3]
   X = [1, 3]
   X = [2]
   X = [2, 3]
   X = [3]
   ```

   (b) Implement predicate `search/3`, such that `search(Needle, Haystack, Position)` is `true` when `Needle` occurs as a sublist in `Haystack` exactly at position `Position`:

   ```
   ?- search([a,b,a], [c,a,b,a,b,a,d], Pos)
   Pos = 1
   Pos = 3

   ?- search(Needle, [c,a,b,a,b,a,d], 5)
   Needle = []
   Needle = [a]
   Needle = [a, d]

   ?- Needle = [a,_,a], search(Needle, [a,b,r,a,c,a,d,a,b,r,a], Pos)
   Needle = [a, c, a], Pos = 3
   Needle = [a, d, a], Pos = 5
   ```

(c) Implement predicate `replace/4`, such that `replace(Old, New, OldWhole, NewWhole)` is `true` when `NewWhole` can be produced from `OldWhole` by replacing zero or more occurrences of `Old` with `New`:

```
?- replace([a,b], [x,y,z], [a,b,r,a,b,a], L)
L = [x, y, z, r, x, y, z, a]
L = [x, y, z, r, a, b, a]
L = [a, b, r, x, y, z, a]
L = [a, b, r, a, b, a]

?- replace([a,a], [x,y], [a,a,a,a], L)
L = [x, y, x, y]
L = [x, y, a, a]
L = [a, x, y, a]
L = [a, a, x, y]
L = [a, a, a, a]
```

(d) Implement a predicate `suffix/2` that checks whether one list is a suffix of another list:

```
?- suffix([a,b,a], [c,a,a,b,a])
true

?- suffix(X, [c,a,a,b,a])
X = [c, a, a, b, a]
X = [a, a, b, a]
X = [a, b, a]
X = [b, a]
X = [a]
X = []
```

(e) Implement a predicate `repeat/2` to check if a list consists of the same element repeating:

```
?- repeat(1, [1,1,1])
true

?- repeat(X, [1,1,1])
X = 1

?- repeat(1, L)
L = []
L = [1]
L = [1, 1]
L = [1, 1, 1]
...
```

2. Implement the following predicates on lists of numbers:

   (a) Implement predicate `allLEQ/2` that checks that a given number is less than or equal (=<) to all elements in a given list:

   ```
   ?- allLEQ(1, [2,4,3,7])
   true
   ?- allLEQ(1, [2,4,3,0])
   false
   ```

   (b) Implement predicate `minimum/2` that checks that a given number is the minimum of a given list:

   ```
   ?- minimum(X, [2,4,3,7])
   X = 2
   ```

   (c) Implement predicate `partition/4` such that `partition(Pivot, List, Less, Greater)` is true when `Less` (`Greater`) contains all elements of `List` that are less than (resp. greater than) `Pivot`.

   ```
   ?- partition(3, [1,2,3,4,5,6,7], Less, Greater)
   Greater = [4, 5, 6, 7],
   Less = [1, 2]
   ```

   (d) Implement predicate `median/2` that checks if a given number is the median of a given list. That is there are exactly as many elements smaller than it as there are elements that are greater than it.

   ```
   ?- median(X, [1,2,3,4,5])
   X = 3
   ```

3. Implement the following predicates on binary numbers, represented as lists of `1`s and `0`s:

   (a) Implement a predicate `increment/2` that checks whether second binary number is an increment of the first one. The predicate should work both ways:

   ```
   ?- increment([1,1,1,1],X)
   X = [1,0,0,0,0]

   ?- increment([1,1,1,0],X)
   X = [1,1,1,1]

   ?- increment(X, [1,1,1,0])
   X = [1,1,0,1]
   ```

   (b) Implement a predicate `countTrailingZeros/2` that allows to count trailing zeros of a binary number:

   ```
   ?- countTrailingZeros([1,1,0,1,0,0],N)
   N = 2
   ```

4. Implement predicate `fib/2` such that `fib(X, Y)` is `true` if `X` and `Y` are consecutive Fibonacci numbers. Using `fib/2`, implement `fib/1` that can generate all Fibonacci numbers:

   ```
   ?- fib2(1,1)
   true

   ?- fib2(5,8)
   true

   ?- fib(Fib)
   Fib = 1
   Fib = 1
   Fib = 2
   ```

```
Fib = 3
Fib = 5
Fib = 8
Fib = 13
Fib = 21
Fib = 34
Fib = 55
Fib = 89
```