TÖL212M Rökstudd Forritun - Hópverkefni 8

Andri Fannar Kristjánsson

10. mars 2025

Hópverkefni 8

1

Sækið skrána `H8-skeleton.dfy` og vistið hana hjá ykkur en breytið nafni hennar í H8.dfy. Klárið að forrita föllin í skránni. Þetta er afbrigði af quickselect sem C.A.R. Hoare fann upp um leið og hann þróaði quicksort.

1.1 Svar:

Hér fyrir neðan má sjá kóðann þar sem föllin hafa verið forrituð. Dafny samþykkir þessa útgáfu. Einnig er hægt að skoða kóðann á þessari slóð: https://tinyurl.com/yntnzx4e.

```
// Author of question:  Snorri Agnarsson, snorri@hi.is

// Author of solution:       Andri Fannar Kristjánsson, afk6@hi.is
// Permalink of solution:    https://tinyurl.com/yntnzx4e

// Finish programminng the two methods

method Partition( m: multiset<int> )
  returns( pre: multiset<int>, p: int, post: multiset<int> )
  decreases |m|
  requires |m| > 0
  ensures p in m
  ensures m == pre+multiset{p}+post
  ensures forall z | z in pre :: z <= p
  ensures forall z | z in post :: z >= p
{
  // The body is missing.
  // You can use a loop or recursion.

  // Remove one value from m.
  var x :| x in m;
  var m' := m-multiset{x};

  // If m' is then empty, we're done and return x,
  // and the empty sets on both sides (m')
  if (m' == multiset{}) { return m', x, m'; }

  // If m' is not empty, then we recursively partition m'.
  pre, p, post := Partition(m');

  // Here we need to remind Dafny that m is the same as m' + multiset{x}.
  assert m == m' + multiset{x};

  // If x is less than or equal to the pivot from that partition,
  // we add x to the left partition and put p as the new pivot.
  if (x <= p) { return pre+multiset{x}, p, post; }

  // If x is greater than the pivot from the partition,
  // we add x to the right partition.
  else { return pre, p, post+multiset{x}; }
}

method QuickSelect( m: multiset<int>, k: int )
  returns( pre: multiset<int>, kth: int, post: multiset<int> )
  decreases |m|
  requires 0 <= k < |m|
  ensures kth in m
  ensures m == pre+multiset{kth}+post
  ensures |pre| == k
  ensures forall z | z in pre :: z <= kth
  ensures forall z | z in post :: z >= kth
{
  // The body is missing.
  // You can use a loop or recursion.
  // Use the Partition method as a helper method.

  // We partition the multiset m.
  var p, piv, r := Partition(m);

  // If the size of the left partition is equal to k, we're done.
  if ( |p| == k ) { return p, piv, r; }

  // If the size of the left partition is less than k,
  // we recursively partition the right partition.
  else if ( |p| < k ) {
    var p2, newpiv, r2 := QuickSelect(r, k-|p|-1);
    return p + multiset{piv} + p2, newpiv, r2;
  }

  // If the size of the left partition is greater than k,
  // we recursively partition the left partition.
  else {
    var p2, newpiv, r2 := QuickSelect(p, k);
    return p2, newpiv, r + multiset{piv} + r2;
  }
}
```