TÖL212M Rökstudd Forritun - Einstaklingsverkefni 1

Andri Fannar Kristjánsson

20. janúar 2025

Einstaklingsverkefni 1

1

Notið Dafny til að sanna formúluna

$$\sum_{k=1}^{n}(2k-1=n^2)$$

þ.e.

$$1+3+5+\cdots+(2n-1)=n^2$$

með því að klára forritstextann í skránni `SumOdds-skeleton.dfy` sem þið finnið í Canvas. Þið munið þá hafa sannað formúluna á þrjá vegu, sem allir eru eitthvert afbrigði af vélrænni þrepasönnun. Til hliðsjónar er ráðlegt að lesa glærur viku 1 (einnig í Canvas) þar sem sannað er að

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

þ.e.

$$1+2+3+\cdots+n=\frac{n(n+1)}{2}$$

1.1 Svar:

Hér fyrir neðan má sjá leystu útgáfuna sem Dafny samþykkir. Hægt er einnig að sjá kóðann hér: https://shorturl.at/JV4nK.

```
function SumOdds( n: int ): int
  requires n >= 0
{
  if n == 0 then
    0
  else
    SumOdds(n-1) + 2*n-1
}

lemma ProveSumOdds( n: int )
  requires n >= 0
  decreases n
  // Not sure if we were allowed to add to the conditions
  // but after trying a lot of things I never was able to prove to
  // Dafny that SumOdds(n-1) == (n-1)*(n-1); without adding the
  // ensures condition.
  ensures SumOdds(n) == n*n
```

```
{
  // Base case
  if n == 0
  {
    assert SumOdds(0) == 0;
    assert 0 == 0*0;
    return;
  }
  else
  {
    // Prove recursively
    ProveSumOdds(n-1);
    assert SumOdds(n) == SumOdds(n-1) + 2*n-1;
    assert SumOdds(n-1) == (n-1)*(n-1);
    assert SumOdds(n) == (n-1)*(n-1) + 2*n-1;
    assert SumOdds(n) == n*n;
  }
}

method ComputeSumOddsLoop( n: int ) returns (s: int)
  requires n >= 0
  ensures s == SumOdds(n)
  ensures s == n*n
{
  // Start from 0
  var i := 0;
  s := 0;

  while i != n
    // Loop invariants to make sure the sum is correct
    invariant 0 <= i <= n
    invariant s == SumOdds(i)
    invariant s == i*i
  {
    // Increase i and calculate the sum
    i := i+1;
    s := s + 2*i-1;
  }

  return s;
}

method ComputeSumOddsRecursive( n: int ) returns (s: int)
  requires n >= 0
  ensures s == SumOdds(n)
  ensures s == n*n
{
  // Base case
  if n == 0 {return 0;}

  // Call the function recursively
  s := ComputeSumOddsRecursive(n-1);

  // Add the current term
  s := s + 2*n-1;
  return s;
}
```

```
lemma SumOddsAll()
  ensures forall n | n >= 0 :: SumOdds(n) == n*n
{
  forall n | n >= 0
  {
    ProveSumOdds(n);
  }
}
```