CMPT 477 Program Assignment 3 description

Bowei Pan

301435285

This Dafny program uses the Dafny plug in to verify the problems. Below is a brief explanation of the program's code.

Design choices:

1. The Find method is used to find the index of a target value v in array a. If found, it returns the index of the value; if not found, it returns -1.

   For this problem, I added two loop invariants to the while loop:

   a) invariant 0 <= i <= a.Length

      i. This is used to ensure that the index does not exceed the boundary and cause error

   b) invariant forall j :: 0 <= j < i ==> a[j] != v

      i. Ensures that the target value v can only be in the unprocessed part [i, a.Length) .

2. This algorithm calculates the sum of the series 10n+10(n−1)+⋯+1010n + 10(n-1) + \dots + 1010n+10(n−1)+⋯+10.

   For this problem, I added two loop invariants for both inner and outer loop:

a) Outer loop: invariant 0 <= i <= n

   i. Make sure i will not out of bound

b) Outer loop: invariant sum == 5 * (n * (n + 1) - i * (i + 1))

   i. The sum in the end is 5 * n * (n + 1), the value of the unprocessed part is 5 * i * (i + 1), the difference is the current sum.

c) Inner loop: invariant 0 <= j <= i

   i. Make sure j will not out of bound.

d) Inner loop: invariant k == 10 * (i - j)

   i. Make sure k for each iteration is equal to 10 * (i - j), in the end k is equal to 10i.

3. The purpose of this method is to find the smallest element in a given array. First, set the first element to min, then traverse the array one by one starting from index 1, compare each element with the current min, and update min if a smaller element is found. The method ensures that the returned min is the minimum value of the array and exists in the array. The loop invariant ensures that after each iteration, min is the minimum value of the traversed part and the index does not exceed bound

4. First, define the array True as the front side and False as the back side. Then check the size of the Boolean array. If size <= 1, no need to do anything and just return. Then use double pointers to store this Boolean array. One pointer points to the head and the other pointer points to the tail. The following are several cases: If the element corresponding to the head pointer is True, the head pointer +1, and the loop continues. If the element

corresponding to the head pointer is False and the element corresponding to the tail pointer is False, the tail pointer -1, and the loop continues. If the element corresponding to the head pointer is False and the element corresponding to the tail pointer is True, swap the two elements and increase the head pointer and the tail pointer by one. The loop ends when the head pointer is less than or equal to the tail pointer.

Feature:

1. The Find method searches through the array a for the target value v by linear search. If found, it returns the index of the target value in the array; if not found, it returns -1. The postcondition of the method ensures that the returned index is legal and the value at the index position is equal to the target value. It also verifies that the target value is not in the array if not found. The safety of the traversal and the correctness of the processed part are guaranteed by the loop invariant.
2. The Sum method calculates the total of the formula 10n+10(n−1)+…+10 and returns the result. The method uses nested loops to progressively accumulate the sum, with each iteration computing the current term 10i10i10i and adding it to the total. The loop invariants ensure the correctness of the relationship between the partial sum and the

unprocessed portion of the total. The method's postcondition verifies that the result conforms to the formula 5 * n * (n + 1)

3. The ArrayMin method is used to find the minimum value in a non-empty integer array a by traversing the array from the first element, comparing it with the current minimum value min and updating it. The postcondition of the method ensures that the return value is the minimum value of all elements in the array and exists in the array. The loop invariant ensures the safety of the traversal and that the current min is always the minimum value of the processed part.
4. The SortCoins method sorts the Boolean array a in-place using a two-pointer method, so that all True (heads) come before False (tails). The left pointer processes heads coins starting at the beginning of the array, and the right pointer processes tails coins starting at the end, swapping positions if they are out of order. The method's postcondition verifies that the sort is correct, while ensuring that the multiset of the array elements remains unchanged.

Issues:

There were no problems when writing the first three programs. The problem occurred in the fourth program. At first, I wanted to use strings to represent the front and back sides, but I found that it was more convenient to use Boolean values. Another problem was when writing the post condition of this program. I

couldn't figure out how to ensure that all the front sides appeared before any back sides. Finally, I thought of using the logical expression imple. Another thing was to ensure that the elements of the array after the change were exactly the same as the original array. I solved this problem by looking up information.