

# Experiment ‘p4\_2\_c\_cumulative\_totals’ Results

December 18, 2025

**Experiment outcome:** SUCCESS

**Bad responses:** 0

**Responses containing assume :** 0

**Resolution attempts:** 5

**Hard fails (resolution):** 0

**Soft fails (resolution):** 0

**Verification attempts:** 5

## Problem Specification

**Problem name:** p4\_2\_c\_cumulative\_totals

**Natural language statement:** Write a method that takes a sequence of integers and returns cumulative totals. For example, if the input is [1, 7, 2, 9], the method should return [1, 8, 10, 19].

**Method signature:** p4\_2\_c\_cumulative\_totals(inputs: seq<int>) returns (totals: seq<int>)

### Ensures

- $|\text{totals}| == |\text{inputs}|$
- $\forall i :: 0 \leq i < |\text{inputs}| \implies \text{totals}[i] == (\text{if } i == 0 \text{ then } \text{inputs}[0] \text{ else } \text{totals}[i - 1] + \text{inputs}[i])$

### Requires

- $0 < |\text{inputs}|$

## GenAI interactions

Below you will find all interactions between the ‘user’ (program) and the ‘assistant’ (OpenAI).

### Program → GenAI

You are given the following task to perform in Dafny:

Write a method that takes a sequence of integers and returns cumulative totals. For example, if the input is [1, 7, 2, 9], the method should return [1, 8, 10, 19].

The signature should be:

```
method p4_2_c_cumulative_totals(inputs: seq<int>) returns (totals: seq<int>)
```

The method should respect the following contract:

```
requires 0 < |inputs|, ensures |totals| == |inputs|, ensures forall i :: 0 <= i < |inputs
    ↪ | ==> totals[i] == (if i == 0 then inputs[0] else totals[i - 1] + inputs[i])
```

Produce and show only the Dafny body of this method, including the curly braces that surround it. Do not show the signature nor contract. You must not use ‘assume’ anywhere in your code.

### GenAI → Program

**System fingerprint:**

**ID:** chatmpl-Co4b9NoPVIIonT25d9lme5Cxoysk2

**Model:** gpt-5.1-chat-latest

**Created at:** 1766050111

**Finish reason:** stop

**Usage:** 428 tokens in, and 215 tokens out

```
{
    var res: seq<int> := [inputs[0]];
    var i: int := 1;
    while i < |inputs|
        invariant 1 <= i <= |inputs|
        invariant |res| == i
        invariant forall j :: 0 <= j < i ==> res[j] == (if j == 0 then inputs[0] else res[j -
            ↪ 1] + inputs[j])
    {
        res := res + [res[i - 1] + inputs[i]];
        i := i + 1;
    }
    totals := res;
}
```

### Program → GenAI

When using dafny verify, the below error is emitted and verify fails:

```

problem_data/p4_2_c_cumulative_totals/program.dfy(11,14): Error: this invariant could not
    ↪ be proved to be maintained by the loop
Related message: loop invariant violation
|
11 | invariant forall j :: 0 <= j < i ==> res[j] == (if j == 0 then inputs[0] else res[j
    ↪ - 1] + inputs[j])
| ~~~~~~

```

Dafny program verifier finished with 1 verified, 1 error

Correct the error by altering only the method body. Produce and show only the Dafny body,  
 ↪ including the curly braces that surround it. Do not show the signature nor  
 ↪ contract. You must not use 'assume' anywhere in your code.

## GenAI → Program

### System fingerprint:

**ID:** chatcmpl-Co4bEUyTxUEXvqvv4091V3x9Qz1wh

**Model:** gpt-5.1-chat-latest

**Created at:** 1766050116

**Finish reason:** stop

**Usage:** 720 tokens in, and 157 tokens out

```
{
    var res: seq<int> := [inputs[0]];
    var i: int := 1;
    while i < |inputs|
        invariant 1 <= i <= |inputs|
        invariant |res| == i
        invariant forall j :: 0 <= j < i ==> res[j] == (if j == 0 then inputs[0] else res[j-1]
            ↪ + inputs[j])
    {
        var next := res[i-1] + inputs[i];
        res := res + [next];
        i := i + 1;
    }
    totals := res;
}
```

## Program → GenAI

When using dafny verify, the below error is emitted and verify fails:

```

problem_data/p4_2_c_cumulative_totals/program.dfy(11,14): Error: this invariant could not
    ↪ be proved to be maintained by the loop
Related message: loop invariant violation
|
11 | invariant forall j :: 0 <= j < i ==> res[j] == (if j == 0 then inputs[0] else res[j
    ↪ - 1] + inputs[j])

```

```
| ~~~~~~
```

```
Dafny program verifier finished with 1 verified, 1 error
```

```
Correct the error by altering only the method body. Produce and show only the Dafny body,  
↳ including the curly braces that surround it. Do not show the signature nor  
↳ contract. You must not use 'assume' anywhere in your code.
```

## GenAI → Program

**System fingerprint:**

**ID:** chatcmpl-Co4bIWUmCkYbBeoID21IfIxIRtqT2

**Model:** gpt-5.1-chat-latest

**Created at:** 1766050120

**Finish reason:** stop

**Usage:** 1044 tokens in, and 545 tokens out

```
{  
    var res: seq<int> := [inputs[0]];  
    var i: int := 1;  
    while i < |inputs|  
        invariant 1 <= i <= |inputs|  
        invariant |res| == i  
        invariant forall j :: 0 <= j < |res| ==> res[j] == (if j == 0 then inputs[0] else res[  
            ↳ j - 1] + inputs[j])  
    {  
        var next := res[i - 1] + inputs[i];  
        res := res + [next];  
        i := i + 1;  
    }  
    totals := res;  
}
```

## Program → GenAI

```
When using dafny verify, the below error is emitted and verify fails:
```

```
problem_data/p4_2_c_cumulative_totals/program.dfy(11,14): Error: this invariant could not  
    ↳ be proved to be maintained by the loop  
Related message: loop invariant violation  
|  
11 | invariant forall j :: 0 <= j < |res| ==> res[j] == (if j == 0 then inputs[0] else  
    ↳ res[j - 1] + inputs[j])  
| ~~~~~~
```

```
Dafny program verifier finished with 1 verified, 1 error
```

```
Correct the error by altering only the method body. Produce and show only the Dafny body,  
↳ including the curly braces that surround it. Do not show the signature nor
```

↪ contract. You must not use 'assume' anywhere in your code.

## GenAI → Program

**System fingerprint:**

**ID:** chatmpl-Co4bPpT5IwO0oeud2z4ivoTvrlrT

**Model:** gpt-5.1-chat-latest

**Created at:** 1766050127

**Finish reason:** stop

**Usage:** 1379 tokens in, and 243 tokens out

```
{  
    var res: seq<int> := [];  
    var running: int := 0;  
    var i: int := 0;  
    while i < |inputs|  
        invariant 0 <= i <= |inputs|  
        invariant |res| == i  
        invariant forall j :: 0 <= j < i ==> res[j] == (if j == 0 then inputs[0] else res[j-1]  
            ↪ + inputs[j])  
        invariant running == (if i == 0 then 0 else res[i-1])  
    {  
        running := running + inputs[i];  
        res := res + [running];  
        i := i + 1;  
    }  
    totals := res;  
}
```

## Program → GenAI

When using dafny verify, the below error is emitted and verify fails:

```
problem_data/p4_2_c_cumulative_totals/program.dfy(12,14): Error: this invariant could not  
    ↪ be proved to be maintained by the loop  
Related message: loop invariant violation  
|  
12 | invariant forall j :: 0 <= j < i ==> res[j] == (if j == 0 then inputs[0] else res[j]  
    ↪ -1] + inputs[j])  
| ^~~~~~
```

Dafny program verifier finished with 1 verified, 1 error

Correct the error by altering only the method body. Produce and show only the Dafny body,  
 ↪ including the curly braces that surround it. Do not show the signature nor  
 ↪ contract. You must not use 'assume' anywhere in your code.

## GenAI → Program

System fingerprint:

ID: chatcmpl-Co4bTGZJ7N4G3EOjajXF7pEsWFFYH

Model: gpt-5.1-chat-latest

Created at: 1766050131

Finish reason: stop

Usage: 1723 tokens in, and 315 tokens out

```
{  
    var res: seq<int> := [];  
    var i: int := 0;  
    while i < |inputs|  
        invariant 0 <= i <= |inputs|  
        invariant |res| == i  
        invariant i == 0 ==> res == []  
        invariant i > 0 ==> res[0] == inputs[0]  
        invariant forall j :: 1 <= j < i ==> res[j] == res[j-1] + inputs[j]  
    {  
        if i == 0 {  
            res := [inputs[0]];  
        } else {  
            res := res + [res[i-1] + inputs[i]];  
        }  
        i := i + 1;  
    }  
    totals := res;  
}
```

## Final Program

```
method p4_2_c_cumulative_totals(inputs: seq<int>) returns (totals: seq<int>)  
    requires 0 < |inputs|  
    ensures |totals| == |inputs|  
    ensures forall i :: 0 <= i < |inputs| ==> totals[i] == (if i == 0 then inputs[0]  
        ↪ else totals[i - 1] + inputs[i])  
{  
    var res: seq<int> := [];  
    var i: int := 0;  
    while i < |inputs|  
        invariant 0 <= i <= |inputs|  
        invariant |res| == i  
        invariant i == 0 ==> res == []  
        invariant i > 0 ==> res[0] == inputs[0]  
        invariant forall j :: 1 <= j < i ==> res[j] == res[j-1] + inputs[j]  
    {  
        if i == 0 {  
            res := [inputs[0]];  
        } else {  
            res := res + [res[i-1] + inputs[i]];  
        }  
    }
```

```
    i := i + 1;
}
totals := res;
}
```

## Total Token Usage

**Input tokens:** 5294

**Output tokens:** 1475

**Reasoning tokens:** 640

**Sum of ‘total tokens’:** 6769

## Experiment Timings

**Overall Experiment** started at 1766050113522, ended at 1766050138310, lasting 24788ms (24.79 seconds)

**Iteration #4** started at 1766050129383, ended at 1766050133463, lasting 4080ms (4.08 seconds)

**Iteration #5** started at 1766050133463, ended at 1766050138310, lasting 4847ms (4.85 seconds)

**Iteration #1** started at 1766050113522, ended at 1766050118450, lasting 4928ms (4.93 seconds)

**Iteration #2** started at 1766050118450, ended at 1766050122150, lasting 3700ms (3.70 seconds)

**Iteration #3** started at 1766050122150, ended at 1766050129383, lasting 7233ms (7.23 seconds)