

Experiment ‘p4_2_a_min_max’ Results

December 18, 2025

Experiment outcome: SUCCESS

Bad responses: 0

Responses containing assume : 0

Resolution attempts: 1

Hard fails (resolution): 0

Soft fails (resolution): 0

Verification attempts: 1

Problem Specification

Problem name: p4_2_a_min_max

Natural language statement: Write a method that takes a sequence of integers and returns the smallest and largest of the inputs.

Method signature: p4_2_a_min_max(inputs: array<int>) returns (min: int, max: int)

Ensures

- $\forall i :: 0 \leq i < \text{inputs.Length} \Rightarrow \text{min} \leq \text{inputs}[i]$
- $\forall i :: 0 \leq i < \text{inputs.Length} \Rightarrow \text{max} \geq \text{inputs}[i]$
- $\exists i :: 0 \leq i < \text{inputs.Length} \& \& \text{min} == \text{inputs}[i]$
- $\exists i :: 0 \leq i < \text{inputs.Length} \& \& \text{max} == \text{inputs}[i]$

Requires

- $0 < \text{inputs.Length}$

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 the smallest and largest of
→ the inputs.

The signature should be:

```
method p4_2_a_min_max(inputs: array<int>) returns (min: int, max: int)
```

The method should respect the following contract:

```
requires 0 < inputs.Length, ensures forall i :: 0 <= i < inputs.Length ==> min <= inputs[  
→ i], ensures forall i :: 0 <= i < inputs.Length ==> max >= inputs[i], ensures  
→ exists i :: 0 <= i < inputs.Length && min == inputs[i], ensures exists i :: 0 <= i  
→ < inputs.Length && max == 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: chatcmpl-Co4aQubsfiEUiAfp1GiBDUH6e0E1f

Model: gpt-5.1-chat-latest

Created at: 1766050066

Finish reason: stop

Usage: 402 tokens in, and 197 tokens out

```
{  
    var i := 0;  
    min := inputs[i];  
    max := inputs[i];  
    i := i + 1;  
    while i < inputs.Length  
        invariant 1 <= i <= inputs.Length  
        invariant forall j :: 0 <= j < i ==> min <= inputs[j]  
        invariant forall j :: 0 <= j < i ==> max >= inputs[j]  
        invariant exists j :: 0 <= j < i && min == inputs[j]  
        invariant exists j :: 0 <= j < i && max == inputs[j]  
    {  
        if inputs[i] < min {  
            min := inputs[i];  
        }  
        if inputs[i] > max {  
            max := inputs[i];  
        }  
    }  
}
```

```

        }
        i := i + 1;
    }
}

```

Final Program

```

method p4_2_a_min_max(inputs: array<int>) returns (min: int, max: int)
    requires 0 < inputs.Length
    ensures forall i :: 0 <= i < inputs.Length ==> min <= inputs[i]
    ensures forall i :: 0 <= i < inputs.Length ==> max >= inputs[i]
    ensures exists i :: 0 <= i < inputs.Length && min == inputs[i]
    ensures exists i :: 0 <= i < inputs.Length && max == inputs[i]
{
    var i := 0;
    min := inputs[i];
    max := inputs[i];
    i := i + 1;
    while i < inputs.Length
        invariant 1 <= i <= inputs.Length
        invariant forall j :: 0 <= j < i ==> min <= inputs[j]
        invariant forall j :: 0 <= j < i ==> max >= inputs[j]
        invariant exists j :: 0 <= j < i && min == inputs[j]
        invariant exists j :: 0 <= j < i && max == inputs[j]
    {
        if inputs[i] < min {
            min := inputs[i];
        }
        if inputs[i] > max {
            max := inputs[i];
        }
        i := i + 1;
    }
}

```

Total Token Usage

Input tokens: 402
Output tokens: 197
Reasoning tokens: 0
Sum of ‘total tokens’: 599

Experiment Timings

Overall Experiment started at 1766050068068, ended at 1766050073106, lasting 5038ms (5.04 seconds)
Iteration #1 started at 1766050068068, ended at 1766050073106, lasting 5038ms (5.04 seconds)