

Experiment ‘p4_1_e_sum_odd_digits’ 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_1_e_sum_odd_digits

Natural language statement: Write a method with loops that computes the sum of all odd digits of an input. (For example, if the input is 32677, the sum would be $3 + 7 + 7 = 17$.)

Method signature: p4_1_e_sum_odd_digits(a: int) returns (sum: int)

Ensures

- `sum >= 0`
- `sum == s_o_d(a)`

Requires

- `a >= 0`

Functional Code Given

```
function s_o_d(a: int): int
requires a >= 0
{
    if a == 0 then 0
    else if a % 2 == 0 then s_o_d(a / 10)
    else a % 10 + s_o_d(a / 10)
}
```

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 with loops that computes the sum of all odd digits of an input. (For ↪ example, if the input is 32677, the sum would be $3 + 7 + 7 = 17$.)

The signature should be:

```
method p4_1_e_sum_odd_digits(a: int) returns (sum: int)
```

The method should respect the following contract:

```
requires a >= 0, ensures sum >= 0, ensures sum == s_o_d(a)
```

The contract uses the following dafny code:

```
function s_o_d(a: int): int
requires a >= 0
{
    if a == 0 then 0
    else if a % 2 == 0 then s_o_d(a / 10)
    else a % 10 + s_o_d(a / 10)
}
```

These function(s) must not be used in your implementation of the method.

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-Co4aIgFyG5zil5Z0nV05CmFICreSc

Model: gpt-5.1-chat-latest

Created at: 1766050058

Finish reason: stop

Usage: 510 tokens in, and 438 tokens out

```
{
    var x := a;
    sum := 0;
    while x > 0
        invariant x >= 0
        invariant sum >= 0
        invariant sum + s_o_d(x) == s_o_d(a)
    {
```

```

var d := x % 10;
if d % 2 != 0 {
    sum := sum + d;
}
x := x / 10;
}
}

```

Final Program

```

function s_o_d(a: int): int
requires a >= 0
{
    if a == 0 then 0
    else if a % 2 == 0 then s_o_d(a / 10)
    else a % 10 + s_o_d(a / 10)
}

method p4_1_e_sum_odd_digits(a: int) returns (sum: int)
    requires a >= 0
    ensures sum >= 0
    ensures sum == s_o_d(a)
{
    var x := a;
    sum := 0;
    while x > 0
        invariant x >= 0
        invariant sum >= 0
        invariant sum + s_o_d(x) == s_o_d(a)
    {
        var d := x % 10;
        if d % 2 != 0 {
            sum := sum + d;
        }
        x := x / 10;
    }
}

```

Total Token Usage

Input tokens: 510
Output tokens: 438
Reasoning tokens: 320
Sum of ‘total tokens’: 948

Experiment Timings

Overall Experiment started at 1766050060475, ended at 1766050067314, lasting 6839ms (6.84 seconds)
Iteration #1 started at 1766050060476, ended at 1766050067314, lasting 6838ms (6.84 seconds)