

# Experiment ‘P\_3\_3\_NumberOfDigits’ Results

December 18, 2025

**Experiment outcome:** SUCCESS

**Bad responses:** 0

**Responses containing assume :** 0

**Resolution attempts:** 4

**Hard fails (resolution):** 0

**Soft fails (resolution):** 0

**Verification attempts:** 4

## Problem Specification

**Problem name:** P\_3\_3\_NumberOfDigits

**Natural language statement:** Write a method that takes an integer and returns how many digits the number has, by checking whether the number is 10, 100, and so on. (Assume that all integers are less than ten billion.) If the number is negative, first multiply it with  $-1$ .

**Method signature:** P\_3\_3\_NumberOfDigits (i:int) returns (n: nat)

### Ensures

- $i \geq 0 \implies n == \text{digits}(i)$
- $i < 0 \implies n == \text{digits}(-i)$

### Requires

- $10000000000 > i > -10000000000$

### Functional Code Given

```
function digits (n:nat): nat
{
  if 0 <= n <= 9 then 1
  else 1 + digits (n/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 that takes an integer and returns how many digits the number has, by  
→ checking whether the number is 10, 100, and so on. (Assume that all integers are  
→ less than ten billion.) If the number is negative, first multiply it with -1.

The signature should be:

method P\_3\_3\_NumberOfDigits (i:int) returns (n: nat)

The method should respect the following contract:

requires  $10000000000 > i > -10000000000$ , ensures  $i \geq 0 \implies n == \text{digits}(i)$ , ensures  $i < 0$   
→  $\implies n == \text{digits}(-i)$

The contract uses the following dafny code:

```
function digits (n:nat): nat
{
  if 0 <= n <= 9 then 1
  else 1 + digits (n/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:** chatcmpl-Co4SElCLKpj1IRciaRs2BV2FxYqq9

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

**Created at:** 1766049558

**Finish reason:** stop

**Usage:** 518 tokens in, and 278 tokens out

```
{
  var j := if i < 0 then -i else i;
  if j < 10 {
    n := 1;
  } else if j < 100 {
    n := 2;
  } else if j < 1000 {
    n := 3;
  }
```

```

} else if j < 10000 {
  n := 4;
} else if j < 100000 {
  n := 5;
} else if j < 1000000 {
  n := 6;
} else if j < 10000000 {
  n := 7;
} else if j < 100000000 {
  n := 8;
} else if j < 1000000000 {
  n := 9;
} else {
  n := 10;
}
}

```

## Program → GenAI

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

```

problem_data/P_3_3_NumberOfDigits/program.dfy(11,0): Error: a postcondition could not be
  ↳ proved on this return path

```

```

|
11 | {
| ^

```

```

problem_data/P_3_3_NumberOfDigits/program.dfy(9,22): Related location: this is the
  ↳ postcondition that could not be proved

```

```

|
9 | ensures i >= 0 ==> n == digits(i)
| ^^

```

```

problem_data/P_3_3_NumberOfDigits/program.dfy(11,0): Error: a postcondition could not be
  ↳ proved on this return path

```

```

|
11 | {
| ^

```

```

problem_data/P_3_3_NumberOfDigits/program.dfy(10,21): Related location: this is the
  ↳ postcondition that could not be proved

```

```

|
10 | ensures i < 0 ==> n == digits(-i)
| ^^

```

Dafny program verifier finished with 2 verified, 2 errors

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-Co4SJ7xeT3l1VSpjvg4FFZ7HFjxbA

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

**Created at:** 1766049563

**Finish reason:** stop

**Usage:** 1032 tokens in, and 212 tokens out

```
{
  var j := if i < 0 then -i else i;
  n := 0;
  var k := j;
  while k > 0
    invariant k >= 0
    invariant n >= 0
    invariant if i >= 0 then (n + digits(k) == digits(i))
      else (n + digits(k) == digits(-i))
  {
    n := n + 1;
    k := k / 10;
  }
  if j == 0 {
    n := 1;
  }
}
```

## Program → GenAI

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

```
problem_data/P_3_3_NumberOfDigits/program.dfy(11,0): Error: a postcondition could not be
  ↳ proved on this return path
```

```
|
11 | {
| ^
```

```
problem_data/P_3_3_NumberOfDigits/program.dfy(9,22): Related location: this is the
  ↳ postcondition that could not be proved
```

```
|
9 | ensures i >= 0 ==> n == digits(i)
| ^^
```

```
problem_data/P_3_3_NumberOfDigits/program.dfy(11,0): Error: a postcondition could not be
  ↳ proved on this return path
```

```
|
11 | {
| ^
```

```
problem_data/P_3_3_NumberOfDigits/program.dfy(10,21): Related location: this is the
  ↳ postcondition that could not be proved
```

```
|
```

```

10 | ensures i < 0 ==> n == digits(-i)
    | ^^

problem_data/P_3_3_NumberOfDigits/program.dfy(18,44): Error: this invariant could not be
    ↪ proved to be maintained by the loop
Related message: loop invariant violation
    |
18 | invariant if i >= 0 then (n + digits(k) == digits(i))
    | ^^

problem_data/P_3_3_NumberOfDigits/program.dfy(19,34): Error: this invariant could not be
    ↪ proved to be maintained by the loop
Related message: loop invariant violation
    |
19 | else (n + digits(k) == digits(-i))
    | ^^

Dafny program verifier finished with 2 verified, 4 errors

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-Co4SO8SaJw7QX8SgfLsbY4iawibOd

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

**Created at:** 1766049568

**Finish reason:** stop

**Usage:** 1705 tokens in, and 408 tokens out

```

{
  var j:nat := if i < 0 then -i else i;
  if j < 10 {
    n := 1;
  } else if j < 100 {
    n := 2;
  } else if j < 1000 {
    n := 3;
  } else if j < 10000 {
    n := 4;
  } else if j < 100000 {
    n := 5;
  } else if j < 1000000 {
    n := 6;
  } else if j < 10000000 {
    n := 7;
  } else if j < 100000000 {
    n := 8;
  }
}

```

```

} else if j < 1000000000 {
  n := 9;
} else {
  n := 10;
}
}

```

## Program → GenAI

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

```

problem_data/P_3_3_NumberOfDigits/program.dfy(11,0): Error: a postcondition could not be
    ↪ proved on this return path

```

```

|
11 | {
| ~

```

```

problem_data/P_3_3_NumberOfDigits/program.dfy(9,22): Related location: this is the
    ↪ postcondition that could not be proved

```

```

|
9 | ensures i >= 0 ==> n == digits(i)
| ^^

```

```

problem_data/P_3_3_NumberOfDigits/program.dfy(11,0): Error: a postcondition could not be
    ↪ proved on this return path

```

```

|
11 | {
| ~

```

```

problem_data/P_3_3_NumberOfDigits/program.dfy(10,21): Related location: this is the
    ↪ postcondition that could not be proved

```

```

|
10 | ensures i < 0 ==> n == digits(-i)
| ^^

```

Dafny program verifier finished with 2 verified, 2 errors

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-Co4SWvhrgp9DBPDEqE3uZWtjW7pan

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

**Created at:** 1766049576

**Finish reason:** stop

**Usage:** 2065 tokens in, and 152 tokens out

```

{
  var j:nat := if i < 0 then -i else i;
  var k:nat := j;
  n := 0;

  if k == 0 {
    n := 1;
    return;
  }

  while k > 0
    invariant k > 0 ==> digits(k) + n == digits(j)
    invariant k == 0 ==> n == digits(j)
    invariant n >= 0
    invariant k <= j
  {
    n := n + 1;
    k := k / 10;
  }
}

```

## Final Program

```

function digits (n:nat): nat
{
  if 0 <= n <= 9 then 1
  else 1 + digits (n/10)
}

method P_3_3_NumberOfDigits (i:int) returns (n: nat)
  requires 10000000000 > i > -10000000000
  ensures i >= 0 ==> n == digits(i)
  ensures i < 0 ==> n == digits(-i)
{
  var j:nat := if i < 0 then -i else i;
  var k:nat := j;
  n := 0;

  if k == 0 {
    n := 1;
    return;
  }

  while k > 0
    invariant k > 0 ==> digits(k) + n == digits(j)
    invariant k == 0 ==> n == digits(j)
    invariant n >= 0
    invariant k <= j
  {
    n := n + 1;

```

```
    k := k / 10;  
  }  
}
```

## Total Token Usage

**Input tokens:** 5320

**Output tokens:** 1050

**Reasoning tokens:** 320

**Sum of ‘total tokens’:** 6370

## Experiment Timings

**Overall Experiment** started at 1766049559813, ended at 1766049582378, lasting 22565ms (22.57 seconds)

**Iteration #4** started at 1766049577936, ended at 1766049582378, lasting 4442ms (4.44 seconds)

**Iteration #1** started at 1766049559814, ended at 1766049565512, lasting 5698ms (5.70 seconds)

**Iteration #2** started at 1766049565512, ended at 1766049570305, lasting 4793ms (4.79 seconds)

**Iteration #3** started at 1766049570305, ended at 1766049577936, lasting 7631ms (7.63 seconds)