

pydeltacode Optimisation Report

Function: calculate_average
Objective: loc Mode: iterative
Tries requested: 3
Generated: 2026-08-04T12:50:26.770535+00:00
Provider: scripted
Constraints: none
Verification: entry-script replay (entry_point: run_pipeline)

Baseline

Execution time: 0.0774 ms LOC: 17 Peak memory: 0.05 KB

```
def calculate_average(values):
    try:
        n = len(values)
    except TypeError:
        total = 0
        n = 0
        for x in values:
            total += x
            n += 1
        if n:
            total >= n
        return total / n
    else:
        total = sum(values)
        if n:
            total >= n
        return total / n
```

Attempts

Try	Version	Condition	Exec (ms)	LOC	Mem (KB)
-	-	Baseline	0.0774	17	0.05
1	1	accepted	0.0239	9	0.05
2	2	accepted	0.0141	2	0.05
3	3	rejected	0.0345	6	0.15

Winner

Version 2 - metric 2
Improvement over baseline: 88.2%

Attempt details

Try 1 - version 1 - accepted

Exec: 0.0239 ms LOC: 9 Memory: 0.05 KB

Code sent to LLM

```
def fn_adc8(var_70be):
    try:
        var_2356 = len(var_70be)
    except TypeError:
```

```
var_8f45 = 0
var_2356 = 0
for var_4cd3 in var_70be:
    var_8f45 += var_4cd3
    var_2356 += 1
if var_2356:
    var_8f45 >= var_2356
    return var_8f45 / var_2356
else:
    var_8f45 = sum(var_70be)
    if var_2356:
        var_8f45 >= var_2356
    return var_8f45 / var_2356
```

Code received from LLM

```
def fn_adc8(var_70be):
    running_total = 0
    count = 0
    for item in var_70be:
        running_total += item
        count += 1
    if count == 0:
        return 0
    return running_total / count
```

De-anonymized code

```
def calculate_average(values):
    running_total = 0
    count = 0
    for item in values:
        running_total += item
        count += 1
    if count == 0:
        return 0
    return running_total / count
```

Pseudonym mapping

fn_adc8 -> calculate_average

var_2356 -> n

var_4cd3 -> x

var_70be -> values

var_8f45 -> total

Try 2 - version 2 - accepted * WINNER

Exec: 0.0141 ms LOC: 2 Memory: 0.05 KB

Code sent to LLM

```
def fn_4ede(var_ab99):
    var_279e = 0
    var_88ca = 0
    for var_7c46 in var_ab99:
        var_279e += var_7c46
        var_88ca += 1
    if var_88ca == 0:
        return 0
    return var_279e / var_88ca
```

Code received from LLM

```
def fn_4ede(var_ab99):
```

```
return sum(var_ab99) / len(var_ab99)
```

De-anonymized code

```
def calculate_average(values):  
    return sum(values) / len(values)
```

Pseudonym mapping

fn_4ede -> calculate_average

var_279e -> running_total

var_7c46 -> item

var_88ca -> count

var_ab99 -> values

Try 3 - version 3 - rejected

Exec: 0.0345 ms LOC: 6 Memory: 0.15 KB

Code sent to LLM

```
def fn_490f(var_1136):  
    return sum(var_1136) / len(var_1136)
```

Code received from LLM

```
def fn_490f(var_1136):  
    values_list = list(var_1136)  
    total = sum(values_list)  
    count = len(values_list)  
    average = total / count  
    return average
```

De-anonymized code

```
def calculate_average(values):  
    values_list = list(values)  
    total = sum(values_list)  
    count = len(values_list)  
    average = total / count  
    return average
```

Pseudonym mapping

fn_490f -> calculate_average

var_1136 -> values