CHAPTER 1

Rewriting Expressions - Solution

In this exercise we ask you to rewrite some expressions or to understand the different forms that a solution can take. Some of the expressions are explained in https://medium.com/concerning-pharo/elegant-pharo-code-bb590f0856d0

Exercise: Examine the block expressions

```
| sum |
sum := 0.
#(21 23 53 66 87) do: [:item | sum := sum + item].
sum
```

Exercise:

- What is the final result of sum?

Solution. The sum of all the numbers of the array: 250

Exercise:

- Rewrite this piece of code to use explicit array indexing (with the message `at:`) to access the array elements. Test your version.

Solution.

```
| array sum |
array := #(21 23 53 66 87).
sum := 0.
1 to: array size do: [ :i | sum := sum + array at: i ].
sum
```

Exercise:

- Rewrite this code using `inject:into:` (check the implementation of `inject:into:` and its users in the system to understand how to use it).

Solution.

```
#(21 23 53 66 87) inject: 0 into: [:item :sum | sum + item]
```

Exercise: Comparing expressions

You can express in different way the same computation. Have a look at each of them and check the messages that you do not know. Look for their implementation.

```
| array |
array := #(2 4 4 4 5 5 7 9).
((array - array average) squared sum / (array size - 1)) sqrt
```

```
[ :input | ((input - input average) squared sum / (input size - 1))
    sqrt ]
    value: #(2 4 4 4 5 5 7 9)
```

```
#(2 4 4 4 5 5 7 9) in: [ :input |
    ((input - input average) squared sum / (input size - 1)) sqrt ]
```

```
#(2 4 4 4 5 5 7 9) stddev
```