Challenge: Web Application for Statistics

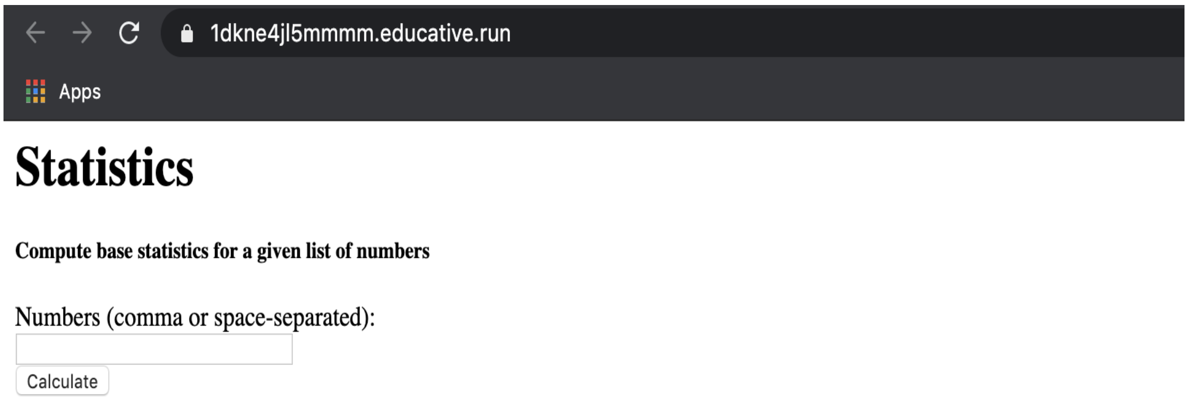
This lesson brings you a challenge to solve.

WE'LL COVER THE FOLLOWING

- Problem statement

Problem statement

Develop a web application that lets the user put in a series of *numbers*, and prints out the numbers, their *count*, their *mean*, and their *median*, like in the following screenshot:



1dkne4jl5mmmm.educative.run

Apps

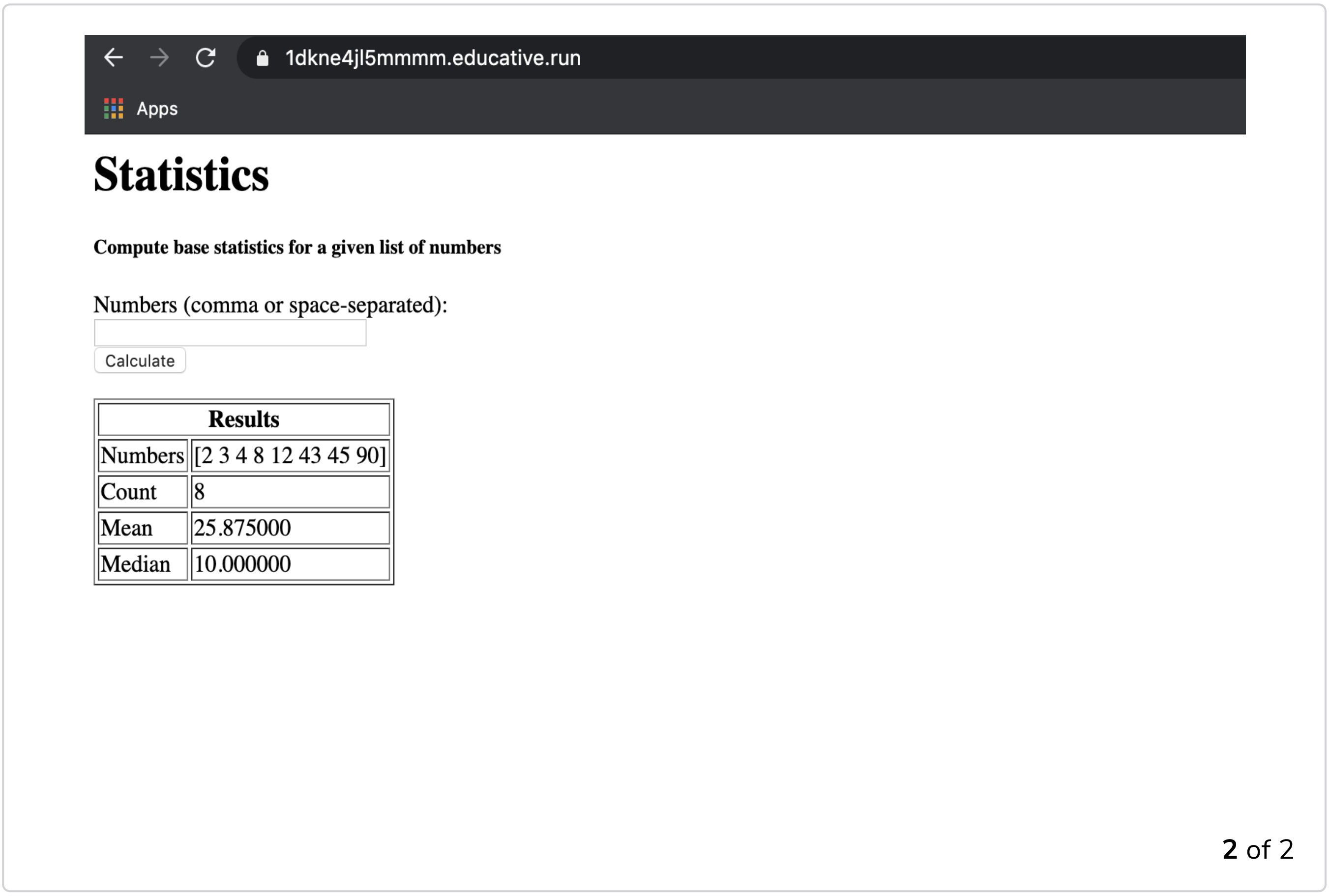
Statistics

Compute base statistics for a given list of numbers

Numbers (comma or space-separated):

Calculate

1 of 2


1dkne4jl5mmmm.educative.run

Apps

Statistics

Compute base statistics for a given list of numbers

Numbers (comma or space-separated):

Calculate

Results	
Numbers	[2 3 4 8 12 43 45 90]
Count	8
Mean	25.875000
Median	10.000000

2 of 2

Remark: Use `0.0.0.0:3000` or `localhost:3000` or simply `:3000` for the connection. If port 3000 is already occupied, use `localhost:9001` for example.

Try to attempt the challenge below. Feel free to view the solution, after giving some shots. Good Luck!

Hint: Do not forget to *import* `log`, `sort`, `strconv`, and `strings` packages.

Environment Variables

Key:	Value:
GOROOT	/usr/local/go
GOPATH	//root/usr/local/go/src
PATH	//root/usr/local/go/src/bin:/usr/local/go...

```go
package main
import (
        "fmt"

        "net/http"
)

type statistics struct {
        numbers []float64
        mean    float64
        median  float64
}

const form = `<html><body><form action="/" method="POST">
<h1>Statistics</h1>
<h5>Compute base statistics for a given list of numbers</h5>
<label for="numbers">Numbers (comma or space-separated):</label><br>
<input type="text" name="numbers" size="30"><br />
<input type="submit" value="Calculate">
</form></html></body>`

const error = `<p class="error">%s</p>`

var pageTop = ""
var pageBottom = ""

// Define a root handler for requests to function homePage, and start the webserver combined
func main() {

}

// Write an HTML header, parse the form, write form to writer and make request for numbers
func homePage(writer http.ResponseWriter, request *http.Request) {
        // write your code here
}

// Capture the numbers from the request, and format the data and check for errors
func processRequest(request *http.Request) ([]float64, string, bool) {
        // write your code here
        return nil, "", false
}

// sort the values to get mean and median
func getStats(numbers []float64) (stats statistics) {
        // write your code here
        return stats
}

// seperate function to calculate the sum for mean
func sum(numbers []float64) (total float64) {
        // write your code here
        return 0
}

// seperate function to calculate the median
func median(numbers []float64) float64 {
        // write your code here
        return 0
}

func formatStats(stats statistics) string {
        return fmt.Sprintf(`<table border="1">
<tr><th colspan="2">Results</th></tr>
<tr><td>Numbers</td><td>%v</td></tr>
<tr><td>Count</td><td>%d</td></tr>
<tr><td>Mean</td><td>%f</td></tr>

<tr><td>Median</td><td>%f</td></tr>
</table>`, stats.numbers, len(stats.numbers), stats.mean, stats.median)
}
```

We hope that you were able to solve the challenge. The next lesson brings you the solution to this challenge.