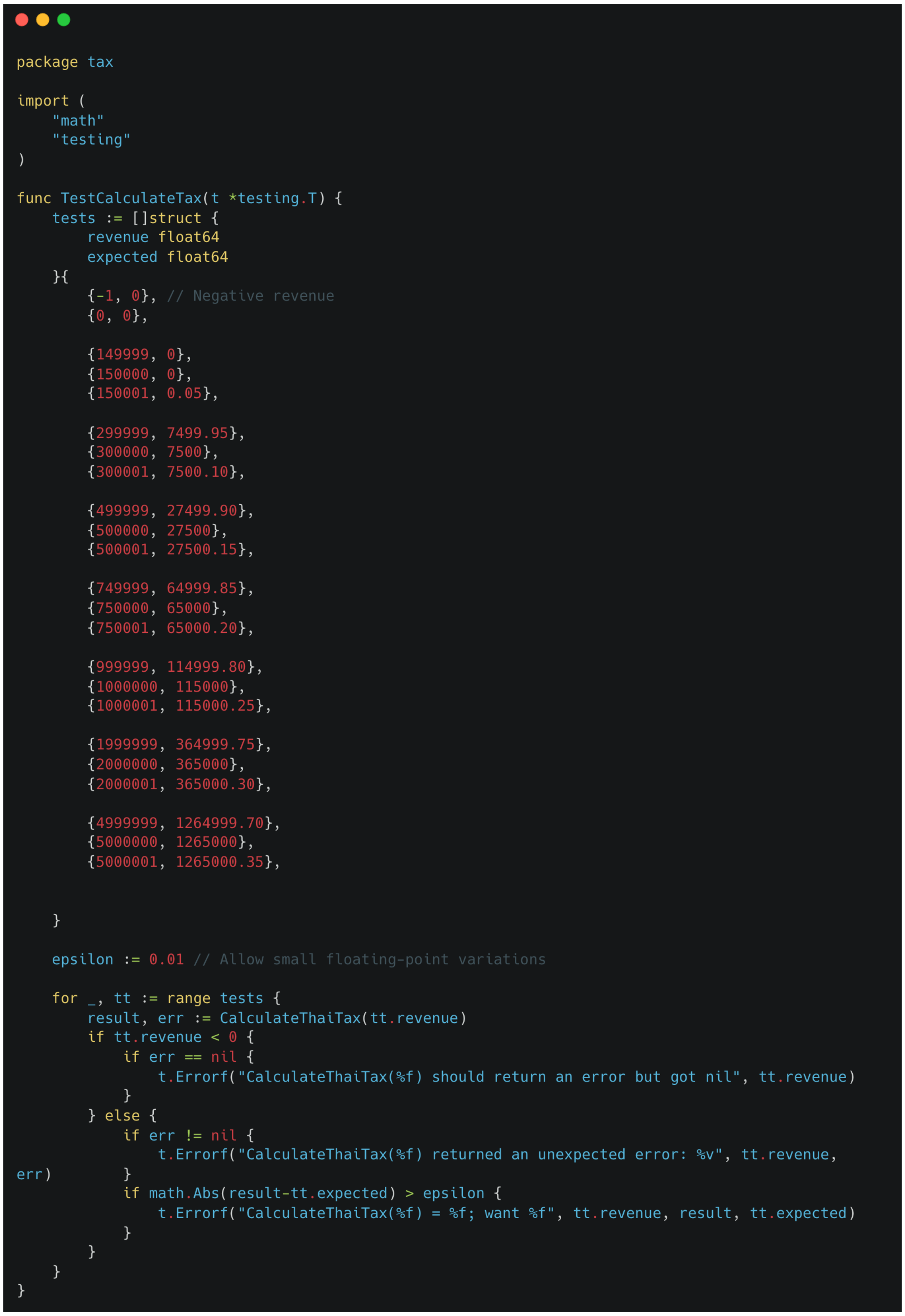
Waranthorn Chansawang 6432154921

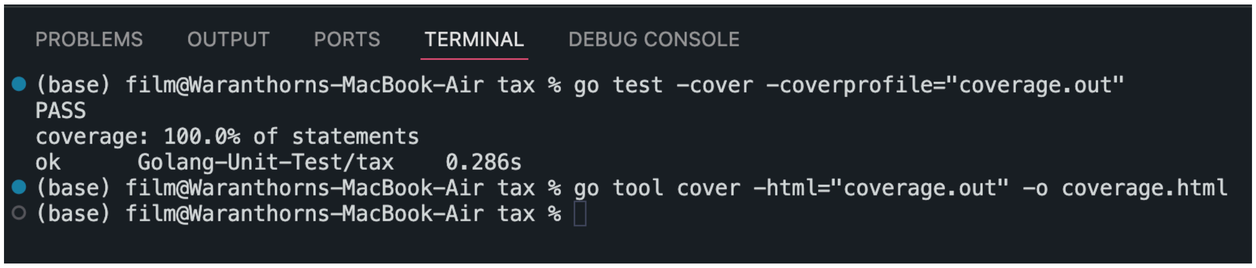
Code



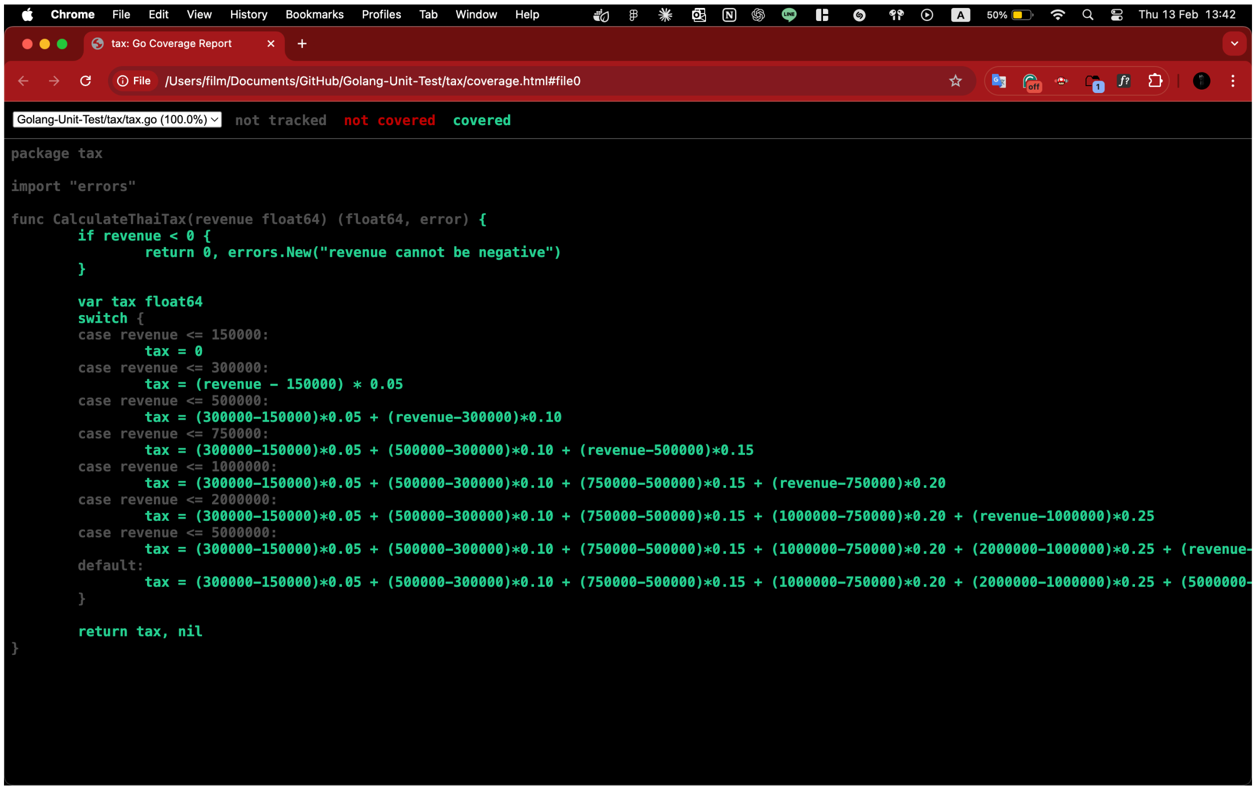
```go
package tax

import (
    "math"
    "testing"
)

func TestCalculateTax(t *testing.T) {
    tests := []struct {
        revenue float64
        expected float64
    }{
        {-1, 0}, // Negative revenue
        {0, 0},

        {149999, 0},
        {150000, 0},
        {150001, 0.05},

        {299999, 7499.95},
        {300000, 7500},
        {300001, 7500.10},

        {499999, 27499.90},
        {500000, 27500},
        {500001, 27500.15},

        {749999, 64999.85},
        {750000, 65000},
        {750001, 65000.20},

        {999999, 114999.80},
        {1000000, 115000},
        {1000001, 115000.25},

        {1999999, 364999.75},
        {2000000, 365000},
        {2000001, 365000.30},

        {4999999, 1264999.70},
        {5000000, 1265000},
        {5000001, 1265000.35},


    }

    epsilon := 0.01 // Allow small floating-point variations

    for _, tt := range tests {
        result, err := CalculateThaiTax(tt.revenue)
        if tt.revenue < 0 {
            if err == nil {
                t.Errorf("CalculateThaiTax(%f) should return an error but got nil", tt.revenue)
            }
        } else {
            if err != nil {
                t.Errorf("CalculateThaiTax(%f) returned an unexpected error: %v", tt.revenue,
err)        }
            if math.Abs(result-tt.expected) > epsilon {
                t.Errorf("CalculateThaiTax(%f) = %f; want %f", tt.revenue, result, tt.expected)
            }
        }
    }
}
```

Waranthorn Chansawang 6432154921

Result



```
● (base) film@Waranthorns-MacBook-Air tax % go test -cover -coverprofile="coverage.out"
  PASS
  coverage: 100.0% of statements
  ok      Golang-Unit-Test/tax    0.286s
● (base) film@Waranthorns-MacBook-Air tax % go tool cover -html="coverage.out" -o coverage.html
○ (base) film@Waranthorns-MacBook-Air tax %
```


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Thu 13 Feb 13:42
tax: Go Coverage Report
File /Users/film/Documents/GitHub/Golang-Unit-Test/tax/coverage.html#file0
Golang-Unit-Test/tax/tax.go (100.0%)
not tracked not covered covered

```
package tax

import "errors"

func CalculateThaiTax(revenue float64) (float64, error) {
	if revenue < 0 {
		return 0, errors.New("revenue cannot be negative")
	}

	var tax float64
	switch {
	case revenue <= 150000:
		tax = 0
	case revenue <= 300000:
		tax = (revenue - 150000) * 0.05
	case revenue <= 500000:
		tax = (300000-150000)*0.05 + (revenue-300000)*0.10
	case revenue <= 750000:
		tax = (300000-150000)*0.05 + (500000-300000)*0.10 + (revenue-500000)*0.15
	case revenue <= 1000000:
		tax = (300000-150000)*0.05 + (500000-300000)*0.10 + (750000-500000)*0.15 + (revenue-750000)*0.20
	case revenue <= 2000000:
		tax = (300000-150000)*0.05 + (500000-300000)*0.10 + (750000-500000)*0.15 + (1000000-750000)*0.20 + (revenue-1000000)*0.25
	case revenue <= 5000000:
		tax = (300000-150000)*0.05 + (500000-300000)*0.10 + (750000-500000)*0.15 + (1000000-750000)*0.20 + (2000000-1000000)*0.25 + (revenue-
	default:
		tax = (300000-150000)*0.05 + (500000-300000)*0.10 + (750000-500000)*0.15 + (1000000-750000)*0.20 + (2000000-1000000)*0.25 + (5000000-
	}

	return tax, nil
}
```