```
using PlutoUI
```

bisection (generic function with 1 method)

```
function bisection(f, a, b; iters=1000, TOL=1e-3)
    a₁, b₁ = a, b
    if sign(f(a₁)) == sign(f(b₁))
        error("Does not satisfy the IVT theorem!")
    end

    for _=1:iters
        p₁ = 0.5 * (a₁ + b₁) # Midpoint
        fₚ₁ = f(p₁)

        if(fₚ₁) == 0 || (b₁-a₁)/2 < TOL
            return p₁
        end

        if f(a₁)*f(p₁) > 0
            a₁ = p₁
        else
            b₁ = p₁
        end
    end

    return "Could not converge for given iterations"
end
```

Root of $f(x) = x^3 + 4x^2 - 10$ is 1.36517333984375:

```
let
    root = bisection(x -> x^3 + 4x^2 - 10, 1, 2, TOL=1e-4)
    md"Root of $f(x) = x^3 + 4x^2 - 10$ is $root:

    "
end
```

fixed_point (generic function with 1 method)

```
function fixed_point(g, x₀; iters=1000, TOL=1e-6)
    x = x₀
    for _=1:iters
        xᵢ = g(x)
        if abs(x - xᵢ) < TOL
            return xᵢ
        end
        x = xᵢ
    end

    "Test inconclusive"
end
```

Root of $f(x) = \frac{1}{x+1}$ is 0.6180338134001252

```
let
    root = fixed_point(x -> 1 / (x + 1), 0)
    md"Root of $f(x) = \frac{1}{x + 1}$ is $root

    "
end
```

```
secant (generic function with 1 method)
```

```
function secant(f, x₁, x₂; iters=1000, TOL=1e-6)

    for _=1:iters
        numerator = f(x₂)
        denominator = (f(x₂) - f(x₁)) / (x₂ - x₁)

        xₙ = x₂ - numerator/denominator
        if abs(xₙ - x₂) < TOL
            return xₙ
        end
        x₁ = x₂
        x₂ = xₙ
    end

    x₂
end
```

```
2.682695795244651
```

```
secant(x -> x^7 - 1000, 2, 3, iters=100)
```

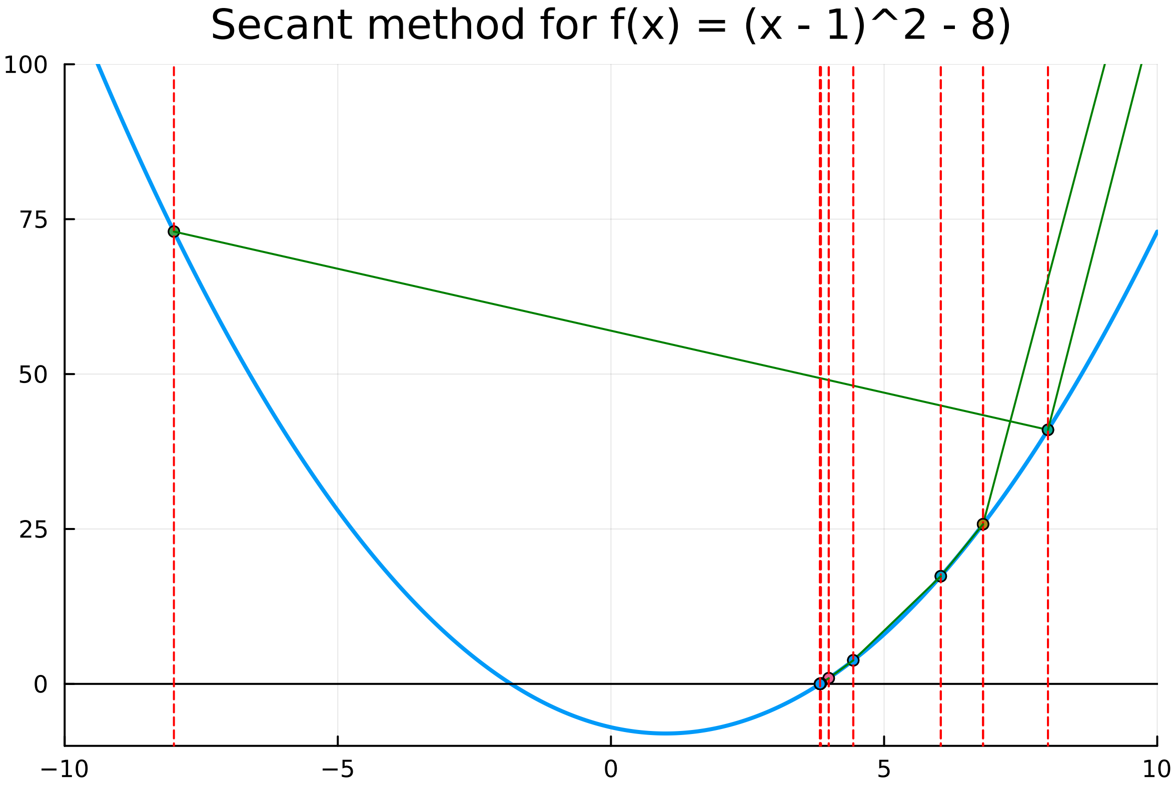
```
using Plots
```

```
next_secant (generic function with 1 method)
```

```
function next_secant(f, x₁, x₂)
    numerator = f(x₂)
    denominator = (f(x₂) - f(x₁)) / (x₂ - x₁)
    return x₂, (x₂ - numerator/denominator)
end
```

18

```
@bind t Slider(1:20, show_value=true)
```



```
begin
    p = plot()
    f(x) = (x - 1)^2 - 8

    to_radians(θ) = θ * (π / 180)

    x_vals = -10:0.01:10
    y_vals = f.(x_vals)

    plot!(p, x_vals, y_vals, label="sin(cos(eˣ))", linewidth=2, legend=false)
    hline!([0], color=:black, linewidth=1, label="y=0")

    xlims!(-10, 10)
    ylims!(-10, 100)

    a, b = -8, 8

    for _ in 1:t
        global a, b
        x, y = next_secant(f, a, b)


        scatter!(p, [a], [f(a)], label="a", markersize=3)
        scatter!(p, [b], [f(b)], label="b", markersize=3)

        vline!([b], color=:red, linewidth=1, linestyle=:dash, linesize=5)
        vline!([a], color=:red, linewidth=1, linestyle=:dash, linesize=5)

        plot!([a, b], [f(a), f(b)], color=:green, linewidth=1)

        a = x
        b = y
        scatter!(p, [], [])
    end

```

```julia
    title!("Secant method for f(x) = (x - 1)^2 - 8)")
    p
end
```