Assignment 2

Meta

Author: Parmandeep Chaddha.

Date: January 8, 2021.

Objective

1. Create a basic 4th root function.

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

```
fourth_root(x) = x ^ (1/4)
```

```
2.0
```

```
# fourth root of 16 is 2
fourth_root(16)
```

```
3.0
```

```
# fourth root of 81 is 3
fourth_root(81)
```

2. Create a function that calculates the nth root of a function

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

```
nth_root(x, n) = x ^ (1/n)
```

```
2.0
```

```
# The third root of 8 is 2.
nth_root(8, 3)
```

```
1.0
```

```
# The 1000th root of 1 is 1.
nth_root(1, 1000)
```

3. Create a 4th root function manually.

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

```
function fourth_root_manual(y:: Float64, guess:: Float64)
    # let the number we are finding be x.
    # we know y = x*x*x*x
    # 0 = x*x*x*x - y
    error= 1.e-6
    iterations = 0

    while (error <= abs(y - guess^4)) && (iterations < 100)
        guess = guess + (y - guess^4) / (4*guess^3)
        iterations += 1
    end

    return guess
end
```

```
answer1 = 2.0305431850181175
```

```
# Requires the number for which the fourth root should be found as well as an intial guess.
answer1 = fourth_root_manual(17.0, 3.0)
```

```
17.000000004996053
```

```
answer1^4 # Close enough to our original guess of 17!
```

4. Create an nth root function manually. Small modification to `fourth_root_manual`.

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

```
function nth_root_manual(y:: Float64, n:: Int64, guess:: Float64)
    error= 1.e-6
    iterations = 0

    while (error <= abs(y - guess^n)) && (iterations < 100)
        guess = guess + (y - guess^n) / (n*guess^(n-1))
        iterations += 1
    end

    return guess
end
```

```
answer2 = 2.51188643150958
```

```
# Calculate the fifth root of 100. A good guess is between 2^5 = 32 and 3^5= 243.
# Therefore, a good guess is 2.5
answer2 = nth_root_manual(100., 5, 2.5)
```

```
100.0
```

```
# Lets see if the answer2 to the fifth power is indeed 100.
answer2 ^ 5
```

Therefore, the Newton Raphson method works quite well!