Zadanie 1

Skrypt prosi użytkownika o podanie liczby $x \in \mathbb{R}$, a następnie liczy jej logarytm naturalny $\ln x$ jeżeli $x > 0$. W przeciwnym wypadku zwraca błąd. Poniżej przykładowe wywołanie skryptu dla $x = 5$.

```
x = input('x = ');

if x > 0
    fprintf('ln(x) = %d\n', log(x));
else
    error('Input must be positive');
end
```

```
ln(x) = 1.609438e+00
```

Zadanie 2

Podpunkt a)

Funkcja wyznacza silnię $n!$ z liczby $n \in \mathbb{N}$.

```
function result = my_factorial(n)
    if n < 0 || floor(n) ~= n
        error('Input must be a natural number');
    elseif n == 0 || n == 1
        result = 1;
    else
        result = n * my_factorial(n - 1);
    end
end
```

Poniżej przykładowe wywołania funkcji

```
my_factorial(5)
```

```
ans =
120
```

```
my_factorial(0)
```

```
ans =
1
```

```
my_factorial(-3)
```

```
Error using solutions>my_factorial (line 10)
Input must be a natural number
```

```
my_factorial(12.3)
```

```
Error using solutions>my_factorial (line 10)
Input must be a natural number
```

Podpunkt b)

Funkcja wyznacza symbol Newtona $\binom{n}{k}$ z liczb $n, k \in \mathbb{N}$ takich, że $0 \leq k \leq n$.

```
function result = my_binomial(n, k)
    if k < 0 || k > n || floor(n) ~= n || floor(k) ~= k
        error('Inputs must satisfy 0 ≤ k ≤ n, with n and k being natural
numbers');
    end
    result = my_factorial(n) / (my_factorial(k) * my_factorial(n - k));
end
```

Poniżej przykładowe wywołania funkcji

```
my_binomial(5, 3)
```

```
ans =
10
```

```
my_binomial(3, 5)
```

```
Error using solutions>my_binomial (line 23)
Inputs must satisfy 0 ≤ k ≤ n, with n and k being natural numbers
```

```
my_binomial(3, -5)
```

```
Error using solutions>my_binomial (line 22)
Inputs must satisfy 0 ≤ k ≤ n, with n and k being natural numbers
```

Zadanie 3

Podpunkt a)

Funkcja rozwiązuje równanie liniowe $ax + b = 0$ dla argumentów a i b.

```
function root = linear_equation(a, b)
    if a == 0
        error('Not a linear equation');
    end

    root = -b / a;
end
```

Poniżej przykładowe wywołania funkcji

```
linear_equation(2, 1)
```

```
ans =
-0.5000
```

```
linear_equation(-5, 3)
```

```
ans =
0.6000
```

```
linear_equation(0, 4)
```

```
Error using solutions>linear_equation (line 32)
Not a linear equation
```

Podpunkt b)

Funkcja znajduje pierwiastki rzeczywiste równania kwadratowego $ax^2 + bx + c = 0$ dla argumentów a, b i c.

```
function roots = quadratic_equation(a, b, c)
    if a == 0
        error('Not a quadratic equation');
    end

    delta = b^2 - 4*a*c;

    if delta > 0
        x1 = (-b + sqrt(delta)) / 2*a;
        x2 = (-b - sqrt(delta)) / 2*a;
        roots = [x1, x2];
    elseif delta == 0
        x0 = -b / 2*a;
        roots = x0;
    else
        disp('No real root');
        roots = NaN;
    end
end
```

Poniżej przykładowe wywołania funkcji

```
quadratic_equation(1, 5, 6)
```

```
ans = 1×2
    -2    -3
```

```
quadratic_equation(-1, -1, 6)
```

```
ans = 1×2
    -3     2
```

```
quadratic_equation(1, 0, 0)
```

```
ans =
0
```

```
quadratic_equation(0, -5, 6)
```

```
Error using solutions>quadratic_equation (line 42)
Not a quadratic equation
```