Сложение

function add_numbers(u::Vector{Int}, v::Vector{Int}, b::Int)

```
n = max(length(u), length(v))
u_padded = [u; zeros(Int, n - length(u))]
v_padded = [v; zeros(Int, n - length(v))]
w = Int[]
carry = 0
for j in 1:n
    t = u_padded[j] + v_padded[j] + carry
    push!(w, t % b)
    carry = t ÷ b
end
if carry > 0
    push!(w, carry)
end
# Удаляем ведущие нули
while length(w) > 1 && last(w) == 0
    pop!(w)
end
return w
```

end

Вычитание (u >= v)

function subtract_numbers(u::Vector{Int}, v::Vector{Int}, b::Int)

```
n = max(length(u), length(v))
u_padded = [u; zeros(Int, n - length(u))]
v_padded = [v; zeros(Int, n - length(v))]
w = Int[]
borrow = 0
for j in 1:n
    t = u_padded[j] - v_padded[j] - borrow
    if t < 0
        t += b
        borrow = 1
    else
        borrow = 0
    end
    push!(w, t)
end
# Удаляем ведущие нули
```

```
    while length(w) > 1 && last(w) == 0
        pop!(w)
    end
    return w
```

end

Умножение (столбиком)

function multiply_numbers(u::Vector{Int}, v::Vector{Int}, b::Int)

```
    m = length(v)
    n = length(u)
    w = zeros(Int, m + n)
    for j in 1:m
        if v[j] == 0
            continue
        end
        k = 0
        for i in 1:n
            idx = i + j - 1
            t = v[j] * u[i] + w[idx] + k
            w[idx] = t % b
            k = t ÷ b
        end
        w[j + n] = k
    end
    # Удаляем ведущие нули
    while length(w) > 1 && last(w) == 0
        pop!(w)
    end
    return w
```

end

Быстрое умножение

function fast_multiply(u::Vector{Int}, v::Vector{Int}, b::Int)

```
    m = length(v)
    n = length(u)
    w = zeros(Int, m + n)
    t = 0
    for s in 0:(m + n - 1)
        for i in 0:s
```

```
            u_idx = n - i
            v_idx = m - (s - i)
            if u_idx < 1 || v_idx < 1 || u_idx > length(u) || v_idx > length(v)
                continue
            end
            t += u[u_idx] * v[v_idx]
        end
        w_idx = m + n - s
        if w_idx > length(w)
            continue
        end
        w[w_idx] = t % b
        t ÷= b
    end
    # Удаляем ведущие нули
    while length(w) > 1 && last(w) == 0
        pop!(w)
    end
    return w
```

end

Вспомогательные функции для тестирования

number_to_digits(n, b) = n == 0 ? [0] : reverse(digits(n, base=b))

digits_to_number(digits, b) = sum(d * b^(i-1) for (i, d) in enumerate(reverse

(digits)))

Пример использования

u = number_to_digits(243, 10) # [3,4,2] (обратный порядок)

v = number_to_digits(99, 10) # [9,9]

w_add = add_numbers(u, v, 10)

println("Сложение: ", digits_to_number(w_add, 10)) # 342

w_sub = subtract_numbers(u, v, 10)

println("Вычитание: ", digits_to_number(w_sub, 10)) # 144

w_mul = multiply_numbers(u, v, 10)

println("Умножение (столбиком): ", digits_to_number(w_mul, 10)) # 24057

w_fast_mul = fast_multiply(u, v, 10)

```
println("Быстрое умножение: ", digits_to_number(w_fast_mul, 10)) # 24057
```