

Аннотированные статьи и ресурсы

1. Get started with Julia	Официальный сайт языка Julia предлагает множество ресурсов для начинающих, включая видеоуроки, документацию и учебные материалы.
2. Tutorials - Julia	-Список учебных материалов, включая вводные курсы, интенсивные мастер-классы и руководства по использованию Julia в науке и инженерии.
3. Books - Julia	Список книг по Julia, написанных членами сообщества, которые охватывают различные аспекты языка.
4. Julia. Знакомство / Хабр	Сайт с вводной информацией по языку программирования Julia. На данном сайте можно ознакомиться с основными свойствами данного языка, синтаксисом и так далее.
5. Краткое описание языка программирования Julia	Целью данной работы является демонстрация возможностей языка программирования Julia для решения научных и технических задач. Представлены краткие сведения о языке, приведены примеры его использования.

Примеры решения задач

Пример 1: Вычисление факториала числа

```
1. # Функция для вычисления факториала числа
2. function factorial(n::Int)
3.     # Если n равно 0, возвращаем 1 (по определению факториала)
4.     if n == 0
5.         return 1
6.     else
7.         # Иначе возвращаем n умноженное на факториал (n-1)
8.         return n * factorial(n - 1)
9.     end
10.end
11.
12.# Пример использования функции
13.println(factorial(5)) # Вывод: 120
```

Пример 2: Нахождение максимального элемента в массиве

```
# Функция для нахождения максимального элемента в массиве
function find_max(arr::Array{Int, 1})
    # Инициализируем переменную max значением первого элемента массива
    max = arr[1]
    # Проходим по всем элементам массива
    for i in 2:length(arr)
        # Если текущий элемент больше max, обновляем max
        if arr[i] > max
            max = arr[i]
        end
    end
    # Возвращаем максимальный элемент
    return max
end

# Пример использования функции
arr = [3, 5, 7, 2, 8, 6]
println(find_max(arr)) # Вывод: 8
```

c = a*b

Пример 3: Сортировка массива методом пузырька

```
# Функция для сортировки массива методом пузырька
function bubble_sort(arr::Array{Int, 1})
    # Получаем длину массива
    n = length(arr)
    # Внешний цикл проходит по всем элементам массива
    for i in 1:n-1
        # Внутренний цикл для сравнения соседних элементов
        for j in 1:n-i
            # Если текущий элемент больше следующего, меняем их местами
            if arr[j] > arr[j+1]
                arr[j], arr[j+1] = arr[j+1], arr[j]
            end
        end
    end
    # Возвращаем отсортированный массив
    return arr
end

# Пример использования функции
arr = [64, 34, 25, 12, 22, 11, 90]
println(bubble_sort(arr)) # Вывод: [11, 12, 22, 25, 34, 64, 90]
```