# Rust Основы

# Арифметические операции

let number = 5.0;

let square = number.powi(2); // Возведение в степень

let square\_root = number.sqrt(); // Квадратный корень

# Ресурсы

Официальная документация: <https://doc.rust-lang.org/>

Книга по Rust: <https://doc.rust-lang.org/book/>

Справочник: <https://doc.rust-lang.org/reference/>

# Типы переменных

let int\_num = 42;

let str\_num = int\_num.to\_string(); // Конвертация в строку

// Узнать тип (во время компиляции)

std::any::type\_name::<T>() // Где T – тип

# Функции с возвратом нескольких значений

fn scale\_etc(vers: &HashMap<String, Vec<Vec<f64>>>, ver: &str, maxx: f64, maxy: f64)

-> (f64, f64, f64, f64, f64) {

let (verminx, verminy, scale, vercenx, verceny) = /\* логика \*/;

(verminx, verminy, scale, vercenx, verceny)

}

// Использование

let (verminx, verminy, scale, vercenx, verceny) = scale\_etc(&vers, ver, maxx, maxy);

# Классы (структуры и реализации)

struct MainWindow {

e3: E3Application, // Поле экземпляра

}

impl MainWindow {

// Конструктор

pub fn new(e3: E3Application) -> Self {

MainWindow { e3 }

}

// Метод экземпляра

pub fn show(&self) {

// Логика

}

}

# Структуры данных

## Статический массив

let devids: [i32; 3] = [0; 3]; // Массив из 3 нулей

## Вектор (динамический массив)

let mut numbers = Vec::new();

numbers.push(10);

numbers.extend(&[20, 30, 40]);

## Многомерный массив

let mut vers2 = [[[0.0; 12]; 12]; // 12x12

vers2[0][0][0] = 3.14;

## Зубчатый массив

let mut vers2: Vec<Vec<Vec<f64>>> = vec![vec![vec![0.0; 12]; 12]; // 12x12

vers2[0][0][0] = 3.14;

## HashMap (словарь) в Rust

### Создание и добавление элементов:

use std::collections::HashMap;

// 1. Создание пустого HashMap

let mut capitals = HashMap::new();

// 2. Добавление элементов

capitals.insert("Russia", "Moscow");

capitals.insert("USA", "Washington, D.C.");

// 3. Альтернативное создание с начальными значениями

let mut ages = HashMap::from([

("Alice", 25),

("Bob", 30),

]);

### Удаление элементов:

rust

Copy

// Удаление по ключу

capitals.remove("Russia");

// Очистка всего HashMap

capitals.clear();

### Доступ к элементам:

// Получение значения по ключу

if let Some(capital) = capitals.get("USA") {

println!("Столица США: {}", capital);

}

// Проверка наличия ключа

if capitals.contains\_key("France") {

println!("Франция есть в словаре");

}

### Итерация по HashMap:

// Итерация по парам ключ-значение

for (country, capital) in &capitals {

println!("Страна: {}, Столица: {}", country, capital);

}

// Только ключи

for country in capitals.keys() {

println!("Страна: {}", country);

}

// Только значения

for capital in capitals.values() {

println!("Столица: {}", capital);

}

### Обновление значений:

// Вставка с проверкой существования

capitals.entry("Germany").or\_insert("Berlin");

// Обновление существующего значения

if let Some(capital) = capitals.get\_mut("USA") {

\*capital = "New York";