# Отчёт по лабораторной работе №6

## Цель работы:

Целью данной лабораторной работы является освоение работы с замыканиями и итераторами в языке программирования Rust. Программа должна включать создание вектора структур, фильтрацию данных с помощью замыканий, а также использование итераторов для сортировки, фильтрации и преобразования данных.

## Задание:

Разработать программу, которая выполняет следующие задачи:

1. Создаёт вектор структур, представляющих людей (с полями `name` и `age`).
2. Использует замыкания для фильтрации людей старше определенного возраста и выводит их на экран.
3. Применяет итераторы для выполнения операций: сортировки, фильтрации и преобразования данных (например, превращение имени в верхний регистр).
4. Реализует сложный итератор, который работает с кортежами, где каждый кортеж состоит из имени и возраста.
5. Пишет функцию, которая возвращает новый итератор с замыканием для выполнения кастомной обработки элементов.

## Код программы:

```rust
// Определяем структуру Person
#[derive(Debug)]
struct Person {
name: String,
age: u32,
}

// Функция для создания вектора людей
fn create_people() -> Vec<Person> {
vec![
Person {
name: String::from("Alice"),
age: 30,
},
Person {
name: String::from("Bob"),
age: 25,
},
Person {
name: String::from("Charlie"),
age: 35,
```

```
},
Person {
name: String::from("Diana"),
age: 22,
},
]
}

// Функция для фильтрации и обработки людей
fn filter_and_process(people: &[Person], min_age: u32) {
let filtered: Vec<String> = people
.iter()
.filter(|p| p.age > min_age) // Фильтруем по возрасту
.map(|p| p.name.to_uppercase()) // Превращаем имена в верхний регистр
.collect();

println!("Люди старше {} лет: {:?}", min_age, filtered);
}

// Определяем итератор, работающий с кортежами
fn custom_iterator(people: Vec<Person>) -> impl Iterator<Item = (String, u32)> {
people.into_iter().map(|p| (p.name, p.age))
}

// Основная функция
fn main() {
let people = create_people();

// Фильтруем и обрабатываем
filter_and_process(&people, 25);

// Используем кастомный итератор
let person_tuples: Vec<(String, u32)> = custom_iterator(people).collect();
println!("Кортежи (имя, возраст): {:?}", person_tuples);
}
```

## Пример выполнения программы:

```
vonrodinus@VonRodinus:~/Projects/Lab6$ cargo run
   Compiling Lab6 v0.1.0 (/home/vonrodinus/Projects/Lab6)
    Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.14s
     Running `target/debug/Lab6`
Люди старше 25 лет: ["ALICE", "CHARLIE"]
Кортежи (имя, возраст): [("Alice", 30), ("Bob", 25), ("Charlie", 35), ("Diana", 22)]
vonrodinus@VonRodinus:~/Projects/Lab6$
```

## Вывод:

В результате выполнения лабораторной работы была успешно реализована программа, использующая замыкания и итераторы для фильтрации,

преобразования и вывода данных. Программа демонстрирует основы работы с коллекциями и итераторами в языке Rust, позволяя эффективно манипулировать данными и производить их обработку с использованием функциональных подходов.