

Университет ИТМО

Факультет программной инженерии и компьютерной техники

**Лабораторная работа №1**  
по «Алгоритмам и структурам данных»  
Базовые задачи

Выполнил:

Студент группы Р3209

Кулагин В.Д.

Преподаватели:

Косяков М.С.

Тараканов Д.С.

Санкт-Петербург

2025

## Задача №N «Свинки-копилки»

```
#include <bits/stdc++.h>
```

```
void process_cycle(int start, const std::vector<int>& a, std::vector<bool>& visited, int& cycles) {
```

```
    std::stack<int> stack;
```

```
    std::vector<bool> on_stack(a.size(), false);
```

```
    int current = start;
```

```
    while (!visited[current]) {
```

```
        if (on_stack[current]) {
```

```
            cycles++;
```

```
            break;
```

```
        }
```

```
        stack.push(current);
```

```
        on_stack[current] = true;
```

```
        current = a[current];
```

```
    }
```

```
    while (!stack.empty()) {
```

```
        visited[stack.top()] = true;
```

```
        stack.pop();
```

```
    }
```

```
}
```

```
int main() {
```

```
    int n;
```

```
    std::cin >> n;
```

```
    std::vector<int> a(n);
```

```
    for (int i = 0; i < n; i++) {
```

```
        std::cin >> a[i];
```

```
        a[i]--;
```

```
    }
```

```
    std::vector<bool> visited(n, false);
```

```
    int cycles = 0;
```

```
    for (int i = 0; i < n; i++) {
```

```
        if (!visited[i]) {
```

```
            process_cycle(i, a, visited, cycles);
```

```
    }  
}  
  
std::cout << cycles;  
return 0;  
}
```

Пояснение к примененному алгоритму:

Представляем задачу в виде перестановок на графе. Далее в алгоритме проходимся по всему графу (по каждой копилке), для каждой новой (еще не обойденной копилке) выполняем обход всех ее путей по ключам; обход осуществляется либо пока не будет найдена уже посещенная копилка (значит цикл обработан), либо пока не будет обнаружен цикл. Если нашли цикл, увеличиваем переменную, помечаем все элементы в цепочке как уже посещенные. Эта переменная – ответ.

Сложность алгоритма линейная, потому что проходимся по всему графы лишь однажды, т.е.  $O(n)$