

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«Московский авиационный институт  
(Национальный исследовательский университет)»

Факультет информационных технологий и прикладной математики  
Кафедра вычислительной математики и программирования

ОТЧЁТ  
по лабораторной работе № 5  
по курсу «Операционные системы»

Студент: Князьков Николай Дмитриевич

Группа: М80–203БВ–24

Преподаватель: Соколов А. А.

Оценка:\_\_\_\_\_

Дата:\_\_\_\_\_

Подпись:\_\_\_\_\_

Москва, 2025

# Лабораторная работа № 5

## 1.1 Постановка задачи

Цель работы: анализ использования системных вызовов в программе ЛР1 (деление чисел через pipe) с помощью инструментов трассировки (**strace** на Linux или Process Monitor на Windows).

Задание: провести диагностику программы ЛР1 и идентифицировать все системные вызовы, происходящие при работе программы. Проанализировать назначение каждого вызова и подготовить краткий отчёт.

## 1.2 Общие сведения о программе

Анализируемая программа: ЛР1 (родительский и дочерний процесс, обмен через pipe). Основные системные вызовы, зарегистрированные с помощью strace: **fork()**, **pipe()**, **dup2()**, **open()**, **read()**, **write()**, **wait4()**, **close()**, **exit()**.

Сначала создаётся pipe(), затем fork(). Дочерний процесс выполняет чтение из pipe, запись в файл (через open/write). Родитель записывает данные в pipe и ожидает завершения дочернего процесса.

## 1.3 Метод и алгоритм анализа

С помощью команды:

```
1 strace ./parent
```

получена последовательность системных вызовов. Пример:

```
1 pipe([3,4]) = 0
2 fork() = 1234
3 close(3)
4 write(4, "100 5 2\n", 8) = 8
5 wait4(1234, NULL, 0, NULL) = 1234
6 dup2(3, 0)
7 close(3)
8 close(4)
9 execl("./child", "child", NULL)
10 read(0, "100 5 2\n", 8)
```

```

11 open("result.txt", O_WRONLY|O_CREAT, 0666)
12 write(5, "20\n", 3)
13 write(5, "10\n", 3)
14 exit(0)

```

Вывод позволяет убедиться, что используются только стандартные системные вызовы, без лишних операций.

## 1.4 Основные файлы анализа

Пример логов strace:

```

1 // 
2 pipe([3, 4]) = 0          // (fd 3
4)
3 fork() = 1234           //
4
5 close(3)                // 
6 write(4, "100 5 2\n", 7) // pipe
7 wait4(1234, NULL, 0, NULL) = 1234 // pipe
8
9 dup2(3, 0)               // stdin
10 close(3); close(4);
11 execl("./child", "child", NULL)
12 read(0, "100 5 2\n", 7) // pipe
13 open("result.txt", O_WRONLY|O_CREAT, 0666) = 5
14 write(5, "20\n10\n", 6) // 
15
16 exit(0)

```

## 1.5 Пример работы

Пример ключевых строк лога:

```

1 pipe([3,4]) = 0
2 fork() = 5678
3 close(3)
4 write(4, "25 5\n", 5)

```

```
5 read(0, "25 5\n", 5)
6 open("result.txt", O_WRONLY|O_CREAT, 0666) = 5
7 write(5, "5\n", 2)
8 write(5, "1\n", 2)
9 exit(0)
10 wait4(5678, NULL, 0, NULL) = 5678
```

## 1.6 Выводы

Анализ показал, что в программе ЛР1 используются стандартные вызовы работы с процессами и вводом/выводом: `pipe()` – создание канала связи, `fork()` – порождение процесса, `dup2()` – переназначение потоков, `read()/write()` – обмен данными через pipe, `open()/write()` – запись в файл, `wait()` – ожидание завершения дочернего процесса. Никаких лишних вызовов не обнаружено. Работа над ЛР5 позволила понять, какие ресурсы ОС задействуются при межпроцессном взаимодействии и убедиться в правильности их использования.