Лабораторная работа №4: Реализация с использованием программных каналов (pipe)

Листинг программы (родительский процесс)

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/wait.h>

#include <string.h>

#include <fcntl.h>

#define BUF\_SIZE 256

int main(int argc, char \*argv[]) {

if (argc < 3) {

fprintf(stderr, "Usage: %s <char> <file1> [file2 ...]\n", argv[0]);

return EXIT\_FAILURE;

}

if (strlen(argv[1]) != 1) {

fprintf(stderr, "Error: Second argument must be a single character\n");

return EXIT\_FAILURE;

}

int num\_files = argc - 2;

int pipes[num\_files][2];

// Создаем каналы для каждого дочернего процесса

for (int i = 0; i < num\_files; ++i) {

if (pipe(pipes[i]) == -1) {

perror("pipe");

return EXIT\_FAILURE;

}

}

// Запускаем дочерние процессы

for (int i = 0; i < num\_files; ++i) {

pid\_t pid = fork();

if (pid == -1) {

perror("fork");

return EXIT\_FAILURE;

} else if (pid == 0) {

close(pipes[i][1]); // Закрываем запись в родителе

// Перенаправляем stdin на чтение из канала

dup2(pipes[i][0], STDIN\_FILENO);

close(pipes[i][0]);

execl("./lab4\_child", "lab4\_child", argv[1], argv[i+2], NULL);

perror("execl");

exit(255);

} else {

close(pipes[i][0]); // Закрываем чтение в родителе

// Отправляем данные в канал: имя файла и символ

char data[BUF\_SIZE];

snprintf(data, BUF\_SIZE, "%s %s", argv[1], argv[i+2]);

write(pipes[i][1], data, strlen(data) + 1);

close(pipes[i][1]);

}

}

// Ожидаем завершения дочерних процессов

for (int i = 0; i < num\_files; ++i) {

int status;

wait(&status);

if (WIFEXITED(status)) {

int code = WEXITSTATUS(status);

if (code == 255) {

printf("File: %s - Error\n", argv[i+2]);

} else {

printf("File: %s - Replacements: %d\n", argv[i+2], code);

}

}

}

return EXIT\_SUCCESS;

}

Листинг программы (дочерний процесс lab4\_child.c)

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <ctype.h>

#define BUF\_SIZE 256

int main() {

char data[BUF\_SIZE];

read(STDIN\_FILENO, data, BUF\_SIZE);

char symbol;

char filename[BUF\_SIZE];

sscanf(data, "%c %s", &symbol, filename);

FILE \*file = fopen(filename, "r+");

if (!file) {

perror("fopen");

return 255;

}

int replaced = 0;

int c;

while ((c = fgetc(file)) != EOF) {

if (isdigit(c)) { // Пример: замена цифр на символ

fseek(file, -1, SEEK\_CUR);

fputc(symbol, file);

replaced++;

}

}

fclose(file);

return replaced;

}

### Ответы на контрольные вопросы

1. **Каналы** — механизм межпроцессного взаимодействия (IPC).  
   **Отличия**:
   * Неименованные: Существуют только во время работы процессов, создаются через pipe().
   * Именованные (FIFO): Существуют как файлы в файловой системе, создаются через mkfifo().
2. **Файлы** требуют явного управления (открытие/закрытие), данные сохраняются на диске. **Каналы** работают в памяти, обеспечивают потоковую передачу данных между процессами.
3. **Создание неименованных каналов**:

int pipe(int fd[2]); // fd[0] — чтение, fd[1] — запись

1. Использование:
   * Родитель и потомок обмениваются данными через read()/write().
2. **Отличие от файлов**:
   * Каналы буферизируются ядром, не сохраняют данные после закрытия.
   * Операции чтения/записи блокируются при отсутствии данных/места.
3. **Функция**popen():  
   Создает канал и запускает процесс. Возвращает файловый указатель для чтения/записи.  
   Пример:

FILE \*fp = popen("ls", "r");

1. **Именованные каналы**:
   * Создаются через mkfifo("name", 0666).
   * Удаляются как файлы (unlink("name")).
   * Могут использоваться несвязанными процессами.
2. **Отличия в работе**:
   * Именованные каналы требуют открытия как файлы (open()).
   * Неименованные доступны только процессам-потомкам.
3. **Блокировки**:
   * При чтении из пустого канала или записи в полный.
   * Для именованных каналов: блокировка open() без флага O\_NONBLOCK.
4. **Обход блокировок**:  
   Использование флага O\_NONBLOCK при открытии канала:

int fd = open("fifo", O\_RDONLY | O\_NONBLOCK);

**11.Особенности функций**:

* + При неблокирующем режиме read()/write() возвращают ошибку, если операция невозможна.
  + select()/poll() позволяют отслеживать готовность канала.