#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/wait.h>

#include <string.h>

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

// Проверка количества аргументов

if (argc < 3) {

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

return EXIT\_FAILURE;

}

// Проверка, что второй аргумент - один символ

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

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

return EXIT\_FAILURE;

}

int num\_files = argc - 2;

char \*\*files = &argv[2];

// Создание дочерних процессов для каждого файла

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

pid\_t pid = fork();

if (pid == -1) {

perror("fork");

return EXIT\_FAILURE;

} else if (pid == 0) {

// Дочерний процесс: запуск программы lab2

execl("./lab2", "lab2", files[i], argv[1], NULL);

// Если execl вернул ошибку

perror("execl");

exit(255); // Код 255 для обозначения ошибки

}

}

// Ожидание завершения всех дочерних процессов

int total\_replaced = 0;

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

int status;

pid\_t child\_pid = wait(&status);

if (child\_pid == -1) {

perror("wait");

return EXIT\_FAILURE;

}

// Обработка статуса завершения

if (WIFEXITED(status)) {

int exit\_code = WEXITSTATUS(status);

if (exit\_code == 255) {

printf("File: %s - Error (-1)\n", files[i]);

} else {

printf("File: %s - Replacements: %d\n", files[i], exit\_code);

}

} else {

printf("File: %s - Process terminated abnormally\n", files[i]);

}

}

return EXIT\_SUCCESS;

}

Пояснения:

1. **Проверка аргументов**: Программа проверяет, что передано минимум два аргумента (символ и хотя бы один файл).
2. **Валидация символа**: Убеждаемся, что второй аргумент — это один символ.
3. **Создание процессов**: Для каждого файла создаётся дочерний процесс, который запускает программу lab2 с аргументами (имя файла и символ).
4. **Обработка ошибок**: Если вызовы fork или execl завершаются неудачей, выводятся соответствующие сообщения.
5. **Ожидание завершения**: Родительский процесс ждёт завершения всех дочерних процессов и выводит результаты их работы.

Требования к программе lab2:

* Должна принимать два аргумента: входной файл и символ.
* Возвращать количество замен (через exit(count)) или 255 в случае ошибки.
* Пример кода для lab2 (согласно варианту задания):

#include <stdio.h>

#include <stdlib.h>

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

if (argc != 3 || strlen(argv[2]) != 1) {

fprintf(stderr, "Usage: %s <file> <character>\n", argv[0]);

exit(255);

}

FILE \*file = fopen(argv[1], "r+");

if (!file) {

perror("fopen");

exit(255);

}

// Логика замены символов (пример: замена всех цифр на заданный символ)

int replaced = 0;

int c;

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

if (isdigit(c)) {

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

fputc(argv[2][0], file);

replaced++;

}

}

fclose(file);

exit(replaced); // Возвращаем количество замен

}