

Московский Авиационный Институт
(Национальный Исследовательский Университет)
Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Лабораторная работа №2 по курсу
«Операционные системы»

Тема работы
«Изучение взаимодействий между процессами»

Студентка: Клитная Анастасия
Викторовна
Группа: М8О-208Б-20
Вариант: 20
Преподаватель:
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2021

Содержание

1. Репозиторий
2. Постановка задачи
3. Общие сведения о программе
4. Общий метод и алгоритм решения
5. Исходный код
6. Демонстрация работы программы
7. Выводы

Репозиторий

https://github.com/klitnaya/OS_2

Постановка задачи

Родительский процесс создает два дочерних процесса. Первой строкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия File с таким именем на запись для child1. Аналогично для второй строки и процесса child2. Родительский и дочерний процесс должны быть представлены разными программами.

Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1 или в pipe2 в зависимости от правила фильтрации. Процесс child1 и child2 производят работу над строками. Процессы пишут результаты своей работы в стандартный вывод.

Правило фильтрации: строки длины больше 10 символов отправляются в pipe2, иначе в pipe1. Дочерние процессы инвертируют строки

Общие сведения о программе

Вся программа содержится в одном файле laba_2.cpp

Общий метод и алгоритм решения

Запуск осуществляется при помощи ввода в командную строку unix:

./laba2

Исходный код

Добавьте исходный код вашей программы

```
#include <iostream>
#include <string>
#include <fstream>
#include "unistd.h"

int main(){
    std::string th_child1;
    std::string th_child2;
    std::cout << "this is parent process" << std::endl;
```

```

std::cout << "enter names for first and second childs" << std::endl;
std::cin >> th_child1;
std::cin >> th_child2;
std::fstream fs;
int fd1[2];
pipe(fd1);

int fd2[2];
pipe(fd2);

if (pipe(fd1) == -1){
    std::cout << "error" << std::endl;
    return 1;
}
if (pipe(fd2) == -1){
    std::cout << "error" << std::endl;
    return 1;
}
int first_id = fork();
if (first_id == -1){
    std::cout << "Error" << std::endl;
    return -1;
}
else if (first_id == 0){ //work with 1 child (len <= 10 words)
    fs.open(th_child1, std::fstream::in | std::fstream::out | std::fstream::app);
    int a=0;
    read(fd1[0], &a, sizeof(int));
    std::cout << "your in child 1 process" << std::endl;
    while (a > 0){
        int size;
        read(fd1[0], &size, sizeof(int));
        char array[size];
        read(fd1[0], array, sizeof(char)*size);
        std::string string;
        for (int i =0; i<size; i++){
            string.push_back(array[i]);
        }
        for(int i = 0; i < size/2; i++){ //invert

            char tmp = string[i];
            string[i] = string[size-i-1];
            string[size-i-1] = tmp;
        }
        fs << string << std::endl;
        std::cout << "After work in child 1 your string look as: " << string << std::endl;
        a = a - 1;
    }
    close(fd1[0]);
    close(fd1[1]);
}

```

```

else
{
    int second_id = fork();
    if (second_id == -1){
        std::cout<<"error"<<std::endl;
        return -1;
    }
    else if (second_id == 0){ //work with 2 child (when > 10 words)
        fs.open(th_child2, std:: fstream:: in | std:: fstream:: out | std:: fstream:: app);
        int a;
        read(fd2[0], &a, sizeof(int));
        std:: cout << "your in child 2 process"<< std:: endl;
        while(a>0){
            int size;
            read(fd2[0], &size, sizeof(int));
            char array[size];
            read(fd2[0], array, sizeof(char)* size);
            std::string string;
            for (int i = 0; i<size; i++){
                string.push_back(array[i]);
            }
            for (int i = 0; i<size/2; i++){ //invert
                char tmp = string[i];
                string[i] = string[size-i-1];
                string[size-i-1] = tmp;
            }
            fs<<string<<std:: endl;
            std::cout<<"After your work in child 2 your string look as: " <<string << std:: endl;
            a = a - 1;
        }
        close(fd2[0]);
        close(fd2[1]);
    }
    else{
        int a;
        std:: cout<<"PARENT: please, enter number of string"<< std:: endl;
        std:: cin >> a;
        write(fd1[1], &a, sizeof(int));
        write(fd2[1], &a, sizeof(int));
        std::cout <<"PARENT: please, enter your strings "<< a <<"time"<<std:: endl;//time == pas
        for(int i = 0; i<a; i++){
            std:: string string;
            std:: cin >> string;
            int num = string.size();
            char array[num];
            for (int i = 0; i<num; i++){
                array[i] = string[i];
            }
            if (string.size() <= 10){
                write(fd1[1], &num, sizeof(int));
            }
        }
    }
}

```

```
        write(fd1[1], array, sizeof(char)*num);
    }
    else {
        write(fd2[1], &num, sizeof(int));
        write(fd2[1], array, sizeof(char)*num);
    }
}

close(fd1[0]);
close(fd1[1]);
close(fd2[0]);
close(fd2[1]);
}

}
return 0;
}
```

Выводы

Я приобрела навыки в управлении процессами в ОС Unix и обеспечении обмена данных между процессами при помощи каналов.