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”

Студент: Борисов Ян Артурович
Группа: М8О-208Б-20
Вариант: 19
Преподаватель: Миронов Евгений Сергеевич
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2021

Содержание

Репозиторий

https://github.com/Yannikupy/OS/tree/master/os_lab2

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: с вероятностью 80% строки отправляются в pipe1, иначе в pipe2. Дочерние процессы удаляют все гласные из строк.

Общие сведения о программе

Вся программа содержится в одном файле main.cpp

Общий метод и алгоритм решения

При помощи вызова fork создаются два процесса.

В родительском процессе вновь вызывается fork, теперь активны 3 процесса – два дочерних и родительский.

Родитель считывает строки string, конвертирует в c-style строки и отсылает их по правилу сортировки в дочерние процессы. Они, в свою очередь, осуществляют требуемую операцию и выводят строки в файл.

Исходный код

Добавьте исходный код вашей программы (для вставки кода в Microsoft Word можно взять способ здесь).

main.cpp

```
#include "unistd.h"
#include <iostream>
#include <string>
#include <fstream>
int main(){
    std::string file1, file2;
    std::cout << "Enter names of files: " << std::endl;
    std::cin >> file1;
    std::cin >> file2;
    std::fstream fos;
    int fd1[2];
    pipe(fd1); //creating pipe from parent to child1
    int fd2[2];
    pipe(fd2); //creating pipe from parent to child2
    if (pipe(fd1) == -1) {
        std:: cout << "error with opening the pipe" << std:: endl;
        return 1;
    }
    if (pipe(fd2) == -1) {
        std:: cout << "error with opening the pipe" << std:: endl;
        return 2;
    }
    int cid1 = fork(); //creating child process 1
    if (cid1 == -1)
    {
        std:: cout << "fork error" << std:: endl;
        return -1;
    }
    else if (cid1 == 0) // --------------------[CHILD1]--------------------
    {
        fos.open(file1, std::fstream::in | std::fstream::out | std::fstream::app);
        int N;
        read(fd1[0], &N, sizeof(int));//reading number of strings
        while(N>0){
            int size;
            read(fd1[0], &size, sizeof(int));
            char strc[size];
            read(fd1[0], strc, sizeof(char) * size);
            std::string str;
            for (int i = 0; i < size; i++){
                str.push_back(strc[i]);
            };
            int j = 0;
            while (j < str.size()) {
                while ((str[j] == 'a') or (str[j] == 'e') or (str[j] == 'i') or (str[j] == 'o') or (str[j] == 'u') or (str[j] == 'y') or
                       (str[j] == 'A') or (str[j] == 'E') or (str[j] == 'I') or (str[j] == 'O') or (str[j] == 'U') or (str[j] == 'Y'))
                {
                    str.erase(str.begin() + j);
                }
                j++;
            };
            fos << str << std::endl;
            std::cout << "[CHILD1] " << str << std:: endl;
            N--;
        };
        close(fd1[0]);
        close(fd1[1]);
    }// --------------------[CHILD1]--------------------
    else
    {
        int cid2 = fork();
        if (cid2 == -1)
        {
            std:: cout << "Fork error!" << std:: endl;
            return -2;
```

```
66.         }
67.         else if (cid2 == 0) {// --------------------[CHILD2]--------------------
68.             fos.open(file2, std::fstream::in | std::fstream::out | std::fstream::app);
69.             int N;
70.             read(fd2[0], &N, sizeof(int));
71.             while(N>0){
72.                 int size;
73.                 read(fd2[0], &size, sizeof(int));
74.                 char strc[size];
75.                 read(fd2[0], strc, sizeof(char) * size);
76.                 std::string str;
77.                 for (int i = 0; i < size; i++){
78.                     str.push_back(strc[i]);
79.                 };
80.                 int j = 0;
81.                 while (j < str.size()) {
82.                     while ((str[j] == 'a') or (str[j] == 'e') or (str[j] == 'i') or (str[j]
    == 'o') or (str[j] == 'u') or (str[j] == 'y') or
83.                            (str[j] == 'A') or (str[j] == 'E') or (str[j] == 'I') or (str[j]
    == 'O') or (str[j] == 'U') or (str[j] == 'Y'))
84.                     {
85.                         str.erase(str.begin() + j);
86.                     }
87.                     j++;
88.                 }
89.                 fos << str << std:: endl;
90.                 std::cout << "[CHILD2] " << str << std:: endl;
91.                 N--;
92.             };
93.             close(fd2[0]);
94.             close(fd2[1]);
95.         }// --------------------[CHILD2]--------------------
96.
97.         else {// --------------------[PARENT]--------------------
98.             int N;
99.             std::cout << "[PARENT] Enter number of strings" << std::endl;
100.              std::cin >> N;
101.              write(fd1[1], &N, sizeof(int));
102.              write(fd2[1], &N, sizeof(int));
103.              std::cout <<"[PARENT] Enter " << N << " string(s): " << std::endl;
104.              for (int i = 0; i < N; i++){
105.                  std::string str1;
106.                  std::cin >> str1;
107.                  int k = str1.size();
108.                  char strtothr1[k];
109.                  for (int i = 0; i < k; i++){
110.                      strtothr1[i] = str1[i];
111.                  }
112.
113.                  if (rand() % 100 + 1 <= 80) { //80% chance to throw text to child 1
114.                      write(fd1[1], &k, sizeof(int));
115.                      write(fd1[1], strtothr1, sizeof(char) * k);
116.                  }
117.
118.                  else{
119.                      write(fd2[1], &k, sizeof(int));
120.                      write(fd2[1], strtothr1, sizeof(char) * k);
121.                  }
122.
123.              }
124.              close(fd1[0]);
125.              close(fd1[1]);
126.              close(fd2[0]);
127.              close(fd2[1]);
128.          }// --------------------[PARENT]--------------------
129.
130.      }
131.      return 0;
132.  }
```

Демонстрация работы программы

Enter names of files:

file1

file2

[PARENT] Enter number of strings

5

[PARENT] Enter 5 string(s):

dakdakkada

[CHILD2] dkdkkd

fldfldlf

[CHILD2] fldfldlf

abadsa

[CHILD1] bds

ewpkfwcscsza

[CHILD1] wpkfwcscsz

dewvdfverlvpe

[CHILD2] dwvdfvrlvp

File1:
bds

wpkfwcscsz

File2:
dkdkkd

fldfldlf

dwvdfvrlvp

Выводы

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