**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика» Кафедра: 806 «Вычислительная математика и программирование» Дисциплина: «Операционные системы»

**Лабораторная работа № 2**

Студент: Зверев М.Е.

Группа: М8О-306Б-19

Преподаватель: Соколов А.А.

Дата:

Оценка:

Москва, 2020

**Цель работы**

Приобретение практических навыков в:

● Управлении процессами в ОС

● Обеспечение обмена данных между процессами посредством каналов

**Задание**

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решение задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системные сигналы/события и/или каналы (pipe). Необходимо обрабатывать системные ошибки, которые могут возникнуть в результате работы. **Вариант:**

Пользователь вводит команды вида: «число число число<endline>». Далее эти числа передаются от родительского процесса в дочерний. Дочерний процесс считает их сумму и выводит её в файл. Числа имеют тип float.

**Общие сведения о программе**

Программа компилируется из файла laba.c. Для реализации поставленной задачи в программе используются следующие системные вызовы: ***pipe*** *-* создает однонаправленный канал данных, который можно использовать для взаимодействия между процессами.

***fork*** *-* создает копию текущего процесса, который является дочерним процессом для текущего процесса.

***open*** *-* открывает файл. Аргументом можно задать, открыть файл на запись или чтение*.*

***close*** *-* закрывает файл.

***write*** *-* записывает в файл из области памяти указанное количество байт. ***read*** *-* читает из файла указанное количество байт в области памяти. ***dup2*** *-* создает дубликат файлового дескриптора.

***perror*** *-* вывод сообщения об ошибке.

**Листинг программы**

**parent.c**

**#include <stdlib.h>**

**#include <stdio.h>**

**#include <unistd.h>**

**#include <string.h>**

**int main()**

**{**

**char \*name = NULL;**

**name = malloc(256 \* sizeof(char));**

**for (int i = 0; i < 256; i++)**

**name[i] = 0;**

**char str[255];**

**printf("Введите имя файла ");**

**fgets(name, 255, stdin);**

**fgets(str, 255, stdin);**

**while (str[0] != '\*')**

**{**

**int fd[2];**

**if (pipe(fd) == -1)**

**{**

**printf("Error creating pipe");**

**return 2;**

**}**

**int id = fork();**

**if (id == -1)**

**{**

**printf("Error creating process");**

**return 3;**

**}**

**else if (id == 0)**

**{ //дочерний процесс**

**close(fd[1]);**

**read(fd[0], str, 255);**

**execl("child.out", " ", name, str, NULL);**

**close(fd[0]);**

**}**

**else**

**{ //родительский процесс**

**close(fd[0]);**

**write(fd[1], str, 255);**

**close(fd[1]);**

**}**

**fgets(str, 255, stdin);**

**}**

**return 0;**

**}**

**child.c**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <unistd.h>**

**#include <string.h>**

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

**{**

**float sum = 0, num = 0, infrac = 0;**

**char c;**

**char \*name = NULL;**

**name = malloc(sizeof(char) \* 255);**

**for (int i = 0; i < 256; i++)**

**name[i] = 0;**

**int j = 0;**

**for (int i = 0; i < 256; i++)**

**{**

**if ((argv[1][i] != '\'') && (argv[1][i] != '\n') && (argv[1][i] != '$'))**

**{**

**name[j] = argv[1][i];**

**j++;**

**}**

**}**

**int minus = 0, i = 0;**

**while (argv[2][i] != '\0')**

**{**

**c = argv[2][i];**

**if (c == ' ' || c == '\t')**

**{**

**sum = minus ? sum - num : sum + num;**

**num = 0;**

**minus = 0;**

**infrac = 0;**

**}**

**else if (c == '-')**

**{**

**minus = 1;**

**}**

**else if (c == '.' || c == ',')**

**{**

**infrac = 10;**

**}**

**else if (c >= '0' && c <= '9')**

**{**

**if (infrac == 0)**

**num = num \* 10 + c - '0';**

**else**

**{**

**float f = (c - '0') / infrac;**

**num = num + f;**

**infrac \*= 10;**

**}**

**}**

**i++;**

**}**

**sum = minus ? sum - num : sum + num;**

**FILE \*filename;**

**if ((filename = fopen(name, "a")) == NULL)**

**{**

**printf("Cannot open file");**

**return 0;**

**}**

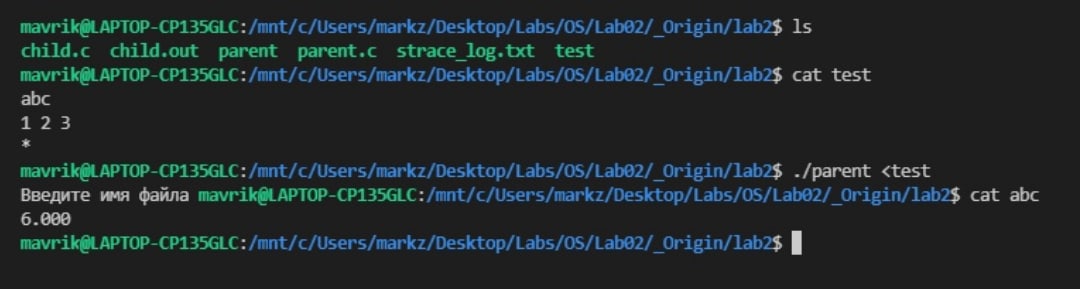
**fprintf(filename, "%.3f\n", sum);**

**fclose(filename);**

**return 0;**

**}**

**Пример работы**



**Вывод**

Существуют специальные системные вызовы(fork) для создания процессов, а специальные каналы pipe позволяют связать процессы для обмена данными. При использовании fork важно помнить, что фактически создается копию текущего процесса и неправильная работа может привести к неожиданным результатам и последствиям, однако создание процессов очень удобно, когда нужно выполнять несколько действий параллельно. Также у каждого процесса есть свой id, по которому его можно определить. При работе с чтением и записью из файла важно помнить, что read и write возвращают количество успешно считанных/записанных байт и оно не обязательно равно тому значению, которое вы указали. И, конечно, важно не забывать, что открыв файл при помощи open, его обязательно надо будет впоследствии закрыть с помощью close.