**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное**

**учреждение высшего образования**

**Национальный исследовательский университет**

**«Высшая школа экономики»**

Департамент прикладной математики

**ОТЧЁТ**

**К ДОМАШНЕЙ РАБОТЕ №2**

**по дисциплине**

**«Операционные системы»**

Работу выполнил

студент группы БПМ-192 И.И. Пчелинцев

подпись, дата

Работу проверил А.А. Внуков

подпись, дата

Москва 2021

**Оглавление**

Введение…………………………………………………………………………………………………….2

Постановка задачи……………………………………………………………………………………..2

Основная часть…………………………………………………………………………………………..2

Литература………………………………………………………………………………………………….4

**Введение**

Операционная система - комплекс программ, обеспечивающий управление аппаратными средствами компьютера, организующий работу с файлами и выполнение прикладных программ, осуществляющий ввод и вывод данных.

На сегодняшний день, операционная система — это первый и основной набор программ, загружающийся в компьютер. Помимо вышеуказанных функций ОС может осуществлять и другие, например предоставление общего пользовательского интерфейса.

Сегодня наиболее известными операционными системами являются ОС семейства: 1)Microsoft Windows

2) UNIX-подобные системы

**Постановка задачи**

6. chown

17. geteuid

56. execlp

**Основная часть**

|  |  |  |  |
| --- | --- | --- | --- |
| **Системный вызов** | **Назначение** | **Аргументы** | **Возвращаемые значения** |
| **chown**  **fchown**  **lchown** | Владелец файла, который задан параметром path или fd, будет изменен. Только суперпользователь может изменить владельца файла. Владелец файла может изменять группу файла на любую группу, к которой он принадлежит.  fchown()  работает с открытым файлом, а  lchown()  работает с символической ссылкой вместо файла, на который эта ссылка указывает. Во всех других отношениях эти три вызова работают идентично. | int chown(const char \*path, uid\_t owner, gid\_t group);  int fchown(int fd, uid\_t owner, gid\_t group);  int lchown(const char \*path, uid\_t owner, gid\_t group); | 0 успех  -1 ошибка, а переменной errno присваивается код ошибки. |
| **getuid**  **geteuid** | получить идентифкатор пользоватля | uid\_t getuid(void);  uid\_t geteuid(void); | getuid возвращает фактический идентификатор ID пользователя в текущем процессе.  geteuid возвращает эффективный идентификатор ID пользователя в текущем процессе.  Фактический ID соответствует ID пользователя, который вызвал процесс. Эффективный ID соответствует установленному setuid биту на исполняемом файле.  Функции всегда завершаются успешно. |
| **execlp**  **execle**  **execvp**  **execv**  **execve** | Функции, загружающие и запускающие другие программы.  Семейство функций exec... загружает и запускает другие программы, известные как "дочерние" процессы. Если вызов функции exec... завершается успешно, "дочерний" процесс накладывается на "родительский" процесс; причем должно быть достаточно памяти для загрузки и выполнения  "дочернего" процесса. pathname - это имя файла вызываемого "дочернего"  процесса.  Суффиксы l, v, p и e, добавляемые к имени семейств exec... обозначают, что данная функция будет работать с некоторыми особенностями:  p - определяет, что функция будет искать "дочернюю"  программу в директориях, определяемых  переменной среды DOS PATH. Без суффикса p поиск  будет производиться только в рабочем каталоге.  l - показывает, что адресные указатели (arg0, arg1,  ..., argn) передаются, как отдельные аргументы.  Обычно суффикс l употребляется, когда число пере-  даваемых аргументов заранее вам известно.  v - показывает, что адресные указатели (arg[0],  arg[1],...arg[n]) передаются, как массив  указателей. Обычно, суффикс v используется,  когда передаeтся переменное число аргументов.  e - показывает, что "дочернему" процессу может быть  передан аргумент envp, который позволяет  выбирать среду "дочернего" процесса. Без  суффикса e "дочерний" процесс унаследует среду  "родительского" процесса. | int execlp(const char \*file, const char \*arg, ...);  int execle(const char \*path, const char \*arg , ..., char \* const envp[]);  int execvp(const char \*file, char \*const argv[]);  int execv(const char \*path, char \*const argv[]);  int execve(const char \*filename, char \*const argv [], char  \*const envp[]); | При успешном завершении функции exec... не возвращают никакого значения. При возникновении ошибки функции exec... возвращают значение -1 |

**Литература**

1. Дейтел Г. Введение в ОС, т.1,2.- М.:Мир, 1987
2. Таненбаум С. Современные ОС.- С.-П.: Изд. Группа BHV, 2002