**Отчёт по лабораторной работе №10**

**Дисциплина: архитектура компьютеров**

Чувакина Мария Владимировна

**Содержание**

**1  Цель работы 4**  **2  Задание 5**  **3  Теоретическое введение 6**  **4  Выполнение лабораторной работы 8** т 4.1 **Написание программ для работы с файлами** . . . . . . . . . . ……..8 т 4.2 **Задание для самостоятельной работы** . . . . . . . . . . . . . ……….. 10

**5  Выводы**  **14**  **6  Список литературы 15**

**Список иллюстраций**

4.1  Создание файлов для лабораторной работы . . . . . . . . . . . ………………….. 8 4.2  Ввод текста программы из листинга 10.1 . . . . . . . . . . . . . ………………….. 8 4.3  Запуск исполняемого файла............................................................................... 9 4.4  Запрет на выполнение файла ............................................................................. 9 4.5  Добавление прав на исполнение ....................................................................... 9 4.6  Предоставление прав доступа в символьном и двоичном виде . . …………10 4.7  Написание текста программы........................................................................... 11 4.8  Запуск исполняемого файла и проверка его работы . . . . . . . …………….... 11

**1 Цель работы**

Приобретение навыков написания программ для работы с файлами.

**2 Задание**

1. Написание программ для работы с файлами.

2. Задание для самостоятельной работы.

**3 Теоретическое введение**

Права доступа определяют набор действий (чтение, запись, выполнение), раз- решённых для выполнения пользователям системы над файлами. Для каждого файла пользователь может входить в одну из трех групп: владелец, член группы владельца, все остальные. Для каждой из этих групп может быть установлен свой набор прав доступа.

Для изменения прав доступа служит команда chmod, которая понимает как символьное, так и числовое указание прав.

Обработка файлов в операционной системе Linux осуществляется за счет ис- пользования определенных системных вызовов. Для корректной работы и досту- па к файлу при его открытии или создании, файлу присваивается уникальный номер (16-битное целое число) – дескриптор файла.

Для создания и открытия файла служит системный вызов sys\_creat, который использует следующие аргументы: права доступа к файлу в регистре ECX, имя файла в EBX и номер системного вызова sys\_creat (8) в EAX.

Для открытия существующего файла служит системный вызов sys\_open, который использует следующие аргументы: права доступа к файлу в регистре EDX, режим доступа к файлу в регистр ECX, имя файла в EBX и номер системного вызова sys\_open (5) в EAX.

Для записи в файл служит системный вызов sys\_write, который использует следующие аргументы: количество байтов для записи в регистре EDX, строку содержимого для записи ECX, файловый дескриптор в EBX и номер системного вызова sys\_write (4) в EAX. Системный вызов возвращает фактическое количество записанных байтов в регистр EAX. В случае ошибки, код ошибки также будет находиться в регистре EAX. Прежде чем записывать в файл, его необходимо создать или открыть, что позволит получить дескриптор файла.

Для чтения данных из файла служит системный вызов sys\_read, который ис- пользует следующие аргументы: количество байтов для чтения в регистре EDX, адрес в памяти для записи прочитанных данных в ECX, файловый дескриптор в EBX и номер системного вызова sys\_read (3) в EAX. Как и для записи, прежде чем читать из файла, его необходимо открыть, что позволит получить дескриптор файла.

Для правильного закрытия файла служит системный вызов sys\_close, который использует один аргумент – дескриптор файла в регистре EBX. После вызова ядра происходит удаление дескриптора файла, а в случае ошибки, системный вызов возвращает код ошибки в регистр EAX.

Для изменения содержимого файла служит системный вызов sys\_lseek, который использует следующие аргументы: исходная позиция для смещения EDX, значение смещения в байтах в ECX, файловый дескриптор в EBX и номер системного вызова sys\_lseek (19) в EAX. Значение смещения можно задавать в байтах.

Удаление файла осуществляется системным вызовом sys\_unlink, который ис- пользует один аргумент – имя файла в регистре EBX.

**4 Выполнение лабораторной работы**

**4.1 Написание программ для работы с файлами**

Создаю каталог для программ лабораторной работы № 10, перехожу в него и создаю файлы lab10-1.asm, readme-1.txt и readme-2.txt. (рис. 4.1)

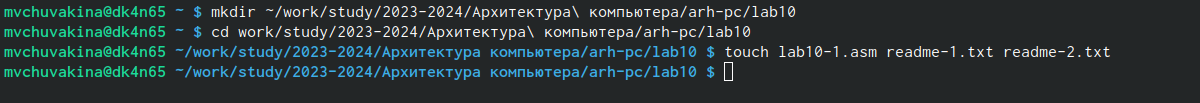


Рис. 4.1: Создание файлов для лабораторной работы

Ввожу в файл lab10-1.asm текст программы, записывающей в файл сообщения, из листинга 10.1. (рис. 4.2)

Изображение выглядит как текст, программное обеспечение, веб-страница, снимок экрана

Автоматически созданное описание

Рис. 4.2: Ввод текста программы из листинга 10.1

Создаю исполняемый файл и проверяю его работу. (рис. 4.3)

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание

Рис. 4.3: Запуск исполняемого файла

Далее с помощью команды chmod u-х изменяю права доступа к исполняемому файлу lab10-1, запретив его выполнение и пытаюсь выполнить файл. (рис. 4.4)

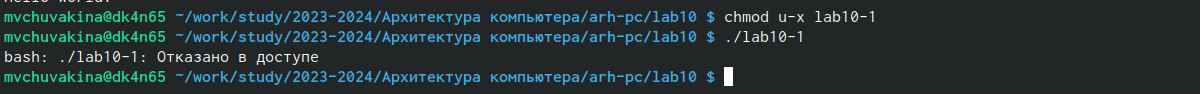


Рис. 4.4: Запрет на выполнение файла

Файл не выполняется, т.к в команде я указала “u” - владелец (себя), “-” - отменить набор прав, “х” - право на исполнение.  
С помощью команды chmod u+х изменяю права доступа к файлу lab10-1.asm с исходным текстом программы, добавив права на исполнение, и пытаюсь выполнить его. (рис. 4.5)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 4.5: Добавление прав на исполнение

Текстовый файл начинает исполнение, но не исполняется, т.к не содержит в себе команд для терминала.

В соответствии со своим вариантом (16) в таблице 10.4 предоставляю права доступа к файлу readme1.txt представленные в символьном виде, а для файла readme-2.txt – в двочном виде:

--x r-x -w-, 001 010 101  
И проверяю правильность выполнения с помощью команды ls -l. (рис. 4.6)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 4.6: Предоставление прав доступа в символьном и двоичном виде

**4.2 Задание для самостоятельной работы**

Пишу код программы, выводящей приглашения “Как Вас зовут?”, считывающей с клавиатуры фамилию и имя и создающую файл, в который записывается сообщение “Меня зовут”ФИ””. (рис. 4.7)

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рис. 4.7: Написание текста программы

Создаю исполняемый файл и проверяю его работу. Проверяю наличие файла и его содержимое с помощью команд ls и cat. (рис. 4.8)

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рис. 4.8: Запуск исполняемого файла и проверка его работы

Программа работает корректно. Код программы:

%include ‘in\_out.asm’  
SECTION .data msg1 db ‘Как Вас зовут?’, 0h filename db ‘name.txt’, 0h msg2 db ‘Меня зовут’, 0h SECTION .bss name resb 255 SECTION .text global \_start \_start: mov eax,msg1 call sprintLF mov ecx, name mov edx, 255 call sread mov ecx, 0777o mov ebx, filename mov eax, 8  
int 80h  
mov ecx, 2  
mov ebx, filename mov eax, 5  
int 80h  
mov esi, eax  
mov eax, msg2 call slen  
mov edx, eax  
mov ecx, msg2 mov ebx, esi  
mov eax, 4  
int 80h mov eax, name call slen  
mov edx, eax mov ecx, name mov ebx, esi mov eax, 4

int 80h  
mov ebx, esi mov eax, 6 int 80h  
call quit

**5 Выводы**

Благодаря данной лабораторной работе я приобрела навыки написания программ для работы с файлами.

**6 Список литературы**

1. GDB:The GNU Project Debugger.—URL:https://www.gnu.org/software/gdb/.
2. GNU Bash Manual.—2016.—URL:https://www.gnu.org/software/bash/manual/.
3. Midnight Commander Development Center.—2021.—URL:https://midnight-

commander. org/.

1. NASM Assembly Language Tutorials.—2021.—URL:https://asmtutor.com/.
2. Newham C. Learning the bash Shell: Unix Shell Programming. — O’Reilly

Media, 2005 — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL:

http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658.

1. Robbins A. Bash Pocket Reference. — O’Reilly Media, 2016. — 156 с. — ISBN

978-1491941591.

1. The NASM documentation.—2021.—URL:https://www.nasm.us/docs.php.
2. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
3. Колдаев В.Д., Лупин С.А. Архитектура ЭВМ.—М.:Форум,2018.
4. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. :

Солон-Пресс, 2017.

1. Новожилов О.П. Архитектура ЭВМ и систем.—М.:Юрайт,2016.
2. Расширенный ассемблер:NASM.—2021.—URL:https://www.opennet.ru/docs/RUS/nasm/.
3. Робачевский А., Немнюгин С.,Стесик О.Операционная система UNIX.—2-е

изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.

1. Столяров А. Программирование на языке ассемблера NASM для ОС Unix.—2-

е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm\_unix.

16

15. Таненбаум Э.Архитектура компьютера.—6-еизд.—СПб.:Питер,2013.— 874 с. — (Классика Computer Science).

16. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер,2015. — 1120 с. — (Классика Computer Science).