Лабораторная работа №10

Работа с файлами средствами Nasm

Богданюк Анна Васильевна

Содержание

# Цель работы

Приобретение навыков написания программ для работы с файлами.

# Задание

1. Выполнение лабораторной работы
2. Задания для самостоятельной работы

# Теоретическое введение

ОС GNU/Linux является многопользовательской операционной системой. И для обеспечения защиты данных одного пользователя от действий других пользователей существуют специальные механизмы разграничения доступа к файлам. Кроме ограничения доступа, данный механизм позволяет разрешить другим пользователям доступ данным для совместной работы. Права доступа определяют набор действий (чтение, запись, выполнение), разрешённых 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. Владельцем файла является его создатель. Для предоставления прав доступа другому пользователю или другой группе командой chown [ключи] [:новая\_группа] или chgrp [ключи] < новая\_группа >

Общий алгоритм работы с системными вызовами в Nasm можно представить в следующем виде: 1. Поместить номер системного вызова в регистр EAX; 2. Поместить аргументы системного вызова в регистрах EBX, ECX и EDX; 3. Вызов прерывания (int 80h); 4. Результат обычно возвращается в регистр EAX.

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 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 использует 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.

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

# Выполнение лабораторной работы

1. Выполнение лабораторной работы

Создаю каталог lab10 для программ лабораторной работы №10, перехожу в него и создаю файлы lab10-1.asm, readme-1.txt и readme-2.txt (рис. [1](#fig:001)).

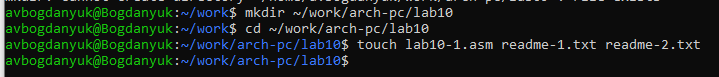


Figure 1: Создаю каталог и файлы

Ввожу в файл lab10-1.asm текст программы из листинга (рис. [2](#fig:002)).

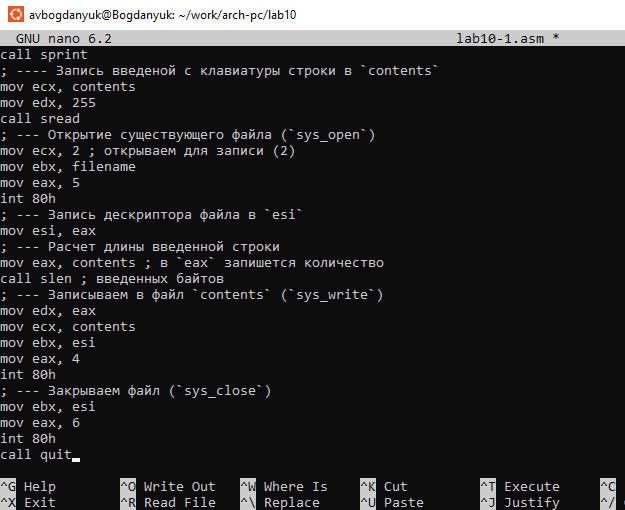


Figure 2: Файл lab10-1.asm

Листинг:

%include 'in\_out.asm'  
SECTION .data  
filename db 'readme.txt', 0h ; Имя файла  
msg db 'Введите строку для записи в файл: ', 0h ; Сообщение  
SECTION .bss  
contents resb 255 ; переменная для вводимой строки  
SECTION .text  
global \_start  
\_start:  
; --- Печать сообщения `msg`  
mov eax,msg  
call sprint  
; ---- Запись введеной с клавиатуры строки в `contents`  
mov ecx, contents  
mov edx, 255  
call sread  
; --- Открытие существующего файла (`sys\_open`)  
mov ecx, 2 ; открываем для записи (2)  
mov ebx, filename  
mov eax, 5  
int 80h  
; --- Запись дескриптора файла в `esi`  
mov esi, eax  
; --- Расчет длины введенной строки  
mov eax, contents ; в `eax` запишется количество  
call slen ; введенных байтов  
; --- Записываем в файл `contents` (`sys\_write`)  
mov edx, eax  
mov ecx, contents  
mov ebx, esi  
mov eax, 4  
int 80h  
; --- Закрываем файл (`sys\_close`)  
mov ebx, esi  
mov eax, 6  
int 80h  
call quit

Создаю исполнительный файл и запускаю его. Проверяю с помощью ls и cat. Программа выполняется корректно (рис. [3](#fig:003)).

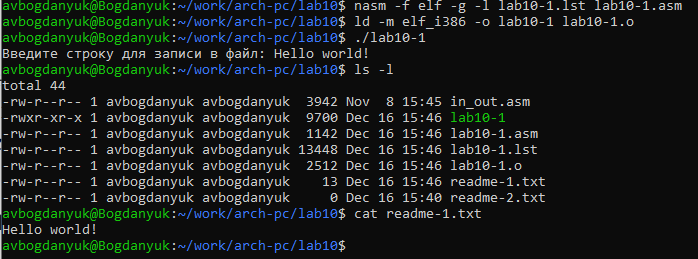


Figure 3: Создание, запуск и проверка

С помощью команды chmod изменяю права доступа к исполняемому файлу lab10-1 (a-x - для всех пользователей отменить право на исполнение), запретив его исполнение для всех. Пытаюсь выполнить его. Выводится сообщение ‘Permission denied’, потому что мне, как owner, запрещено исполнять файл (рис. [4](#fig:004)).

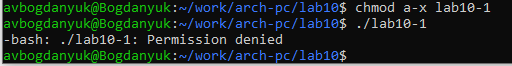


Figure 4: Изменение права доступа, проверка

С помощью chmod изменяю право доступа к файлу lab10-1.asm с исходным текстом программы, добавив права на исполнение (a+x - для всех пользователей добваить право на исполнение) (рис. [5](#fig:005)).

Figure 5: Изменение права доступа lab10-1.asm

Figure 5: Изменение права доступа lab10-1.asm

Пытаюсь выполнить lab10-1.asm. Выводится построчно содержимое файла (рис. [6](#fig:006)).

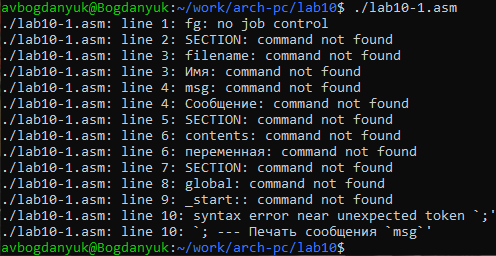


Figure 6: Запуск lab10-1.asm

Мой вариант - 4. Сначала изменяю права доступа к файлу readme-1.txt в символьном виде. Проверяю с помощью ls, все сработало корректно (рис. [7](#fig:007)).

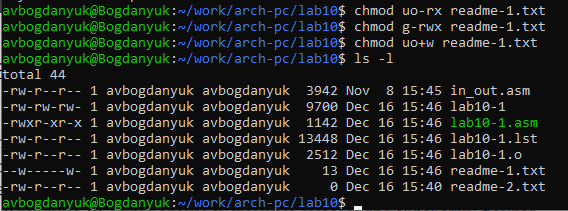


Figure 7: Изменение права доступа в символьном виде

Теперь изменяю права доступа к файлу readme-2.txt двоичном виде. 001 - 1, 011 - 3, 110 - 6. Проверяю с помощью ls, работает корректно(рис. [8](#fig:008)).

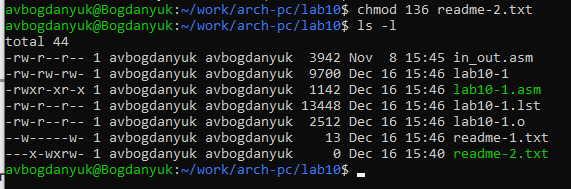


Figure 8: Изменение права доступа в двоичном виде

1. Задания для самостоятельной работы

Создаю файл lab10-2.asm, ввожу текст программы, чтобы:вывод приглашения “Как Вас зовут?”, ввести с клавиатуры свои фамилию и имя, создать файл с именем name.txt, записать в файл сообщение “Меня зовут”, дописать в файл строку введенную с клавиатуры, закрыть файл. (рис. [9](#fig:010)).

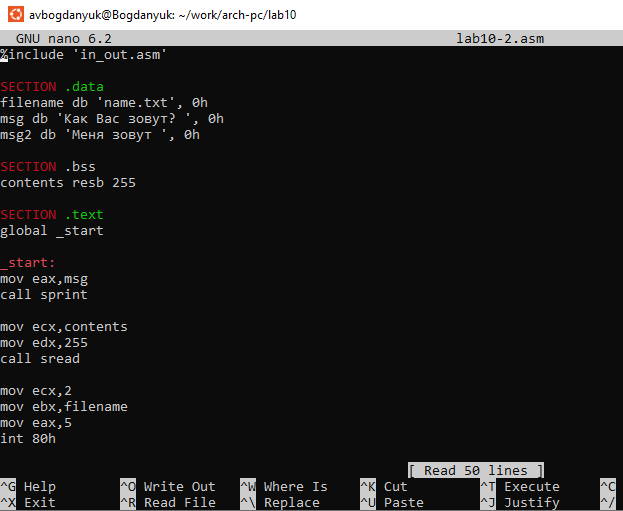


Figure 9: Файл lab10-2.asm

Листинг:

%include 'in\_out.asm'  
  
SECTION .data  
filename db 'name.txt', 0h  
msg db 'Как Вас зовут? ', 0h  
msg2 db 'Меня зовут ', 0h  
  
SECTION .bss  
contents resb 255  
  
SECTION .text  
global \_start  
  
\_start:  
mov eax,msg  
call sprint  
  
mov ecx,contents  
mov edx,255  
call sread  
  
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,contents  
call slen  
  
mov edx, eax  
mov ecx, contents  
mov ebx, esi  
mov eax, 4  
int 80h  
  
mov ebx, esi  
mov eax, 6  
int 80h  
call quit

Создаю исполняемый файл и запускаю его, проверяю с помощью cat. Работает корректно (рис. [10](#fig:009)).

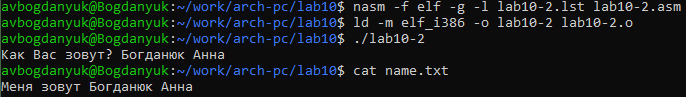


Figure 10: Создание и запуск

Проверяю с помощью ls (рис. [11](#fig:011)).

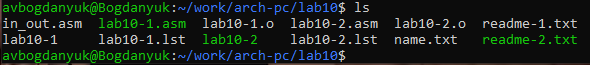


Figure 11: Проверка

# Выводы

В ходе выполнения лабораторной работы были приобретены навыки написания программ для работы с файлами.

# Список литературы