

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 10

дисциплина: *Архитектура компьютера*

Студент: Грицко Сергей

Группа: НКАбд-02-25

МОСКВА

2025 г.

Оглавление

Цель работы.....	3
Теоретическое введение	4
2.1 Права доступа к файлам в GNU/Linux	4
2.2 Системные вызовы NASM для работы с файлами.....	4
Описание результатов выполнения лабораторной работы	5
Список Литературы	9

Цель работы

Приобретение навыков написания программ для работы с файлами на языке ассемблера NASM. Изучение системных вызовов для создания, открытия, чтения, записи и закрытия файлов, а также команд изменения прав доступа.

Теоретическое введение

2.1 Права доступа к файлам в GNU/Linux

Операционная система GNU/Linux является многопользовательской и использует механизмы разграничения доступа для защиты данных. Права доступа определяют, какие действия разрешены трем категориям пользователей:

1. **Владелец** файла (u - user).
2. **Группа** владельца (g - group).
3. **Остальные** пользователи (o - other).

Для каждой категории устанавливается комбинация из трех базовых прав:

1. **r** (read) — чтение.
2. **w** (write) — запись/изменение.
3. **x** (execute) — выполнение (для программ) или просмотр содержимого каталога.

Права могут задаваться в **символьном виде** (например, u=rwx,g=rw,o=r) или в **восьмеричном виде** (где каждое право — это бит: r=4, w=2, x=1. Например, rwx = 4+2+1=7, rw- = 4+2+0=6). Для изменения прав доступа используется команда `chmod`.

2.2 Системные вызовы NASM для работы с файлами

В ассемблере NASM работа с файлами осуществляется через системные вызовы ядра Linux (int 80h), где номер вызова передается в регистре EAX.

Вызов	Номер EAX	Назначение	Регистры (Параметры)
sys_creat	8	Создание нового файла	EBX (имя файла), ECX (права доступа, octal)
sys_write	4	Запись данных в файл	EBX (дескриптор файла), ECX (адрес буфера), EDX (количество байтов)
sys_close	6	Закрытие файла	EBX (дескриптор файла)

Таблица 1. Системные вызовы ядра Linux

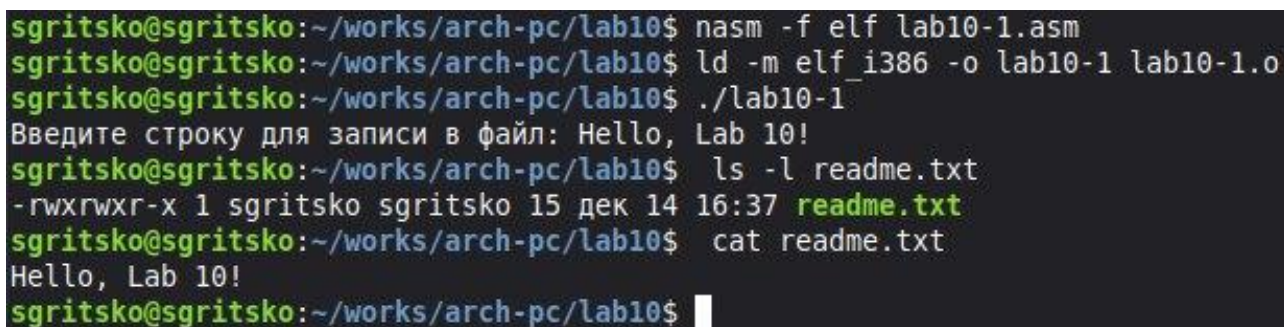
После успешного выполнения `sys_creat` дескриптор созданного файла (числовой идентификатор) возвращается в регистре EAX. Этот дескриптор затем используется в качестве параметра (EBX) для всех последующих операций (запись, чтение, закрытие).

Описание результатов выполнения лабораторной работы

Я создал каталог для работы и необходимые файлы с помощью команд `mkdir` и `touch`.

Далее я ввел текст программы из листинга 10.1 в файл `lab10-1.asm`. Программа запрашивает строку у пользователя и записывает ее в файл `readme.txt`.

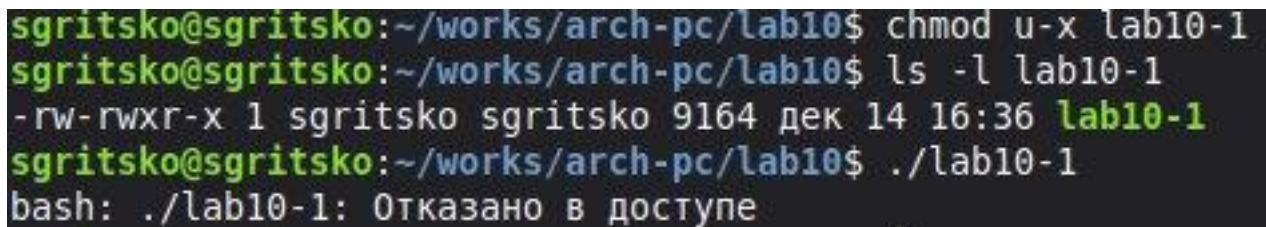
Я скомпилировал файл и запустил его.



```
sgritsko@sgritsko:~/works/arch-pc/lab10$ nasm -f elf lab10-1.asm
sgritsko@sgritsko:~/works/arch-pc/lab10$ ld -m elf_i386 -o lab10-1 lab10-1.o
sgritsko@sgritsko:~/works/arch-pc/lab10$ ./lab10-1
Введите строку для записи в файл: Hello, Lab 10!
sgritsko@sgritsko:~/works/arch-pc/lab10$ ls -l readme.txt
-rwxrwxr-x 1 sgritsko sgritsko 15 дек 14 16:37 readme.txt
sgritsko@sgritsko:~/works/arch-pc/lab10$ cat readme.txt
Hello, Lab 10!
sgritsko@sgritsko:~/works/arch-pc/lab10$
```

Рисунок 1. Компиляция и проверка работы программы

С помощью команды `chmod` я убрал право на исполнение у файла `lab10-1` (бинарного файла) и попытался его запустить.

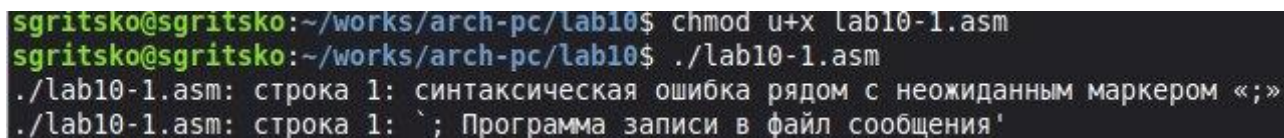


```
sgritsko@sgritsko:~/works/arch-pc/lab10$ chmod u-x lab10-1
sgritsko@sgritsko:~/works/arch-pc/lab10$ ls -l lab10-1
-rw-rwxr-x 1 sgritsko sgritsko 9164 дек 14 16:36 lab10-1
sgritsko@sgritsko:~/works/arch-pc/lab10$ ./lab10-1
bash: ./lab10-1: Отказано в доступе
```

Рисунок 2. Запрет выполнения

Вывод: Операционная система запретила выполнение файла, так как у текущего пользователя (владельца) отсутствует бит права на исполнение (x).

Я добавил права на исполнение исходному текстовому файлу с кодом `lab10-1.asm` и попытался его запустить.



```
sgritsko@sgritsko:~/works/arch-pc/lab10$ chmod u+x lab10-1.asm
sgritsko@sgritsko:~/works/arch-pc/lab10$ ./lab10-1.asm
./lab10-1.asm: строка 1: синтаксическая ошибка рядом с неожиданным маркером «;»
./lab10-1.asm: строка 1: `; Программа записи в файл сообщения'
```

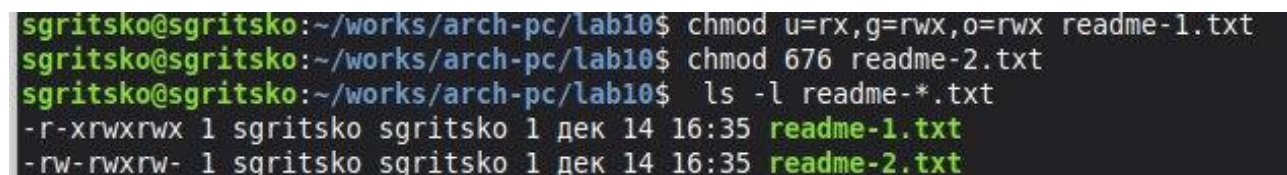
Рисунок 3. Запуск исходного кода

Вывод: Командная оболочка попыталась исполнить текстовый файл как скрипт. Поскольку

синтаксис ассемблера не является командами bash, возникли ошибки. Исходный код нужно сначала скомпилировать.

Согласно таблице вариантов (Вариант 14), я установил права доступа:

1. Для readme-1.txt (Символьный вид): r-x rwx rwx.
 - 1.1. Владелец: чтение и исполнение (rx)
 - 1.2. Группа: чтение, запись, исполнение (rwx)
 - 1.3. Остальные: чтение, запись, исполнение (rwx)
2. Для readme-2.txt (Двоичный вид): 110 111 110.
 - 2.1. $110 = 6$ (rw-)
 - 2.2. $111 = 7$ (rwx)
 - 2.3. $110 = 6$ (rw-)
 - 2.4. Итоговое восьмеричное значение: 676.



```
sgritsko@sgritsko:~/works/arch-pc/lab10$ chmod u=rx,g=rwx,o=rwx readme-1.txt
sgritsko@sgritsko:~/works/arch-pc/lab10$ chmod 676 readme-2.txt
sgritsko@sgritsko:~/works/arch-pc/lab10$ ls -l readme-*.txt
-r-xrwxrwx 1 sgritsko sgritsko 1 дек 14 16:35 readme-1.txt
-rw-rwxrw- 1 sgritsko sgritsko 1 дек 14 16:35 readme-2.txt
```

Рисунок 4. Установка прав

Описание результатов выполнения самостоятельной работы

Задание: Написать программу, которая:

1. Выводит приглашение "Как Вас зовут?".
2. Вводит с клавиатуры фамилию и имя.
3. Создает файл name.txt.
4. Записывает в файл сообщение "Меня зовут ".
5. Дописывает введенное имя.

Текст программы lab10-2.asm: (Смотри файл lab10-2.asm, приложенный к отчету)

```
sgritsko@sgritsko:~/works/arch-pc/lab10$ nasm -f elf lab10-2.asm
sgritsko@sgritsko:~/works/arch-pc/lab10$ ld -m elf_i386 -o lab10-2 lab10-2.o
sgritsko@sgritsko:~/works/arch-pc/lab10$ ./lab10-2
Как Вас зовут? Грицко Сергей
sgritsko@sgritsko:~/works/arch-pc/lab10$ ls -l name.txt
-rwxrwxr-x 1 sgritsko sgritsko 46 дек 14 16:49 name.txt
sgritsko@sgritsko:~/works/arch-pc/lab10$ cat name.txt
Меня зовут Грицко Сергей
```

Рисунок 5. Проверка работы самостоятельной программы

Выводы

В ходе выполнения лабораторной работы я научился работать с файлами в ОС Linux на языке ассемблера NASM. Я изучил системные вызовы `sys_creat`, `sys_open`, `sys_write`, `sys_close` и научился использовать дескрипторы файлов. Также я закрепил навыки управления правами доступа к файлам с помощью команды `chmod` в символьном и числовом режимах.

Список Литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
3. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
4. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : СолонПресс, 2017.
5. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. М. : МАКС Пресс, 2011. 6. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. 12