Отчёт по лабораторной работе №10

Дисциплина: архитектура компьютера

Федорова Анжелика Игоревна

Содержание

# 1 Цель работы

Приобретение навыков написания программ для работы с файлами.

# 2 Задание

Напишите программу работающую по следующему алгоритму: • 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

# 3 Теоретическое введение

С GNU/Linux является многопользовательской операционной системой. И для обеспече- ния защиты данных одного пользователя от действий других пользователей существуют специальные механизмы разграничения доступа к файлам. Кроме ограничения доступа, дан- ный 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.Набор прав доступа задается тройками битов и состоит из прав на чтение, запись и ис- полнение файла. В символьном представлении он имеет вид строк rwx, где вместо любого символа может стоять дефис. Всего возможно 8 комбинаций, приведенных в таблице 10.1. Буква означает наличие права (установлен в единицу второй бит триады r — чтение, первый бит w — запись, нулевой бит х — исполнение), а дефис означает отсутствие права (нулевое значение соответствующего бита). Также права доступа могут быть представлены как вось- меричное число. Так, права доступа rw- (чтение и запись, без исполнения) понимаются как три двоичные цифры 110 или как восьмеричная цифра 6.

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 полу- чить 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.

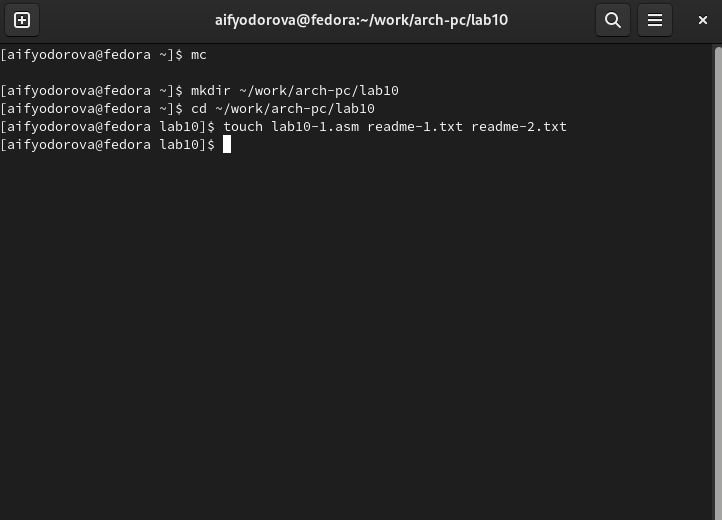
Для правильного закрытия файла служит системный вызов sys\_close, который использует один аргумент – дескриптор файла в регистре EBX. После вызова ядра происходит удаление дескриптора файла, а в случае ошибки, системный вызов возвращает код ошибки в регистр EAX.

Для изменения содержимого файла служит системный вызов sys\_lseek, который исполь- зует 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. В случае ошибки, системный вызов возвращает код ошибки в регистр EAX.

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

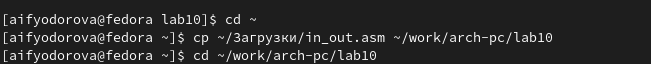
# 4 Выполнение лабораторной работы

Создадим каталог для программ лабораторной работы и создадим в нем необходимые файлы (рис. fig:001).



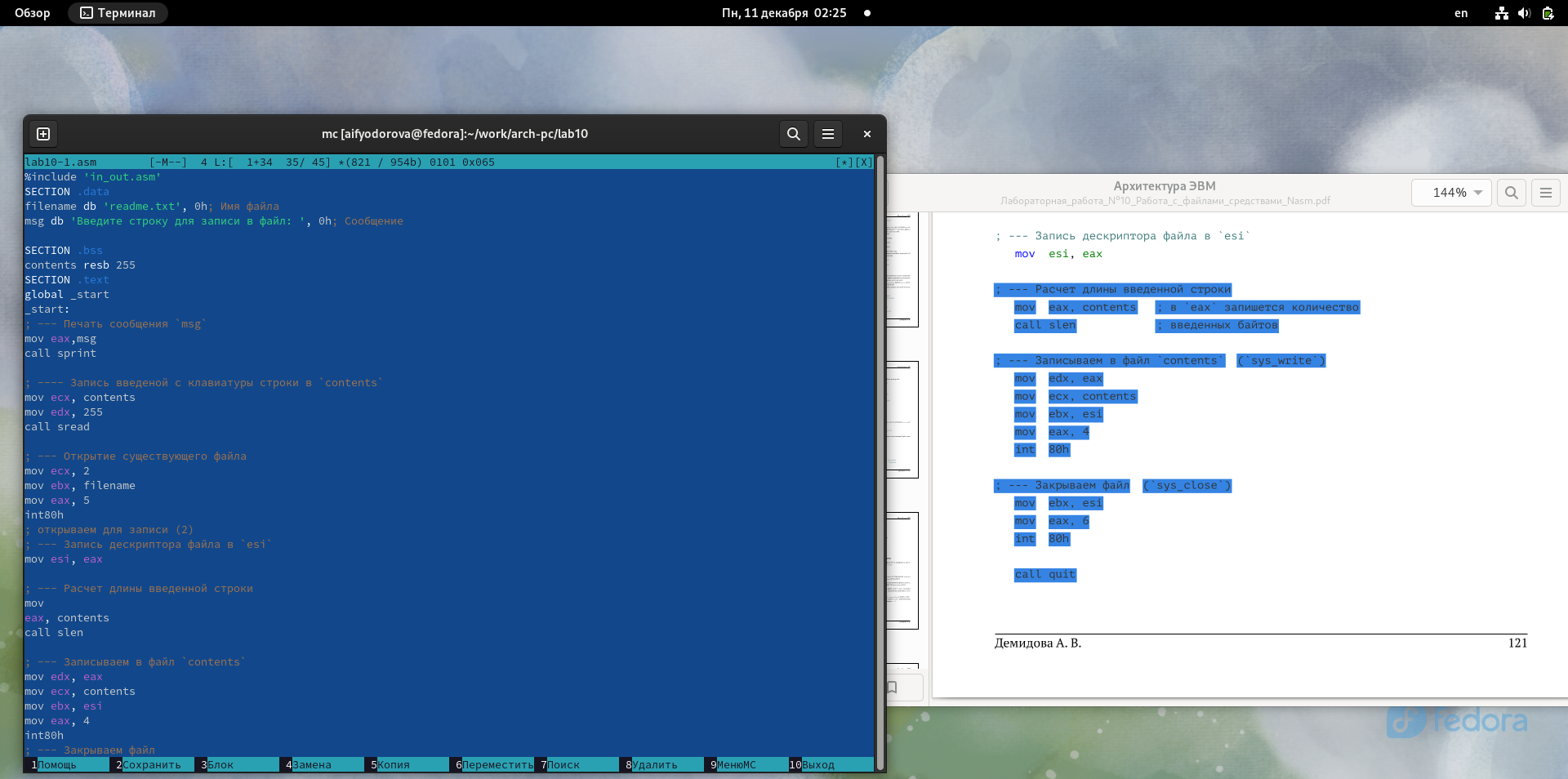
Cоздание нужного каталога и файла lab10-1.asm

Скопирую файл in-out.asm в нужный мне каталог. (рис. fig:002)



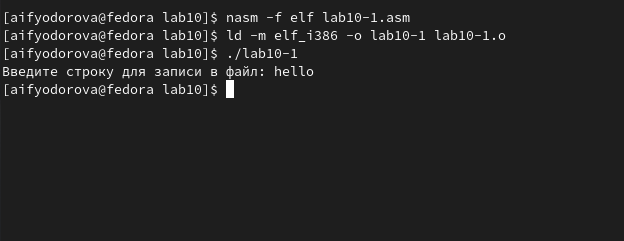
Копирование файла in-out.asm

Введём в файл lab10-1.asm текст программы из листинга (рис. fig:003).



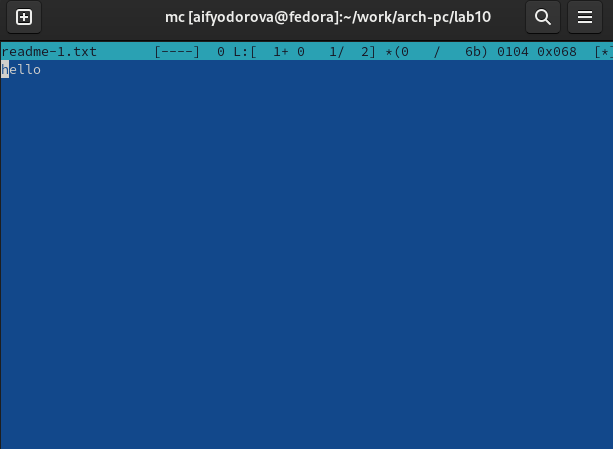
Заполнение файла

Создадим исполняемый файл и проверим его работу (рис. fig:004).



Создание исполняемого файла

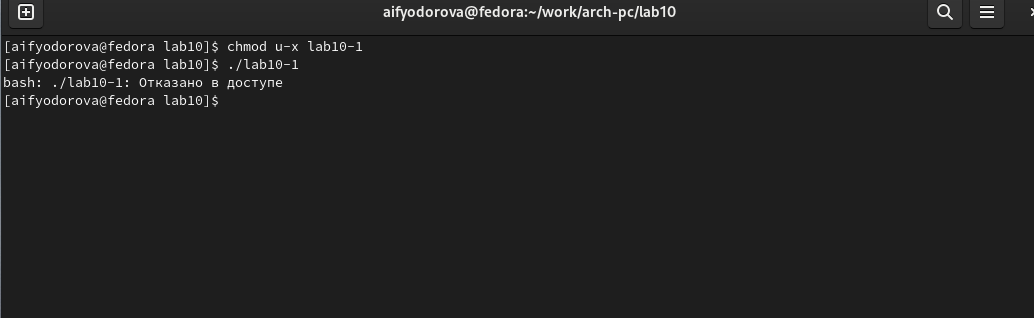
Теперь проверю содержимое файла readme-1.txt (рис.fig:005).



Проверка содержимого файла readme-1.txt

Вижу, что программа сработала успешно.

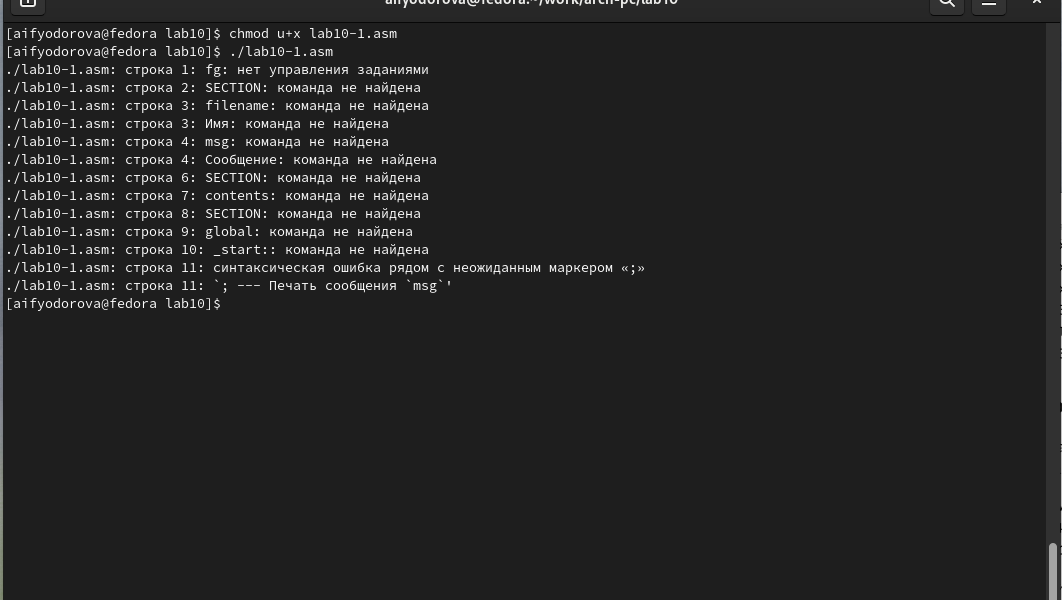
С помощью команды chmod изменим права доступа к исполняемому файлу lab10-1, запретив его выполнение (рис.fig:006).



Изменение прав доступа с помощью сhmod

Теперь доступ к файлу запрещён.

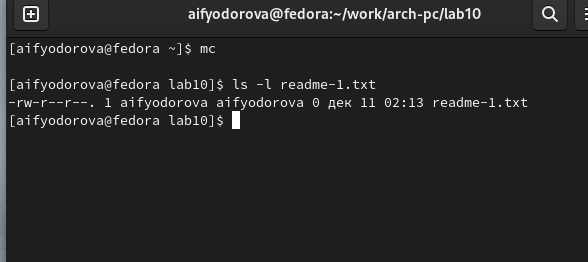
Изменим права доступа к файлу lab10-1.asm, добавив права на исполнение (рис. fig:007).



Изменение прав доступа к файлу lab10-1.asm

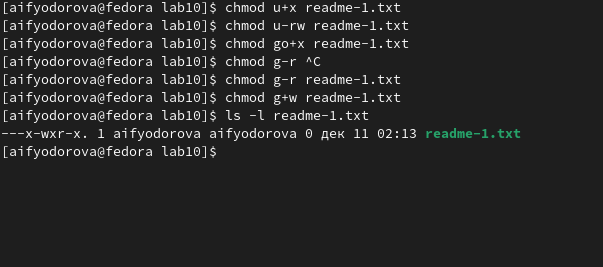
Файл нельзя выполнить, т.к. он не содержит указаний для терминала, а просто является текстом программы.

Проверю какие права имеются у пользователей файла readme-1.txt с помощью команды ls с ключом -l(рис. fig:008).



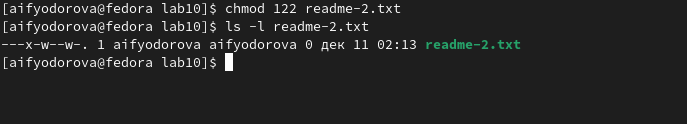
Проверка прав на доступ

Предоставим права доступа к этому файлу в соответствии с вариантом 12 (рис.fig:009).



Предоставление прав на доступ по варианту 12

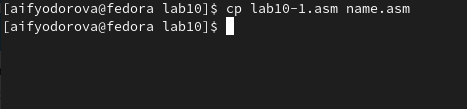
Проделаю то же самое, но с помощью двоичного кода с файлом readme-2.txt (рис.fig:010).



Предоставление прав на доступ для файла readme-2.txt

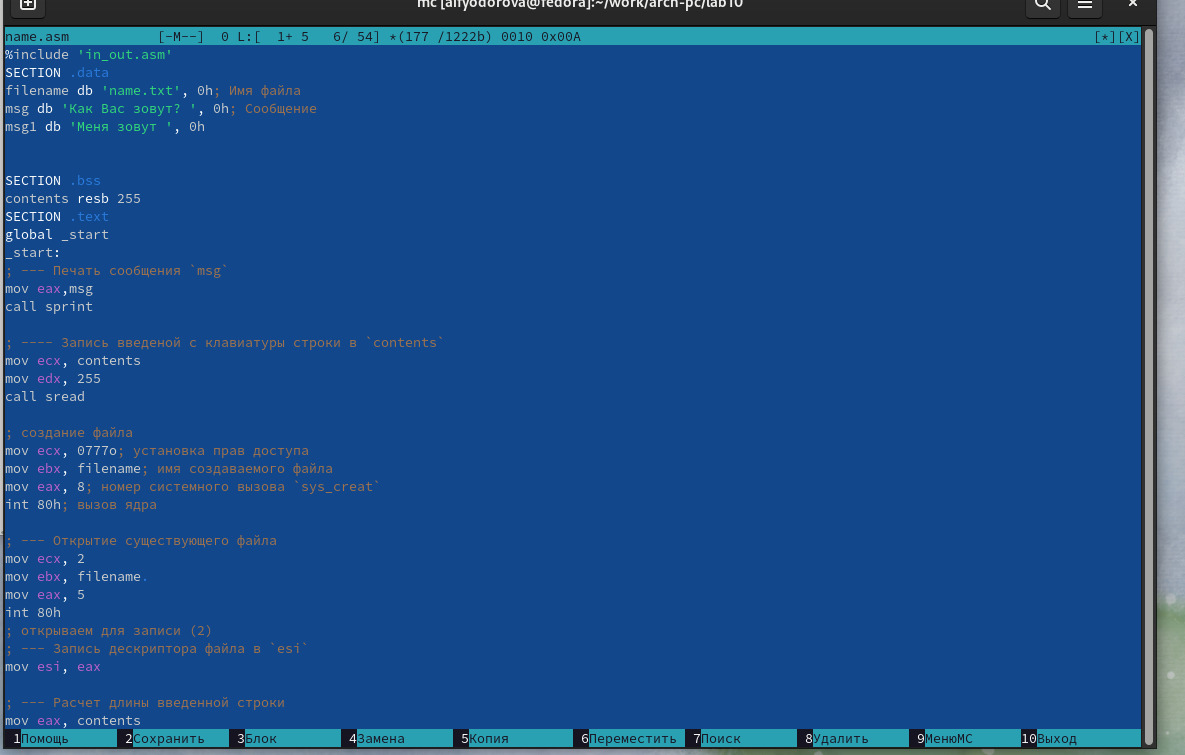
# 5 Самостоятельная работа

Скопирую содержимое файла lab10-1.asm в файла name.asm (рис.fig:011).



Копирование файла кода lab10-1.asm в name.asm

Редактирую код данного файла в соответствии с целью задания (рис.fig:012).



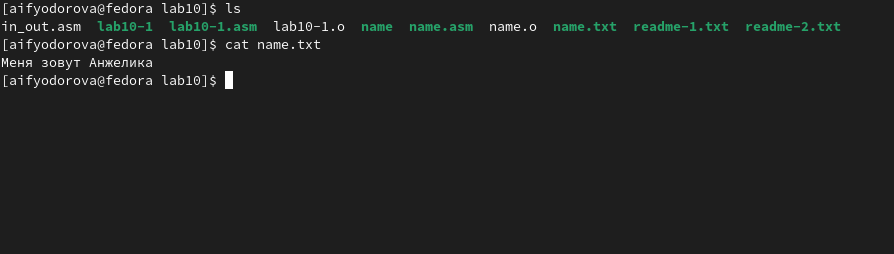
Редактирования файла name.asm

Создаю исполняемый файл и запускаю его. (рис.fig:013)



Исполнение файла name.asm

Теперь проверяю содержимое каталога с помощью ls (должен быть создан файл name.txt) и проверяю его содержимое с помощью команды сat(рис.fig:014)



Проверка содержимого в name.txt

Программа работает в соответствии с требованиями задания.

Код программы:

%include 'in\_out.asm'  
SECTION .data  
filename db 'name.txt', 0h; Имя файла  
msg db 'Как Вас зовут? ', 0h; Сообщение  
msg1 db 'Меня зовут ', 0h  
  
  
SECTION .bss  
contents resb 255  
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  
  
; создание файла  
mov ecx, 0777o; установка прав доступа  
mov ebx, filename; имя создаваемого файла  
mov eax, 8; номер системного вызова sys\_creat  
int 80h; вызов ядра  
  
; --- Открытие существующего файла  
mov ecx, 2  
mov ebx, filename   
mov eax, 5  
int 80h  
; открываем для записи (2)  
; --- Запись дескриптора файла в esi  
mov esi, eax  
  
mov eax, msg1; в eax запишется количество  
call slen ; введенных байтов  
  
; --- Записываем в файл contents (sys\_write)  
mov edx, eax  
mov ecx, msg1  
mov ebx, esi  
mov eax, 4  
int 80h  
  
; --- Расчет длины введенной строки  
mov eax, contents  
call slen  
  
; --- Записываем в файл contents  
  
mov edx, eax  
mov ecx, contents  
mov ebx, esi  
mov eax, 4  
int 80h  
; --- Закрываем файл  
mov ebx, esi  
mov eax, 6  
int 80h  
  
call quit

# 6 Выводы

Я приобрела навыки написания программ для работы с файлами.

# 7 Список литературы

[Лабораторная работа №10. Работа с файлами средствами Nasm](https://esystem.rudn.ru/pluginfile.php/2089097/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%9610.%20%D0%A0%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%D1%81%20%D1%84%D0%B0%D0%B9%D0%BB%D0%B0%D0%BC%D0%B8%20%D1%81%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B2%D0%B0%D0%BC%D0%B8%20Nasm.pdf)