РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

дисциплина: Архитектура компьютера

Студент: Нурмухаметова Каролина

Группа: НКАбд-05-25

МОСКВА

2025 г.

Оглавление

1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера mov и int.

2 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной. Программа на языке ассемблера NASM, как правило, состоит из трёх секций: секция кода программы (SECTION .text), секция инициированных (известных во время компиляции) данных (SECTION .data) и секция неинициализированных данных (тех, под которые во время компиляции только отводится память, а значение присваивается в ходе выполнения программы) (SECTION .bss). Для объявления инициированных данных в секции .data используются директивы DB, DW, DD, DQ и DT, которые резервируют память и указывают, какие значения должны храниться в этой памяти: - DB (define byte) — определяет переменную размером в 1 байт; - DW (define word) — определяет переменную размеров в 2 байта (слово); - DD (define double word) — определяет переменную размером в 4 байта (двойное слово); - DQ (define quad word) — определяет переменную размером в 8 байт (учетве- рённое слово); - DT (define ten bytes) — определяет переменную размером в 10 байт. Директивы используются для объявления простых переменных и для объявления массивов. Для определения строк принято использовать директиву DB в связи с особенностями хранения данных в оперативной памяти. Инструкция языка ассемблера mov предназначена для дублирования данных источника в приёмнике.

mov dst,src. Здесь операнд dst — приёмник, а src — источник. В качестве операнда могут выступать регистры (register), ячейки памяти (memory) и непосредственные значения (const). Инструкция языка ассемблера intпредназначена для вызова прерывания с указанным номером.

int n Здесь n — номер прерывания, принадлежащий диапазону 0–255. При программировании в Linux с использованием вызовов ядра sys_calls n=80h (принято задавать в шестнадцатеричной системе счисления).

3 Выполнение лабораторной работы

3.1 Lab5-1

Открыла Midnight Commander командой mc.

Пользуясь клавишами ↑ , ↓ и Enter перешла в каталог ~/work/arch-pc созданный при выполнении лабораторной работы №4 (Рис. 3.1.1).

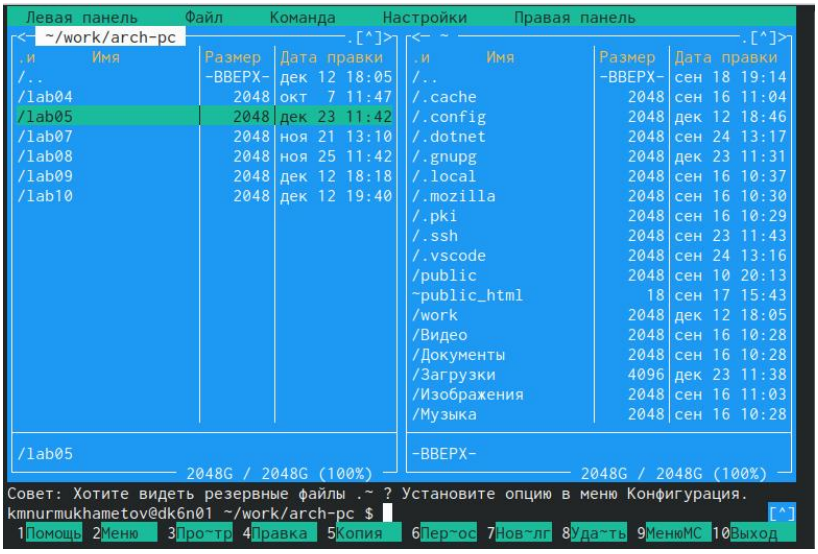

Рис. 3.1.1 — Переход в каталог

С помощью функциональной клавиши F7 создала папку lab05 и перешла в созданный каталог (Рис. 3.1.2).

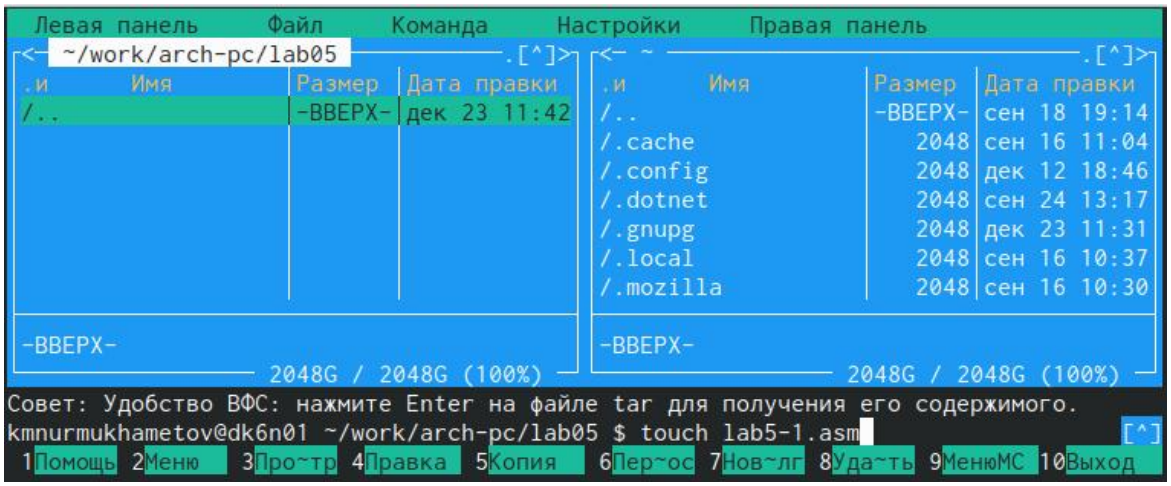

Рис. 3.1.2 — Создание папки lab05

Пользуясь строкой ввода и командой touch создала файл lab5-1.asm (Рис. 3.1.3).

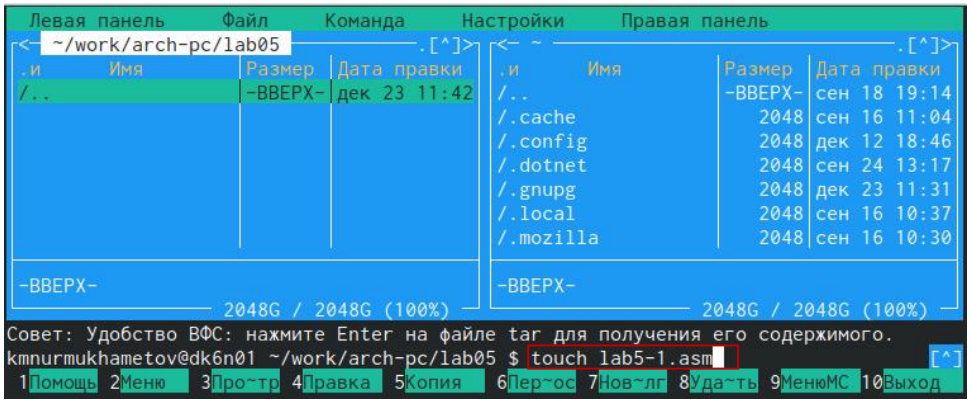

Рис. 3.1.3 — Файл lab5-1.asm

С помощью функциональной клавиши F4 открыла файл lab5-1.asm для редактирования во встроенном редакторе.

Введите текст программы из листинга 5.1, сохранила изменения и закрыла файл.

С помощью функциональной клавиши F3 открыла файл lab5-1.asm для просмотра.

Убедилась, что файл содержит текст программы (Рис. 3.1.4).

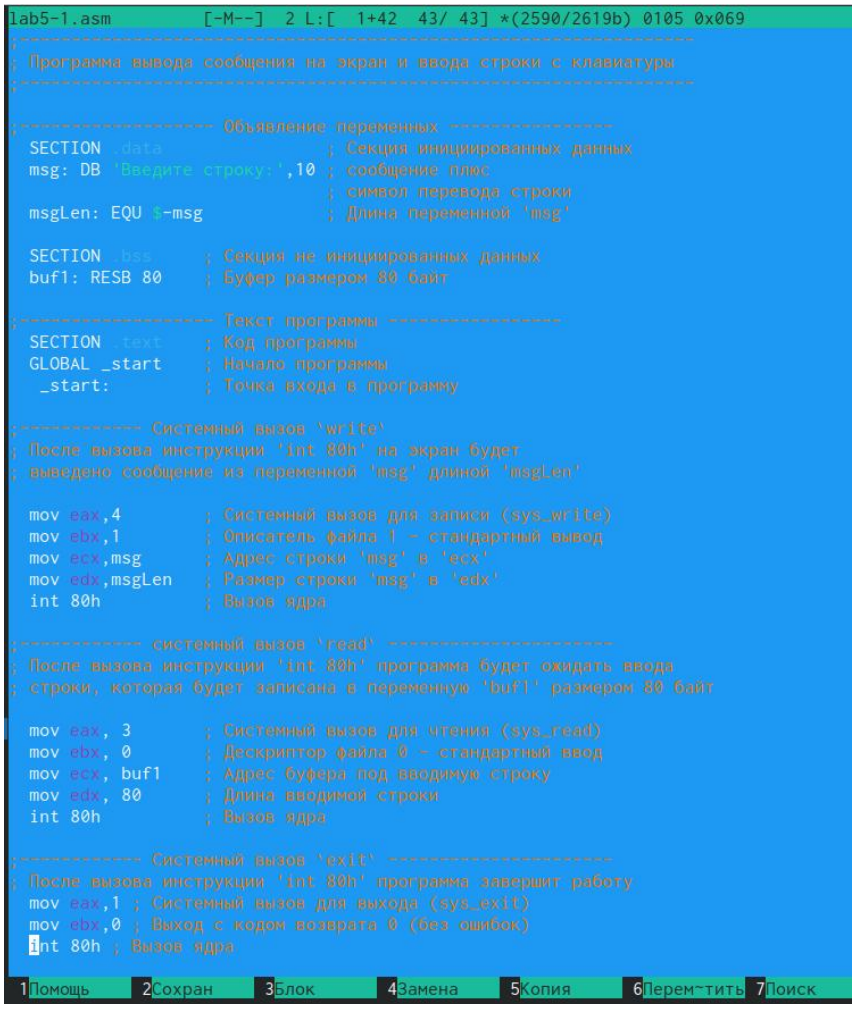

Рис. 3.1.4 — Текст программы lab5-1.asm

Оттранслировала текст программы lab5-1.asm в объектный файл. Выполнила компоновку объектного файла и запустила получившийся исполняемый файл. Программа вывела строку 'Введите строку:' и ожидала ввода с клавиатуры. На запрос ввела Nurmukhametova Karolina Marselevna (Рис. 3.1.5).

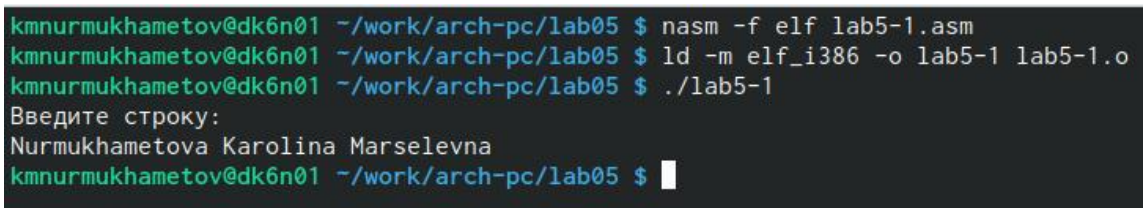

Рис 3.1.5 — Вывод программы lab5-1.asm

3.2 Подключение внешнего файла in_out.asm

Скачала файл in_out.asm со страницы курса в ТУИС в тот же каталог, в котором хранится файл с используемой программой.

Скопировала файл in_out.asm в каталог с файлом lab5-1.asm с помощью функциональной клавиши F5 (Рис. 3.1.6).

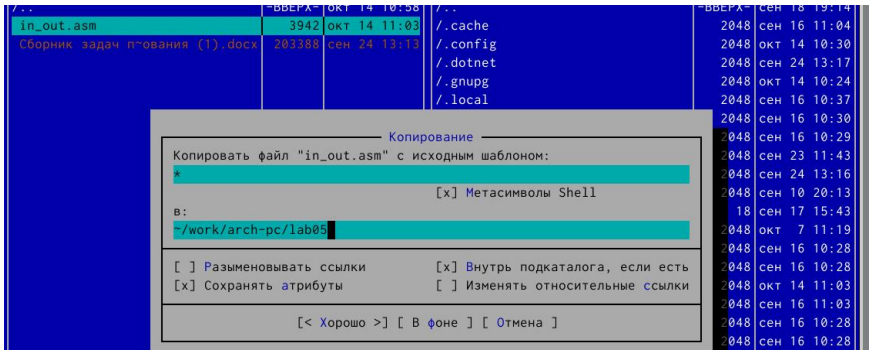

Рис. 3.1.6 — Файл in_out.asm

С помощью функциональной клавиши F6 создала копию файла lab5-1.asm с именем lab5-2.asm. Выделила файл lab5-1.asm, нажала клавишу F6 , ввела имя файла lab5-2.asm и нажала клавишу Enter.

Исправила текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in_out.asm в соответствии с листингом 5.2. Создала исполняемый файл и проверила его работу (Рис. 3.1.7).

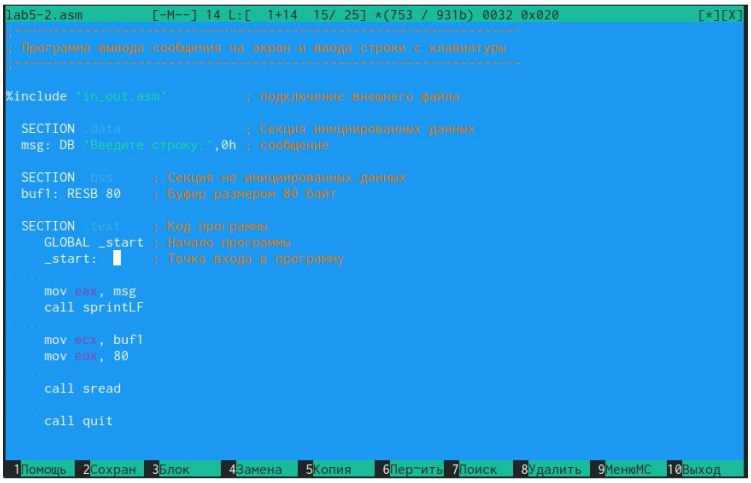

Рис. 3.1.7 — Текст программы lab5-2.asm

Создала файл lab5-2-1.asm заменила подпрограмму sprintLF на sprint. Создала исполняемый файл и проверила его работу.

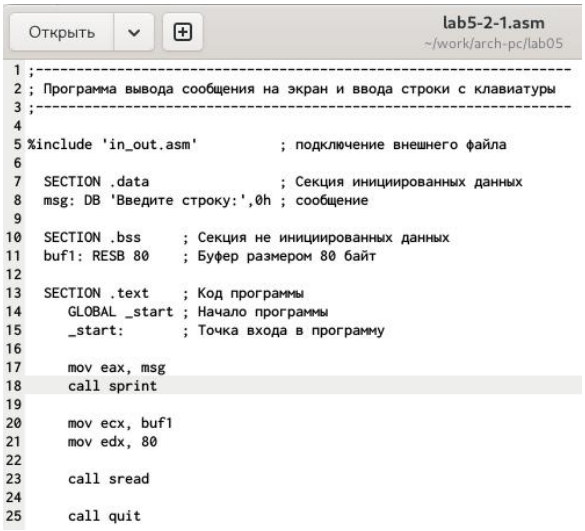

Рис. 3.1.8 — Текст программы lab5-2-1.asm

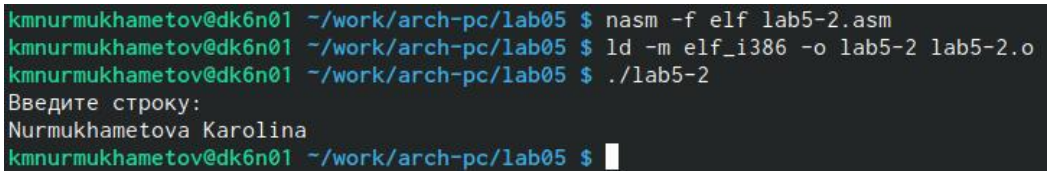

Рис. 3.1.9 — Вывод программы lab5-2.asm

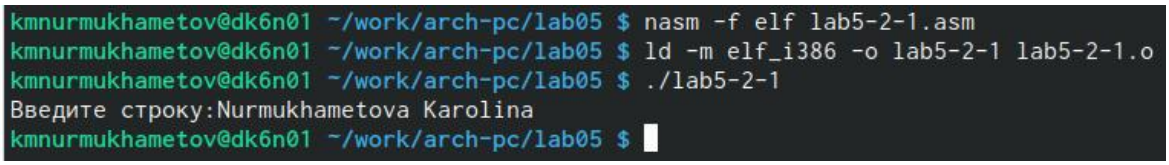

Рис. 3.1.9 — Вывод программы lab5-2-1.asm

Разница в том, что в первом варианте строка вводится с новой строки, а во втором — на этой же строке.

4 Задание для самостоятельной работы

Создала копию файла lab5-1.asm с именем lab5-1-sam.asm. Внесла изменения в программу (без использования внешнего файла in_out.asm), так чтобы она работала по следующему алгоритму: выводила приглашение типа “Введите строку:”; вводила строку с клавиатуры; выводила введённую строку на экран. (Рис. 4.1)

```
;-----------------------------------------------------------------
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----------------------------------------------------------------

;------------------- Объявление переменных ----------------
  SECTION .data               ; Секция инициированных данных
  msg: DB 'Введите строку:',10 ; сообщение плюс
                              ; символ перевода строки
  msgLen: EQU $-msg ; Длина переменной 'msg'

  SECTION .bss ; Секция не инициированных данных
  buf1: RESB 80 ; Буфер размером 80 байт

;------------------- Текст программы -----------------
  SECTION .text ; Код программы
  GLOBAL _start ; Начало программы
   _start: ; Точка входа в программу

;------------ Системный вызов `write`
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'

   mov eax,4 ; Системный вызов для записи (sys_write)
   mov ebx,1 ; Описатель файла 1 - стандартный вывод
   mov ecx,msg ; Адрес строки 'msg' в 'ecx'
   mov edx,msgLen ; Размер строки 'msg' в 'edx'
   int 80h ; Вызов ядра

;------------ системный вызов `read` ----------------------
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт

   mov eax, 3 ; Системный вызов для чтения (sys_read)
   mov ebx, 0 ; Дескриптор файла 0 - стандартный ввод
   mov ecx, buf1 ; Адрес буфера под вводимую строку
   mov edx, 80 ; Длина вводимой строки
   int 80h ; Вызов ядра
;------------ Повторный вызов `write`
   mov eax,4 ; Системный вызов для записи (sys_write)
   mov ebx,1 ; Описатель файла 1 - стандартный вывод
   mov ecx,buf1 ; Адрес строки 'buf1' в 'ecx'
   mov edx,msgLen ; Размер строки 'msg' в 'edx'
   int 80h ; Вызов ядра
;------------ Системный вызов `exit` ----------------------
; После вызова инструкции 'int 80h' программа завершит работу

   mov eax,1 ; Системный вызов для выхода (sys_exit)
   mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
   int 80h ; Вызов ядра
```

Рис. 4.1 — Текст программы lab5-1-sam.asm

Получила исполняемый файл и проверила его работу. На приглашение ввести строку ввела свою фамилию и имя (Рис. 4.2).

```
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1-sam.asm
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1-sam lab5-1-sam.o
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $ ./lab5-1-sam
Введите строку:
Nurmukhametova Karolina
Nurmukhametova Karolina
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $
```

Рис. 4.2 — Вывод программы lab5-1-sam.asm

Создала копию файла lab5-2.asm с именем lab5-2-sam.asm. Исправила текст

программы с использованием подпрограмм из внешнего файла in_out.asm, так чтобы она работала по следующему алгоритму: выводила приглашение типа “Введите строку:”; вводила строку с клавиатуры; выводила введённую строку на экран (Рис. 4.3).

```
;-------------------------------------------------------------
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-------------------------------------------------------------
%include 'in_out.asm'          ; подключение внешнего файла

  SECTION .data                ; Секция инициированных данных
  msg: DB 'Введите строку:',0h ; сообщение

  SECTION .bss     ; Секция не инициированных данных
  buf1: RESB 80    ; Буфер размером 80 байт
  SECTION .text    ; Код программы
     GLOBAL _start ; Начало программы
     _start:       ; Точка входа в программу
     mov eax, msg
     call sprintLF
     mov ecx, buf1
     mov edx, 80
     call sread
;---------------- Изменения -------------------
     mov eax, buf1
     mov ecx, eax
     call sprint

     call quit
```

Рис 4.3 — Текст программы lab5-2-sam.asm

Создала исполняемый файл и проверила его работу. (Рис. 4.4)

```
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2-sam.asm
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2-sam lab5-2-sam.o
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $ ./lab5-2-sam
Введите строку:
Nurmukhametova Karolina
Nurmukhametova Karolina
kmnurmukhametov@dk6n01 ~/work/arch-pc/lab05 $
```

Рис 4.4 — Текст программы lab5-2-sam.asm

5 Выводы

При выполнении данной лабораторной работы я приобрела практические навыки работы в Midnight Commander, а также освоила инструкции языка ассемблера mov и int.

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/.
4. NASM Assembly Language Tutorials. — 2021. — URL: https://asmtutor.com/.
5. *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/Learning-bash-Shell-Programming-Nutshell/dp/0596009658.
6. *Robbins A.* Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: https://www.nasm.us/docs.php.
8. *Zarrelli G.* Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. *Колдаев В. Д., Лупин С. А.* Архитектура ЭВМ. — М. : Форум, 2018.
10. *Куляс О. Л., Никитин К. А.* Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
11. *Новожилов О. П.* Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: https://www.opennet.ru/docs/RUS/nasm/.
13. *Робачевский А., Немнюгин С., Стесик О.* Операционная система UNIX. — 2-е изд. — БХВ-Петербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
14. *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.
15. *Таненбаум Э.* Архитектура компьютера. — 6-е изд. — 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).