Отчёт по лабораторной работе №8

Дисциплина: архитектура компьютера

Адмиральская Александра Андреевна

Содержание

# 1 Цель работы

Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

# 2 Задание

1.Реализация циклов в NASM 2.Обработка аргументов командной строки 3.Вычисление суммы аргументов командной строки 4.Выполнение задания для самостоятельной работы

# 3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. 1 приведено краткое описание стандартных каталогов Unix.

Таблица 1: Описание некоторых каталогов файловой системы GNU Linux

| Имя каталога | Описание каталога |
| --- | --- |
| / | Корневая директория, содержащая всю файловую |
| /bin | Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям |
| /etc | Общесистемные конфигурационные файлы и файлы конфигурации установленных программ |
| /home | Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя |
| /media | Точки монтирования для сменных носителей |
| /root | Домашняя директория пользователя root |
| /tmp | Временные файлы |
| /usr | Вторичная иерархия для данных пользователя |

Более подробно про Unix см. в [1–4].

# 4 Выполнение лабораторной работы

Для начала создаем каталог для программам лабораторной работы № 8, переходим в него и создаем файл lab8-1.asm (рис. 1).



Рис. 1: Создание каталога и файла

Открываем созданной файл командой mc и вводим в него текст программы из листинга 8.1 (рис. 2).

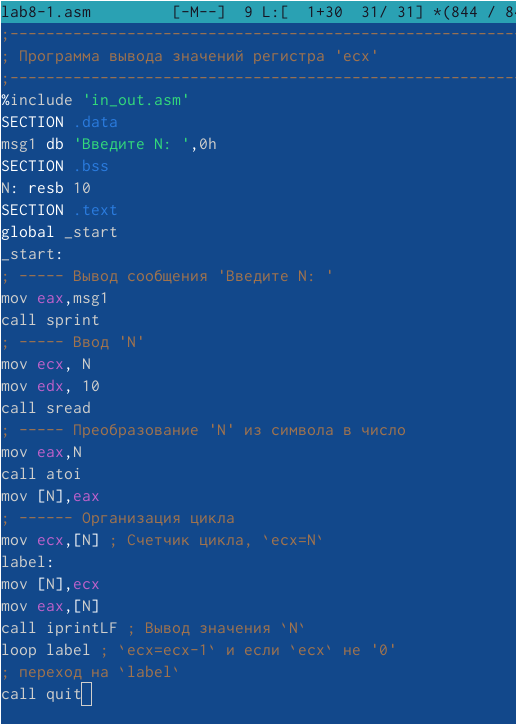


Рис. 2: Текст программы в файле lab8-1.asm

Создаем исполняемый файл и проверяем его работу (рис. 3).

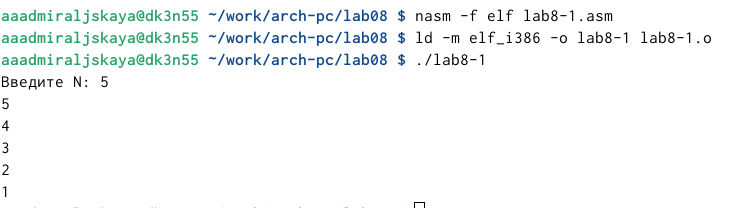


Рис. 3: Цикл, выводящий цифры от 5 до 1

Затем в этом же файле изменяем текст программы добавив изменение значение регистра ecx в цикле (рис. 4).

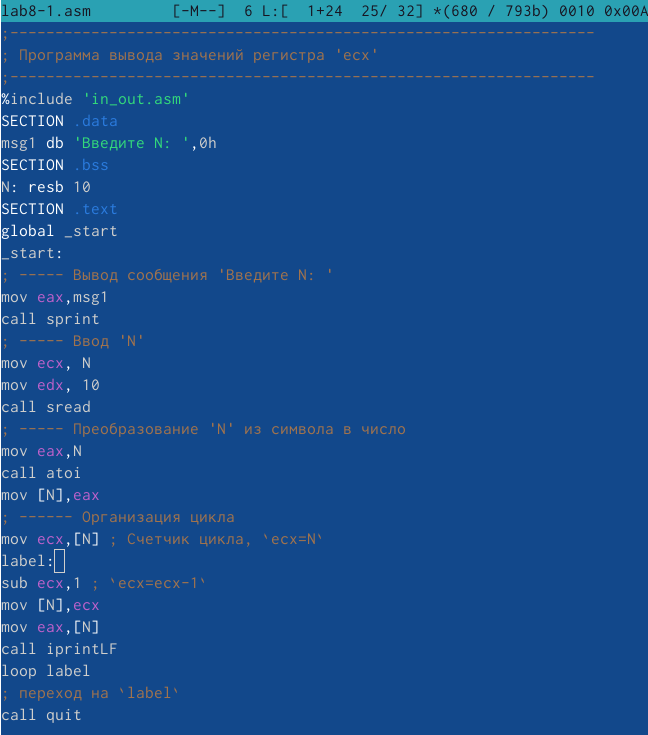


Рис. 4: Измененный текст программы

Создаем исполняемый файл и проверяем его работу (рис. 5).

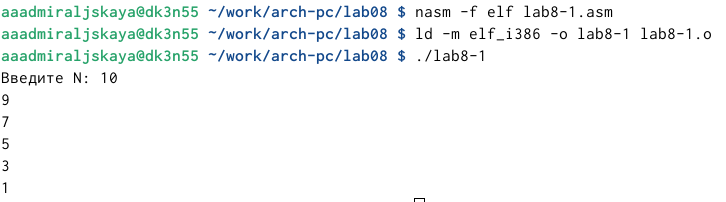


Рис. 5: Цикл, выводящий нечетные цифры от 10 до 1

Далее в этом же файле вносим изменения в текст программы добавив команды push и pop для сохранения значения счетчика цикла loop (рис. 6).

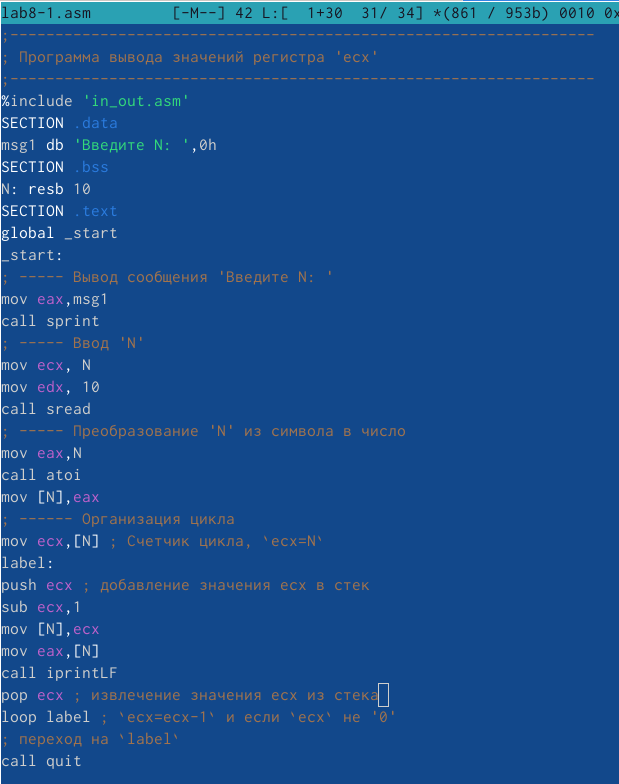


Рис. 6: Внесенные в файл изменения

Создаем исполняемый файл и проверяем его работу (рис. 7).

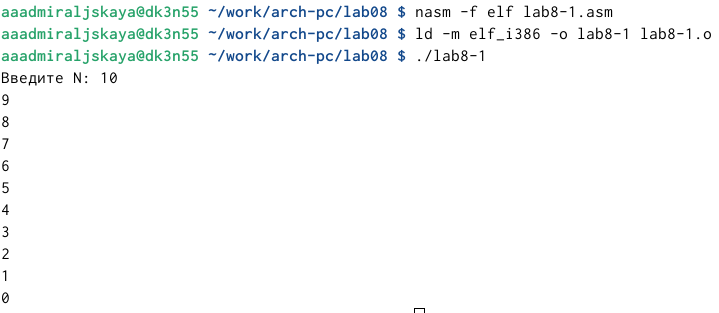


Рис. 7: Цикл, выводящий цифры от 9 до 0

Следующим шагом создаем файл lab8-2.asm и вводим в него текст программы из листинга 8.2 (рис. 8).

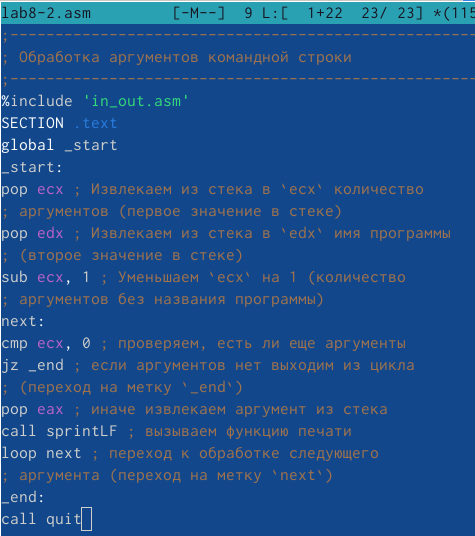


Рис. 8: Текст программы в файле lab8-2.asm

Создаем исполняемый файл и проверяем его работу, указав аргументы (рис. 9).

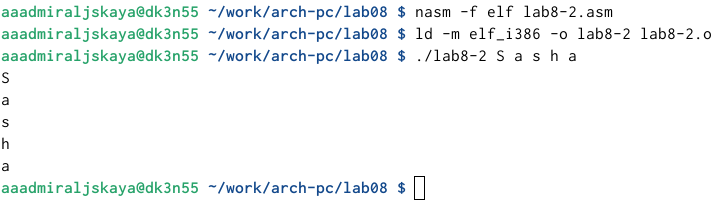


Рис. 9: Работа файла с указанными аргументами

Теперь создаем файл lab8-3.asm и вводим в него текст программы из листинга 8.3 (рис. 10).

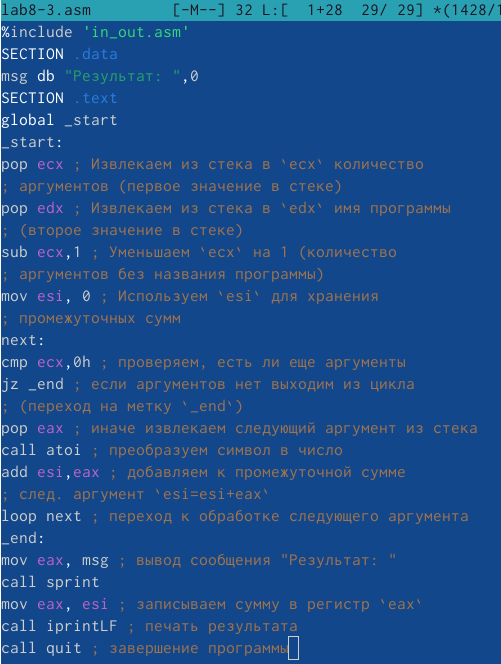


Рис. 10: Текст программы в файле lab8-3.asm

Создаем исполняемый файл и запускаем его, указав аргументы (рис. 11).

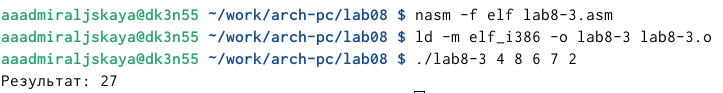


Рис. 11: Работа программы, которая складыает числа, введенные пользователем

В этом же файле lab8-3.asm изменяем программу так, чтобы она умножала введенные числа (рис. 12).

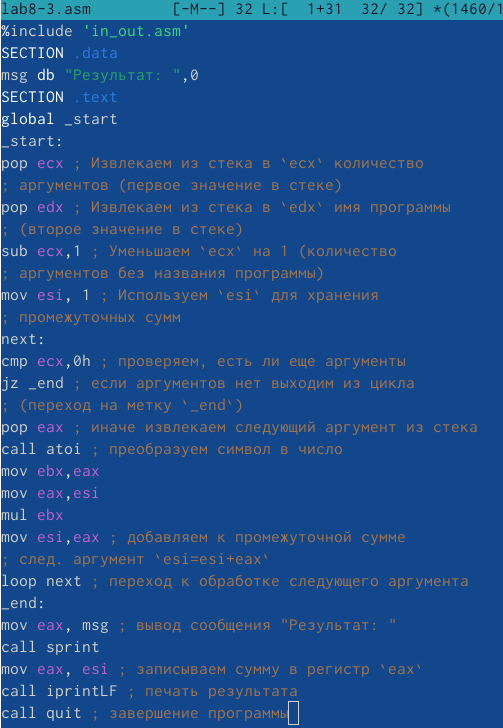


Рис. 12: Измененный текст файла lab8-3.asm

Создаем исполняемый файл и запускаем его, указав аргументы (рис. 13).

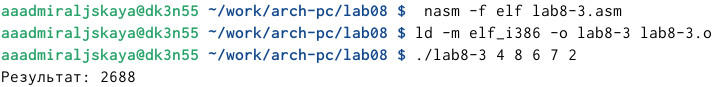


Рис. 13: Работа программы, которая умножает числа, введенные пользователем

Приступим к выполнению задания для самостоятельной работы. Создаем файл lab8-4.asm и вводим в него программу, которая находит сумму значений функции. Вид функции берем из варианта №7 (рис. 14).

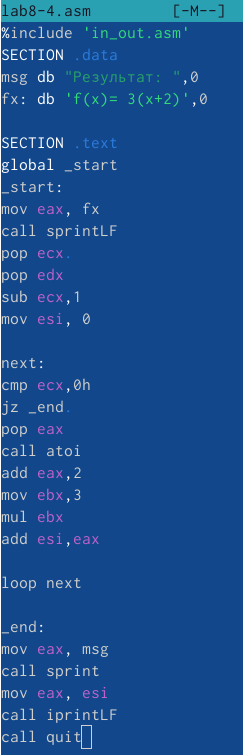


Рис. 14: Программа, находящая сумму значений функции

Создаем исполняемый файл и проверяем его работу для значений: x=1, x=2, x=3, x=4 - указываем цифры как аргументы (рис. 15).

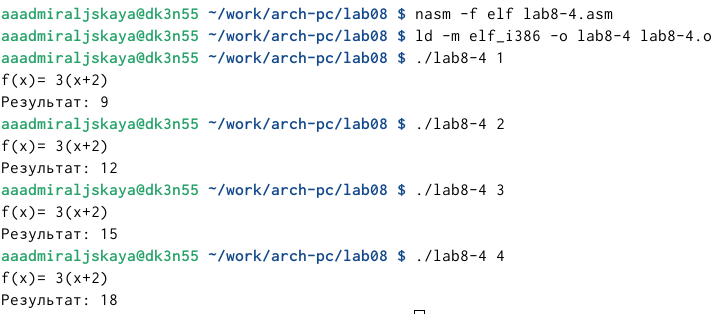


Рис. 15: Работы программы при разных значениях x

Листинг программы из задания самостоятельной работы:

%include ‘in\_out.asm’ SECTION .data msg db “Результат:”,0 fx: db ‘f(x)= 3(x+2)’,0

SECTION .text global \_start \_start: mov eax, fx call sprintLF pop ecx pop edx sub ecx,1 mov esi, 0

next: cmp ecx,0h jz \_end pop eax call atoi add eax,2 mov ebx,3 mul ebx add esi,eax

loop next

\_end: mov eax, msg call sprint mov eax, esi call iprintLF call quit

# 5 Выводы

В процессе выполнения лабораторной работы я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.

# Список литературы

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.

2. Robbins A. Bash Pocket Reference. O’Reilly Media, 2016. 156 с.

3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.

4. Newham C. [Learning the bash Shell: Unix Shell Programming](http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658). O’Reilly Media, 2005. 354 с.