Отчет по лабораторной работе №7

Дисциплина: архитектура компьютера

Адмиральская Александра Андреевна

Содержание

# 1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

# 2 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].

# 3 Выполнение лабораторной работы

Для начала создаем каталог для программам лабораторной работы № 7, переходим в него и создаем файл lab7-1.asm (рис. 1).



Рис. 1: Создание каталога и файла

Открываем созданный файл и вводим в него текст программы из листинга 7.1. Программа выводит 2, 3 (рис. 2).

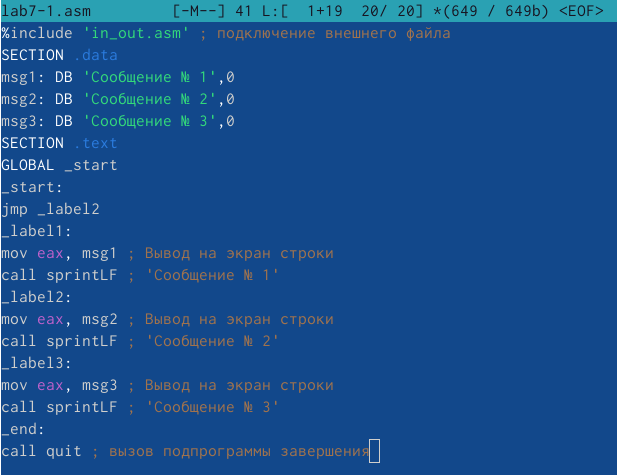


Рис. 2: Ввод текста программы

Создаем исполняемый файл и запускаем его (рис. 3).

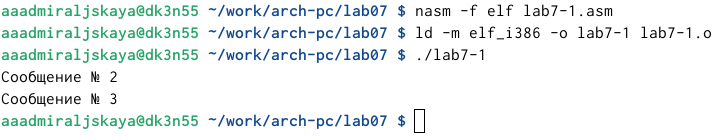


Рис. 3: Создание и запуск исполняемого файла

Заново открываем файл lab7-1.asm и изменяем текст программы, вывод стал 2, 1 (рис. 4).

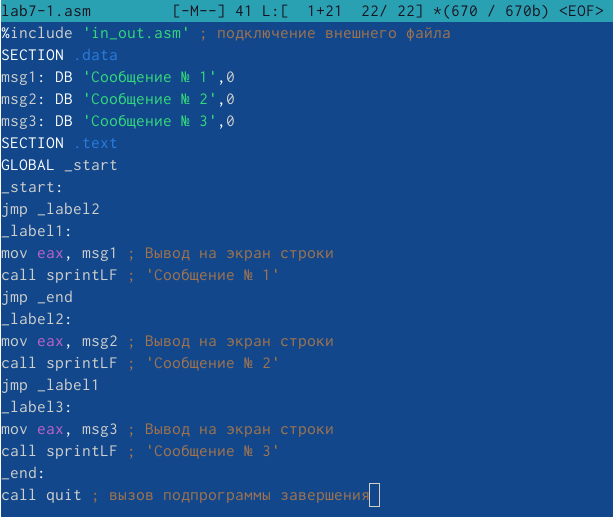


Рис. 4: Изменение текста программы

Создаем исполняемый файл и запускаем его (рис. 5).

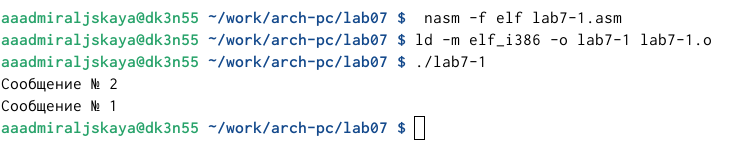


Рис. 5: Создание и запуск исполняемого файла

Затем изменяем текст программы в этом же файле, чтобы вывод программы был 3, 2, 1 (рис. 6).

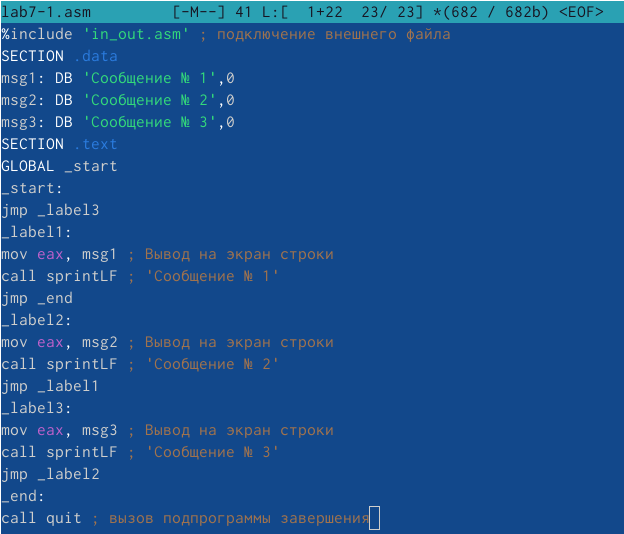


Рис. 6: Изменение текста программы

Создаем и запускаем исполняемый файл (рис. 7).

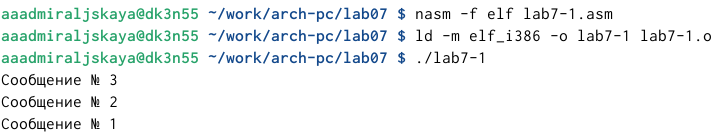


Рис. 7: Запуск исполняемого файла

Далее создаем файл lab7-2.asm и открываем его (рис. 8).

Рис. 8: Создание файла и переход в него

Рис. 8: Создание файла и переход в него

Вводим в него текст программы из листинга 7.3 (рис. 9).

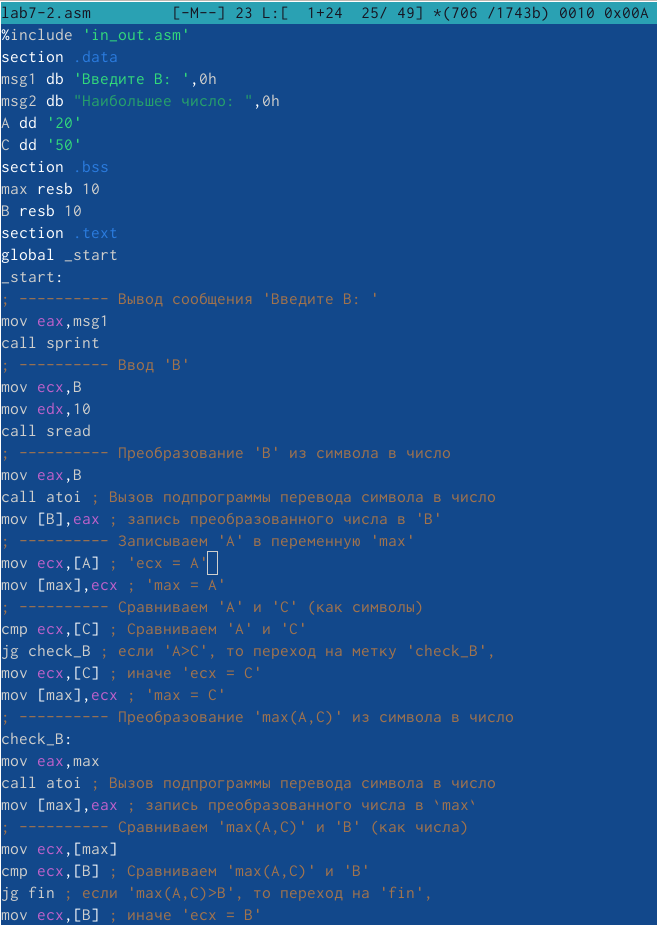


Рис. 9: Ввод текста программы

Создаем исполняемый файл и проверяем его работу для разных значений B (рис. 10).

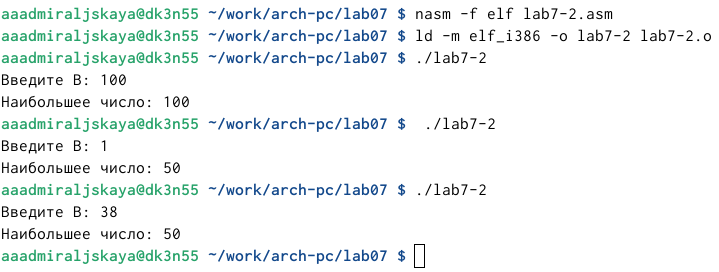


Рис. 10: Проверка работы исполняемого файла

Затем создаем файл листинга для программы из файла lab7-2.asm (рис. 11).

Рис. 11: Создание файла листинга

Рис. 11: Создание файла листинга

Открываем этот файл и изучаем его (рис. 12).

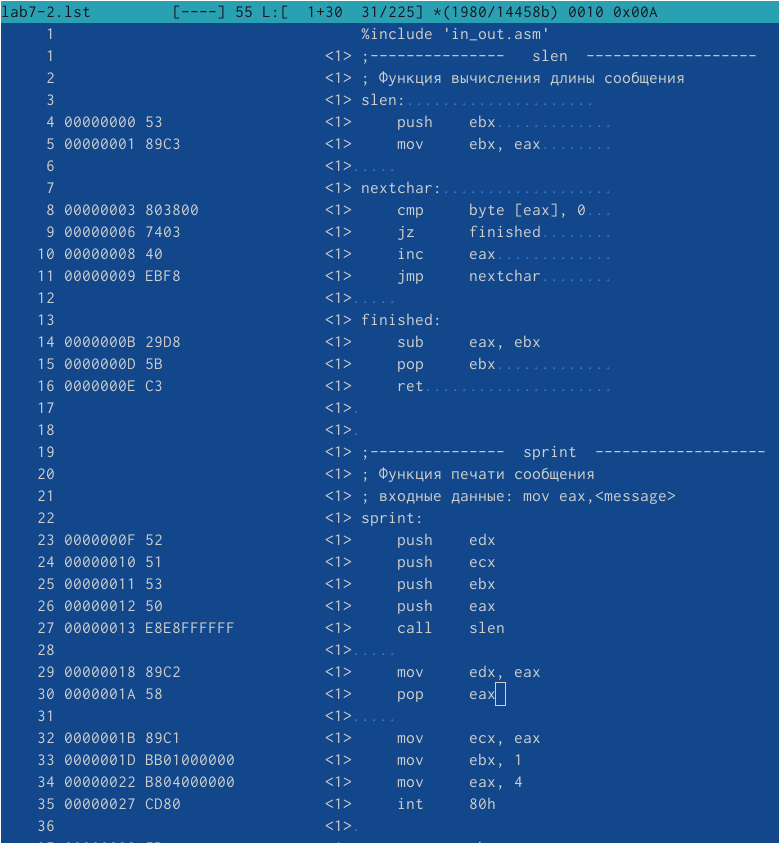


Рис. 12: Файл листинга

Затем открываем файл с программой lab7-2.asm и удаляем один операнд (рис. 13).

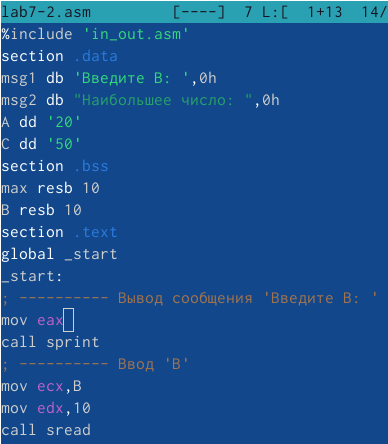


Рис. 13: Удаление одного операнда

Выполняем трансляцию с получением файла листинга (рис. 14).

Рис. 14: Выполнение трансляции

Рис. 14: Выполнение трансляции

Открываем файл lab7-2.lst и проверяем на наличие ошибки (рис. 15).

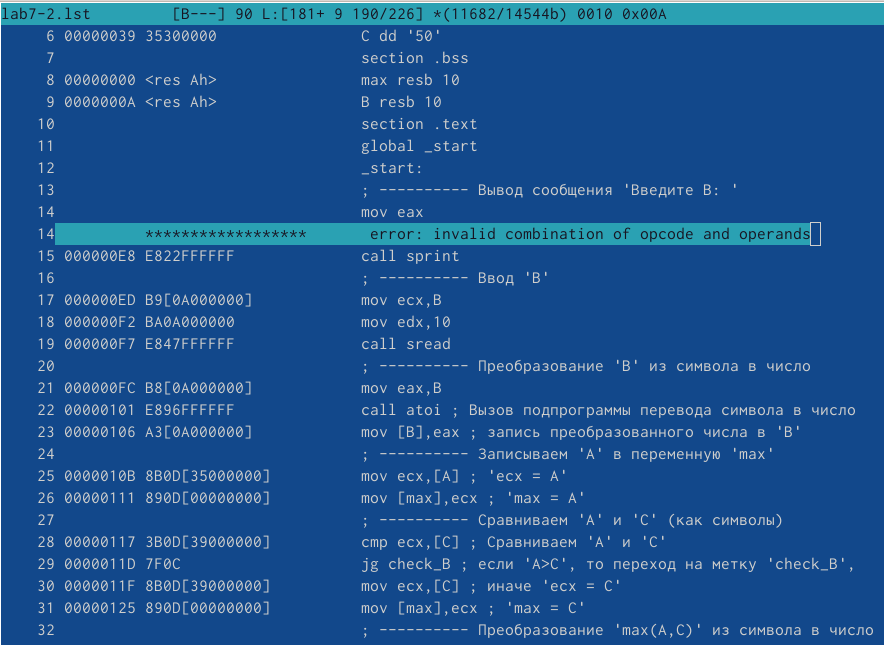


Рис. 15: Ошибка в листинге файла

Перейдем к выполнению заданий для самостоятельной работы. Создаем файл lab7-3.asm (рис. 16).

Рис. 16: Создание файла

Рис. 16: Создание файла

В файл вводим программу для нахождения наименьшей из 3 целочисленных переменных. Значение переменных берем из 7 варианта (рис. 17).

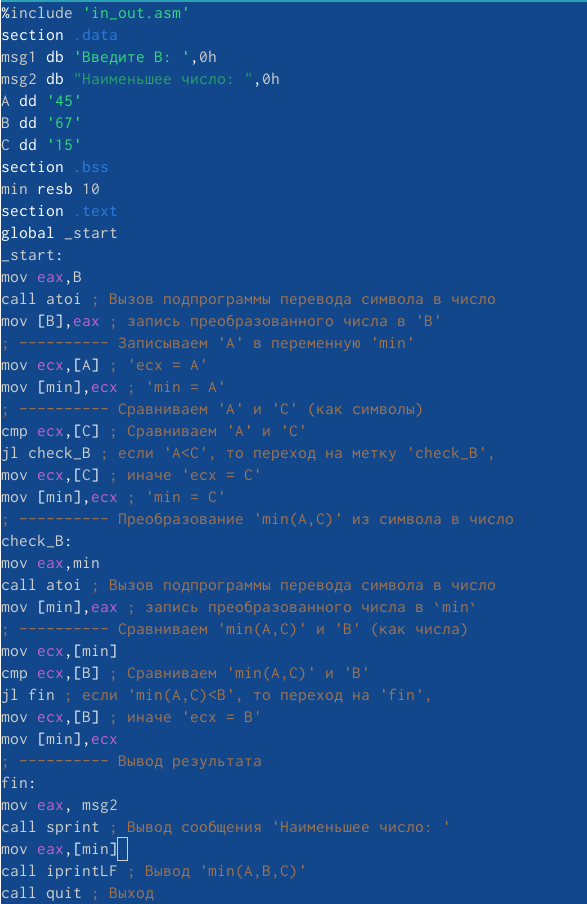


Рис. 17: Ввод текста программы

Создаем исполняемый файл и проверяем его работу. Все верно: из переменных 45, 67 и 15 - 15 является наименьшей, и на выводе мы получаем именно ее (рис. 18).

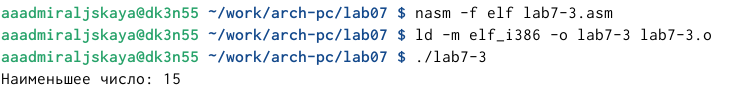


Рис. 18: Проверка работы исполняемого файла

Теперь создаем файл lab7-4.asm (рис. 19).

Рис. 19: Создание файла

Рис. 19: Создание файла

В файле пишем программу, которая для введенных с клавиатуры значений 𝑥 и 𝑎 вычисляет значение заданной функции 𝑓(𝑥) и выводит результат вычислений. Функцию берем из 7 варианта (рис. 20).

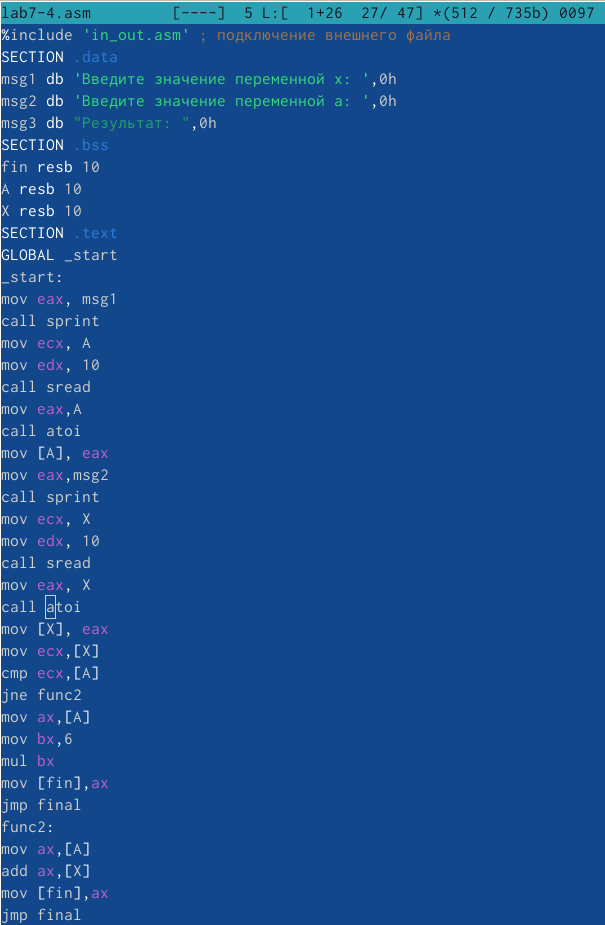


Рис. 20: Написание программы

Создаем исполняемый файл и проверяем его работу для значений a,x из варианта №7: x=1, a=1 (рис. 21).

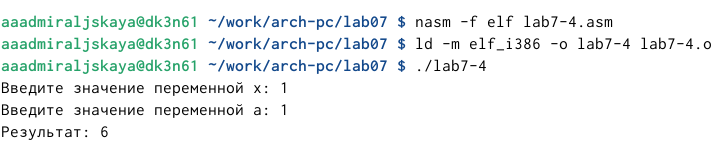


Рис. 21: Создание и проверка работы исполняемого файла

Теперь проверяем его работу для значений x=2, a=1. Все верно (рис. 22).

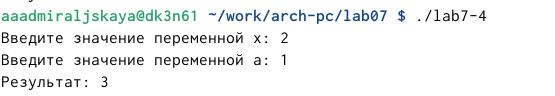


Рис. 22: Проверка работы исполняемого файла

Текст программы первого задания самостоятельной работы:

%include ‘in\_out.asm’ section .data msg1 db ‘Введите B:’,0h msg2 db “Наименьшее число:”,0h A dd ‘45’ B dd ‘67’ C dd ‘15’ section .bss min resb 10 section .text global \_start \_start: mov eax,B call atoi ; Вызов подпрограммы перевода символа в число mov [B],eax ; запись преобразованного числа в ‘B’ ; ———- Записываем ‘A’ в переменную ‘min’ mov ecx,[A] ; ‘ecx = A’ mov [min],ecx ; ‘min = A’ ; ———- Сравниваем ‘A’ и ‘С’ (как символы) cmp ecx,[C] ; Сравниваем ‘A’ и ‘С’ jl check\_B ; если ‘A<C’, то переход на метку ‘check\_B’, mov ecx,[C] ; иначе ‘ecx = C’ mov [min],ecx ; ‘min = C’ ; ———- Преобразование ‘min(A,C)’ из символа в число check\_B: mov eax,min call atoi ; Вызов подпрограммы перевода символа в число mov [min],eax ; запись преобразованного числа в min ; ———- Сравниваем ‘min(A,C)’ и ‘B’ (как числа) mov ecx,[min] cmp ecx,[B] ; Сравниваем ‘min(A,C)’ и ‘B’ jl fin ; если ‘min(A,C)<B’, то переход на ‘fin’, mov ecx,[B] ; иначе ‘ecx = B’ mov [min],ecx ; ———- Вывод результата fin: mov eax, msg2 call sprint ; Вывод сообщения ‘Наименьшее число:’ mov eax,[min] call iprintLF ; Вывод ‘min(A,B,C)’ call quit ; Выход

Текст программы второго задания самостоятельной работы:

%include ‘in\_out.asm’ ; подключение внешнего файла SECTION .data msg1 db ‘Введите значение переменной x:’,0h msg2 db ‘Введите значение переменной a:’,0h msg3 db “Результат:”,0h SECTION .bss fin resb 10 A resb 10 X resb 10 SECTION .text GLOBAL \_start \_start: mov eax, msg1 call sprint mov ecx, A mov edx, 10 call sread mov eax,A call atoi mov [A], eax mov eax,msg2 call sprint mov ecx, X mov edx, 10 call sread mov eax, X call atoi mov [X], eax mov ecx,[X] cmp ecx,[A] jne func2 mov ax,[A] mov bx,6 mul bx mov [fin],ax jmp final func2: mov ax,[A] add ax,[X] mov [fin],ax jmp final final: mov eax,msg3 call sprint mov eax,[fin] call iprintLF call quit

# 4 Выводы

В процессе выполнения лабораторной работы я изучила команды условного и безусловного переходов, приобрела навыки написания программ с использованием переходов и познакомилась с назначением и структурой файла листинга.

# 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.