Отчёт по лабораторной работе №8

Уткина Алина Дмитриевна

Содержание

# 1 Цель работы

Цель данной работы является изучение команд условного и безусловного переходов, приобретение навыков написания программ с использованием переходов, знакомство с назначением и структурой файла листинга.

# 2 Выполнение лабораторной работы

## 2.1 Реализация переходов в NASM

Создадим каталог для программам лабораторной работы № 8, перейдем в него и создадим файл lab8-1.asm. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции jmp. Введем в файл lab8-1.asm текст программы из листинга 8.1 (рис. 1).

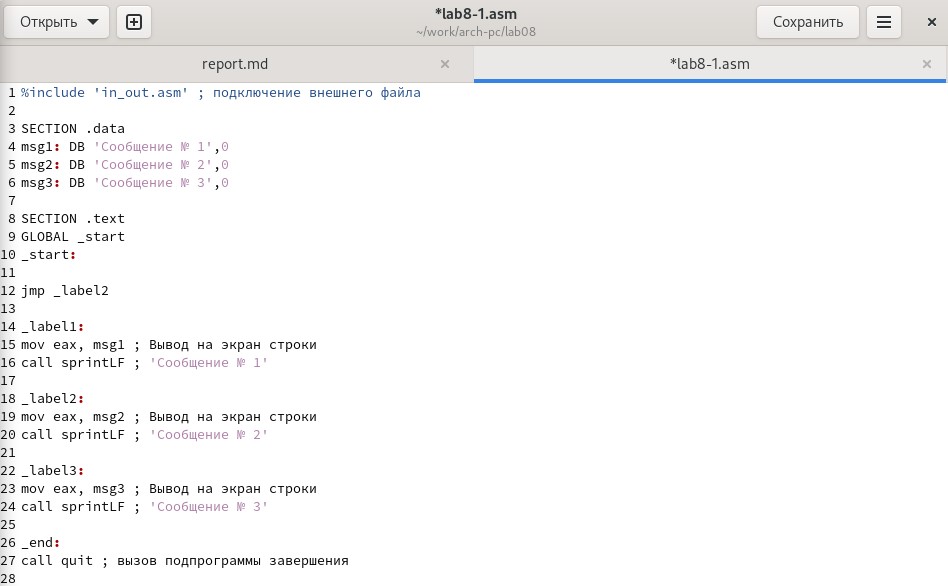


Рис. 1: Листинг 8.1. Программа с использованием инструкции jmp

Создадим исполняемый файл и запустим его. В результате работы данной программы будут выведены строки “Сооющение №2” и “Сообщение №3” (рис. 2).

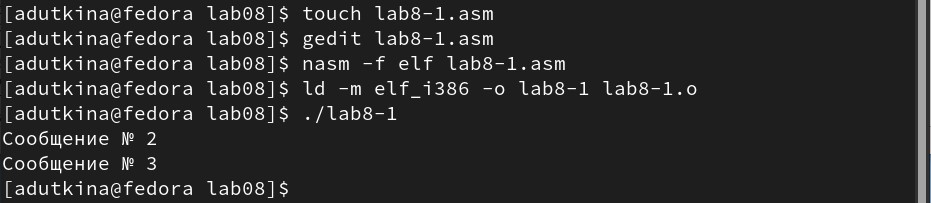


Рис. 2: Результат работы программы lab8-1.asm

Таким образом, использование инструкции jmp \_label2 меняет порядок исполнения инструкций и позволяет выполнить инструкции начиная с метки \_label2, пропустив вывод первого сообщения.

Инструкция jmp позволяет осуществлять переходы не только вперед но и назад. Изменим программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию jmp с меткой \_label1 (т.е. переход к инструкциям вывода сообщения № 1) и после вывода сообщения № 1 добавим инструкцию jmp с меткой \_end (т.е. переход к инструкции call quit). Изменим текст программы в соответствии с листингом 8.2 (рис. 3). При запуске программы выводится именно то, что нам нужно (рис. 4).

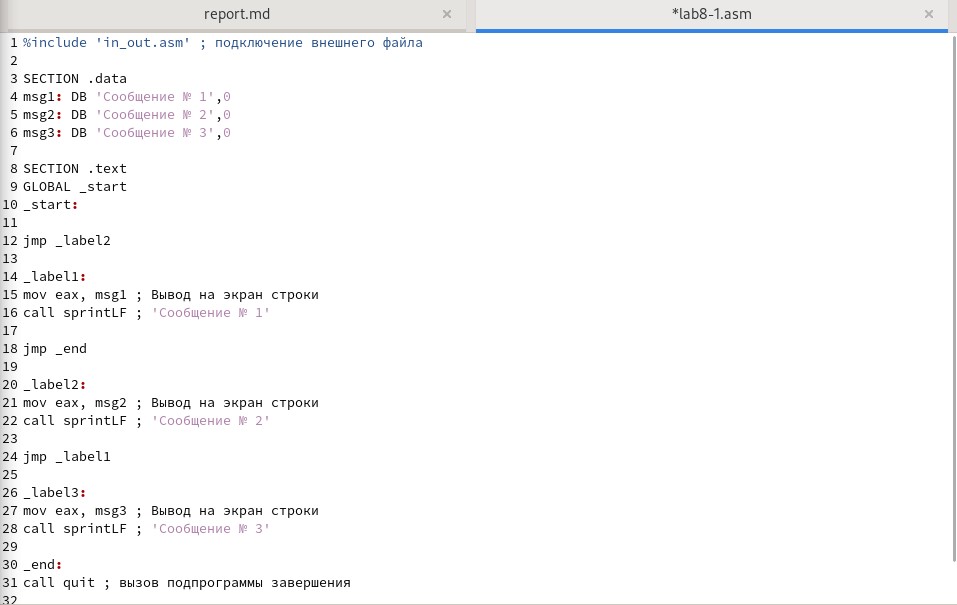


Рис. 3: Листинг 8.2. Измененный текст программы lab8-1.asm



Рис. 4: Результат измененной программы

Изменим текст программы добавив или изменив инструкции jmp, чтобы программа выводила сообщения в обратном порядке (рис. 5). Запустим исполняемый файл, чтобы проверить его работу (рис. 6).

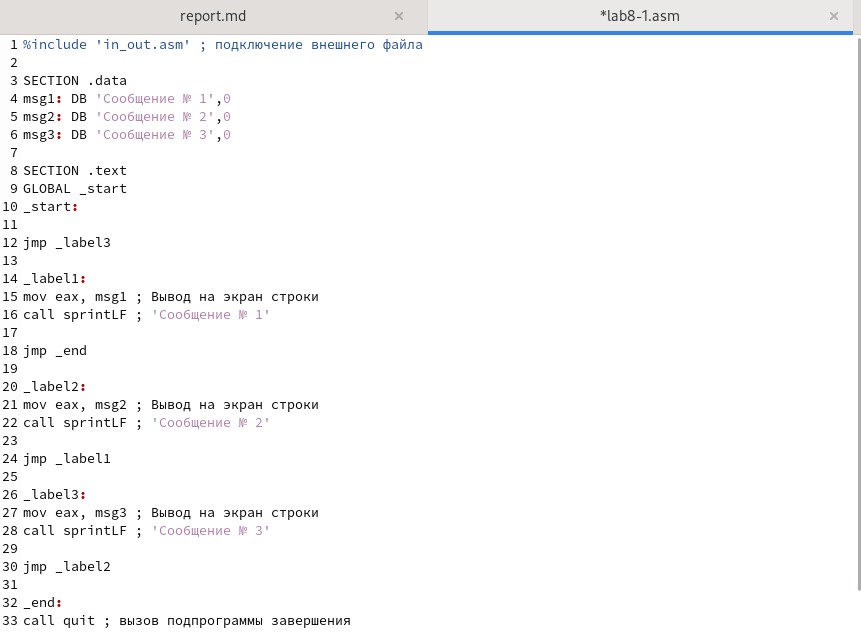


Рис. 5: Другой вариант программы lab8-1.asm

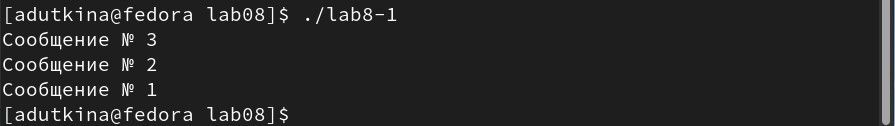


Рис. 6: Результат работы третьей программы

Использование инструкции jmp приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, т.е. переход должен происходить если выполнено какое-либо условие. В качестве примера рассмотрим программу, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: A,B и C. Значения для A и C задаются в программе, значение B вводиться с клавиатуры. Создадим файл lab8-2.asm в каталоге ~/work/arch-pc/lab08. Внимательно изучим текст программы из листинга 8.3 и введем его в lab8-2.asm (рис. 7). Создадим исполнительный файл и запустим его (рис. 8)

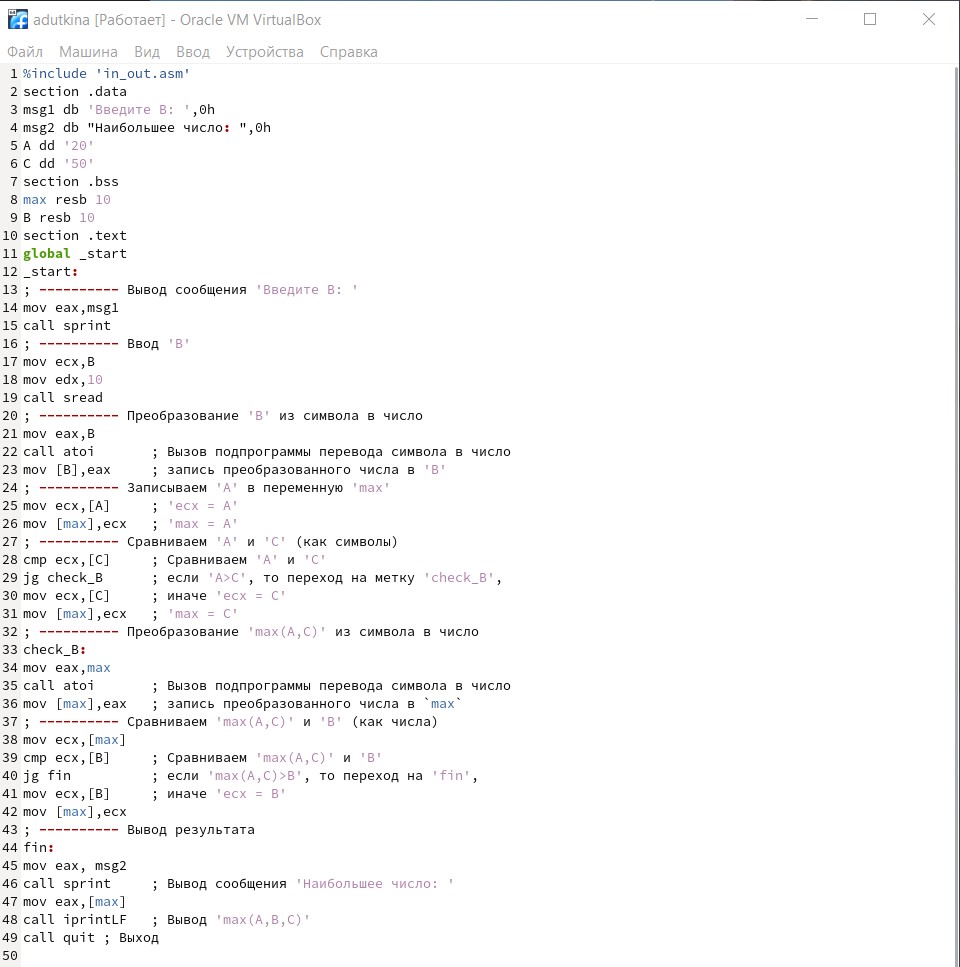


Рис. 7: Листинг 8.3. Программа определения максимального из 3 чисел

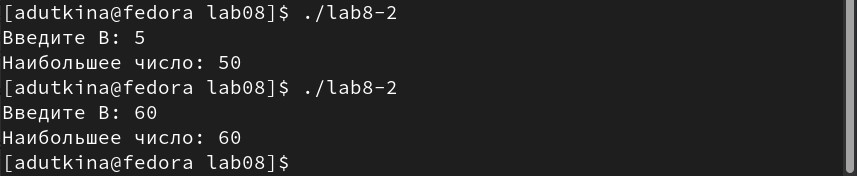


Рис. 8: Результат работы программы lab8-2.asm

Следует заметить, что в данном примере переменные A и С сравниваются как символы, а переменная B и максимум из A и С как числа (для этого используется функция atoi преобразования символа в число). Это сделано для демонстрации того, как сравниваются данные. Данную программу можно упростить и сравнивать все 3 переменные как символы (т.е. не использовать функцию atoi). Однако если переменные преобразовать из символов в числа, над ними можно корректно проводить арифметические операции.

## 2.2 Изучение структуры файлов листинга

Обычно nasm создаёт в результате ассемблирования только объектный файл. Получить файл листинга можно, указав ключ -l и задав имя файла листинга в командной строке. Создадим файл листинга для программы из файла lab8-2.asm (рис. 9).

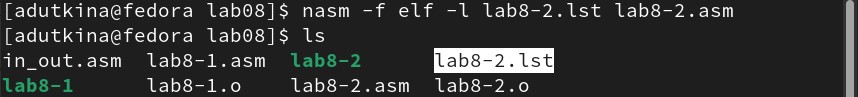


Рис. 9: Создание файла листинга

Откроем файл листинга lab8-2.lst с помощью текстового редактора, ознакомиться с его форматом и содержимым. Рассмотрим содержимое трёх строк (192-194) файла листинга (рис. 10).

Рис. 10: Формат файла листинга

Рис. 10: Формат файла листинга

Можно замеить, что номера строки файла листиинга не совпадают с номерами строк исходного текста программ. Это связано с тем, что перед самой программой в листинге присутствует информация о функциях, используемых из подключаемого файла. В первой из трех рассматриваемых строк нет адреса и машинного кода, так как исходный текст программы - это только комментарий, а значит и машинный код не генерируется. В следующих двух строчках исходного кода содуржатся команды, поэтому у них есть и адрес - смещение машинного кода от начала сегмента, и машинный код - ассемблированная исходная строка, инструкция на машинном языке, вызывающая прерывние ядра: 000000EB и B8[00000000] - адрес и код соответсвенно для команды mov, 000000ED и E81DFFFFFF - для команды call sprint.

Откроем файл с программой lab8-2.asm и в инструкции mov (строка 22) с двумя операндами (ecx, B) удалим второй операнд (рис. 11). Выполним трансляцию с получением файла листинга (рис. 12). В результате работы выдается ошибка с указанием на номер неправильной строки (22). Создается файл lab8-2.lst, в котором добавляется дополнительная строка (с тем же номером) отмеченная звездочками с указанием проблемы (рис. 13).



Рис. 11: Область изменения программы

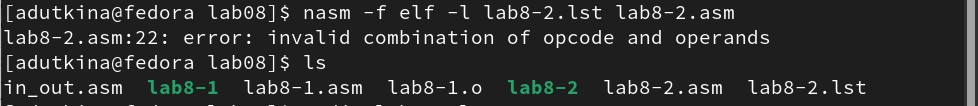


Рис. 12: Результат трансляции файла с ошибкой

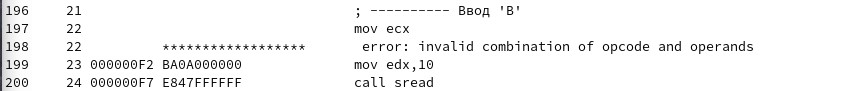


Рис. 13: Запись в файле листинга с указанием ошибки

## 2.3 Самостоятельная работа

Напишем программу нахождения наименьшей из 3 целочисленных переменных a, b и c, где значения переменных равны 32, 6 и 54 соответственно (рис. 14). Создадим исполняемый файл и проверим его работу (рис. 15).

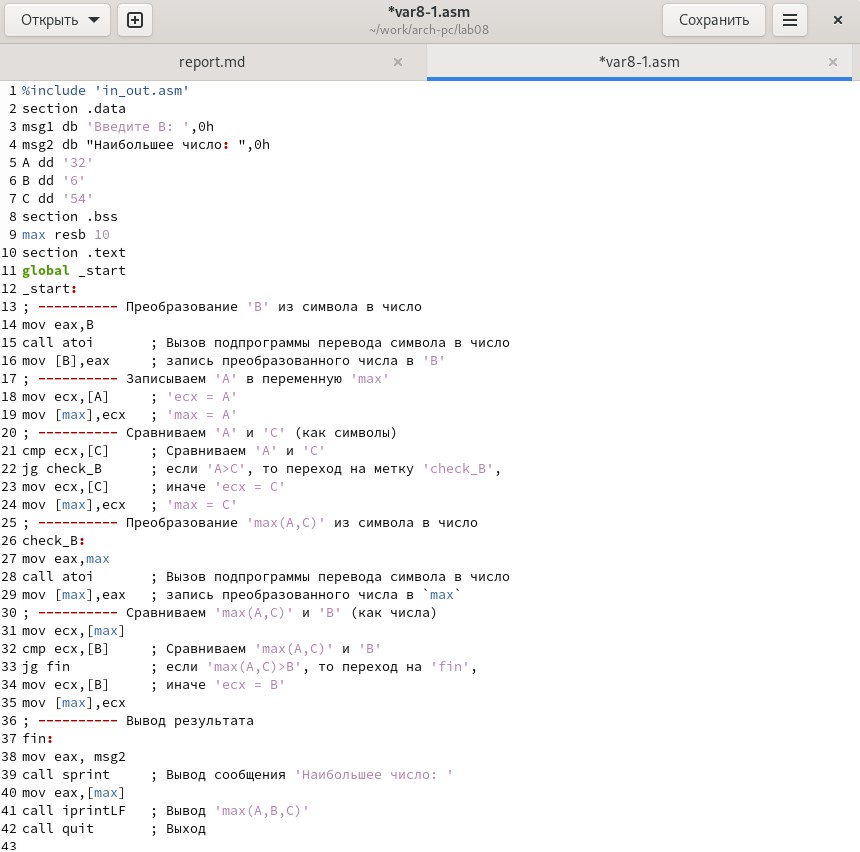


Рис. 14: программа для первого задания самостоятельной работы

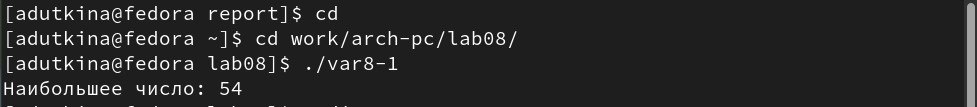


Рис. 15: Результат выполнения первой программы

Напишем программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции №15 (рис. 16) и выводит результат вычислений (рис. 17). Создадим исполняемый файл и проверим его работу для значений x и a равныч 2, 3 соответственно для первого теста и 4, 2 - для второго (рис. 18).

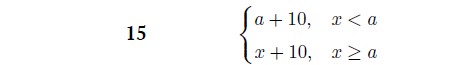


Рис. 16: Функция для выполнения второй программы

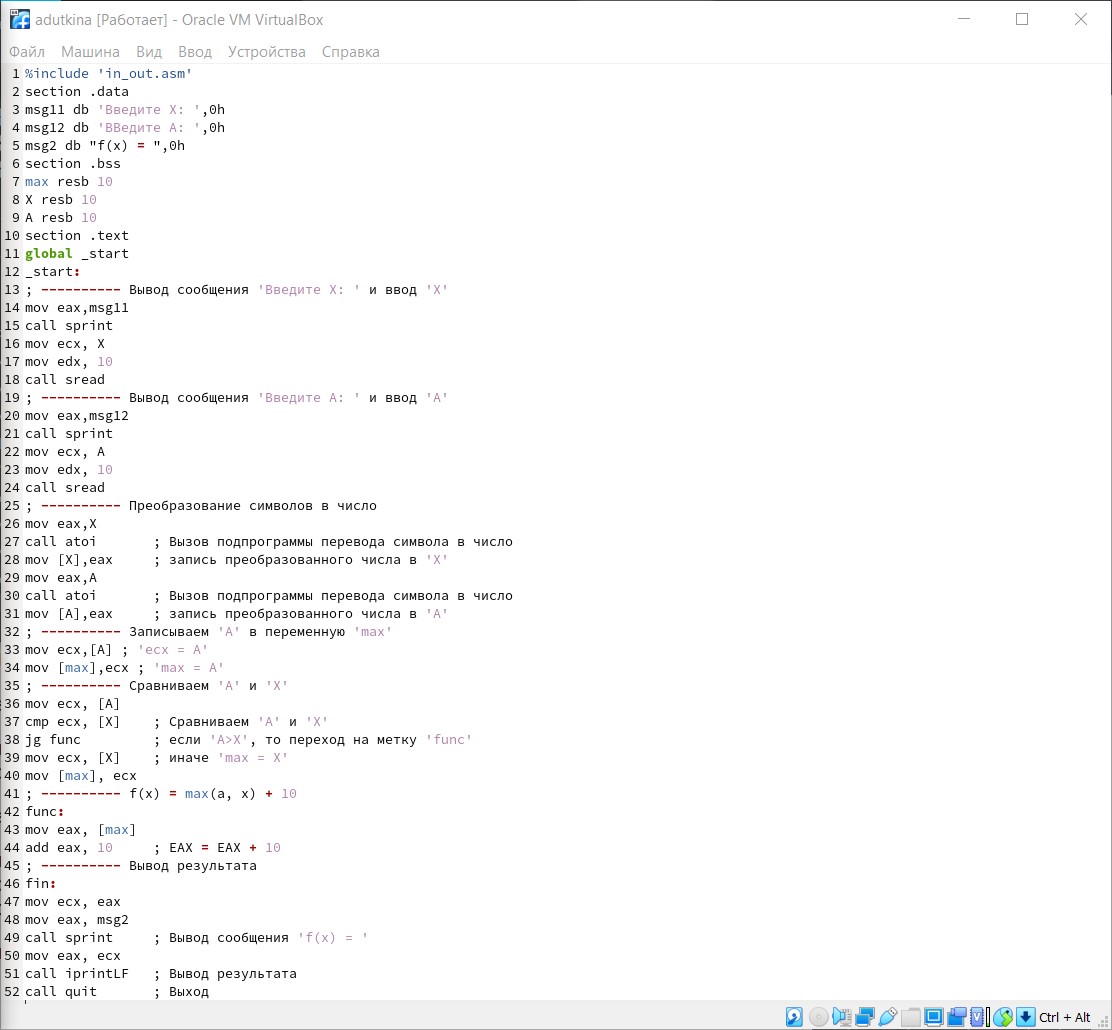


Рис. 17: Вторая программа

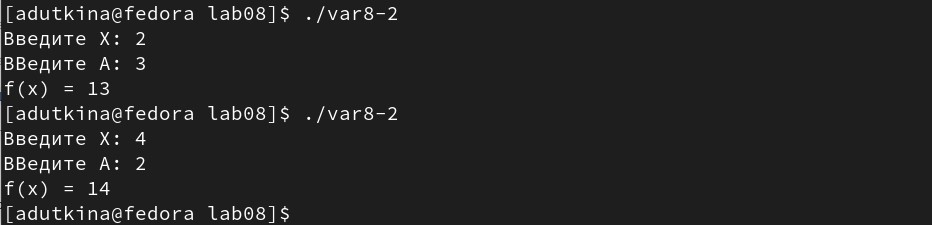


Рис. 18: Результат выполнения второй программы

# 3 Выводы

В ходе данной работы были изучены команды условного, безусловного переходов и назначение, структура файла листинга, приобретены навыки написания программ с использованием переходов.