Отчёт по лабораторной работе №5

2023

Просина Ксения Максимовна

Содержание

# 1 Цель работы

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

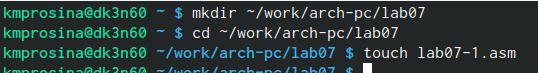
# 2 Теоретическое введение

Безусловный переход выполняется инструкцией jmp (от англ. jump – прыжок), которая включает в себя адрес перехода, куда следует передать управление: jmp Адрес перехода может быть либо меткой, либо адресом области памяти, в которую предва- рительно помещен указатель перехода. Кроме того, в качестве операнда можно использовать имя регистра, в таком случае переход будет осуществляться по адресу, хранящемуся в этом регистре Таблица 7.1. Типы операндов инструкции jmp Тип операнда Описание jmp label переход на метку label jmp [label] переход по адресу в памяти, помеченному меткой label jmp eax переход по адресу из регистра eax

# 3 Выполнение лабораторной работы

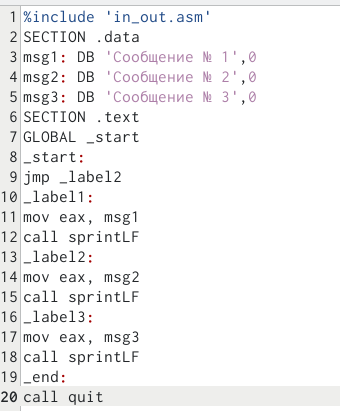
## 3.1 Реализация переходов в NASM

1. Создайте каталог для программам лабораторной работы No 7, перейдите в него и со- здайте файл lab7-1.asm:



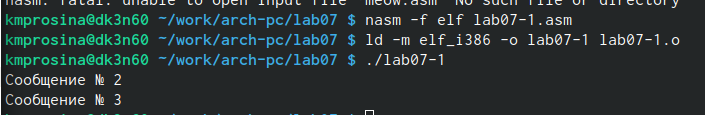
Создание каталога и файла

1. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмот- рим пример программы с использованием инструкции jmp. Введите в файл lab7-1.asm текст программы из листинга



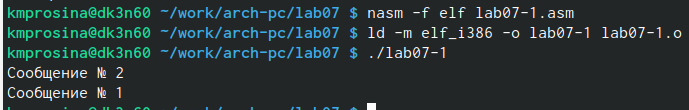
Код

Создайте исполняемый файл и запустите его. Результат работы данной программы будет следующим:



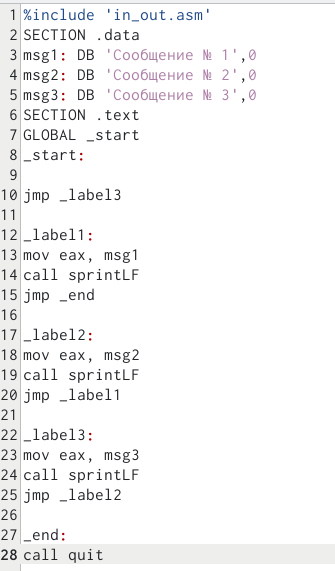
Вывод

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 No 2’, потом ‘Сообщение No 1’ и завершала работу. Для этого в текст программы после вывода сообщения No 2 добавим инструкцию jmp с меткой \_label1 (т.е. переход к инструкциям вывода сообщения No 1) и после вывода сообщения No 1 добавим инструкцию jmp с меткой \_end (т.е. переход к инструкции call quit). Измените текст программы в соответствии с листингом 7.2. Создайте исполняемый файл и проверьте его работу.

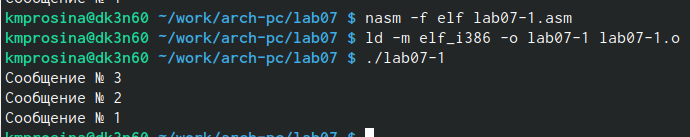


Проверка

Измените текст программы добавив или изменив инструкции jmp, чтобы вывод програм- мы был следующим:

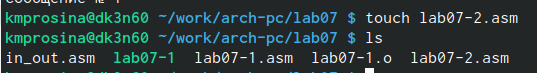


Код

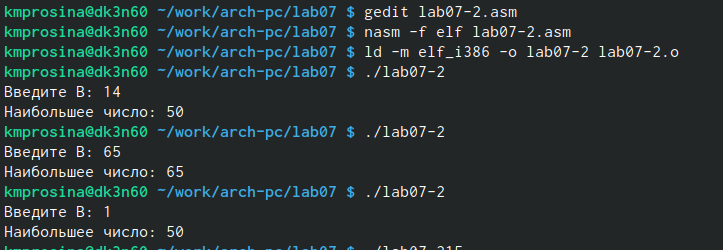


Проверка

1. Использование инструкции jmp приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, т.е. переход дол- жен 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. Создайте файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. Внимательно изучите текст программы из листинга 7.3 и введите в lab7-2.asm. Создайте исполняемый файл и проверьте его работу для разных значений B.



Создание



Проверка

Обратите внимание, 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 преобра- зования 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 числа, над ними можно корректно проводить арифметические операции.

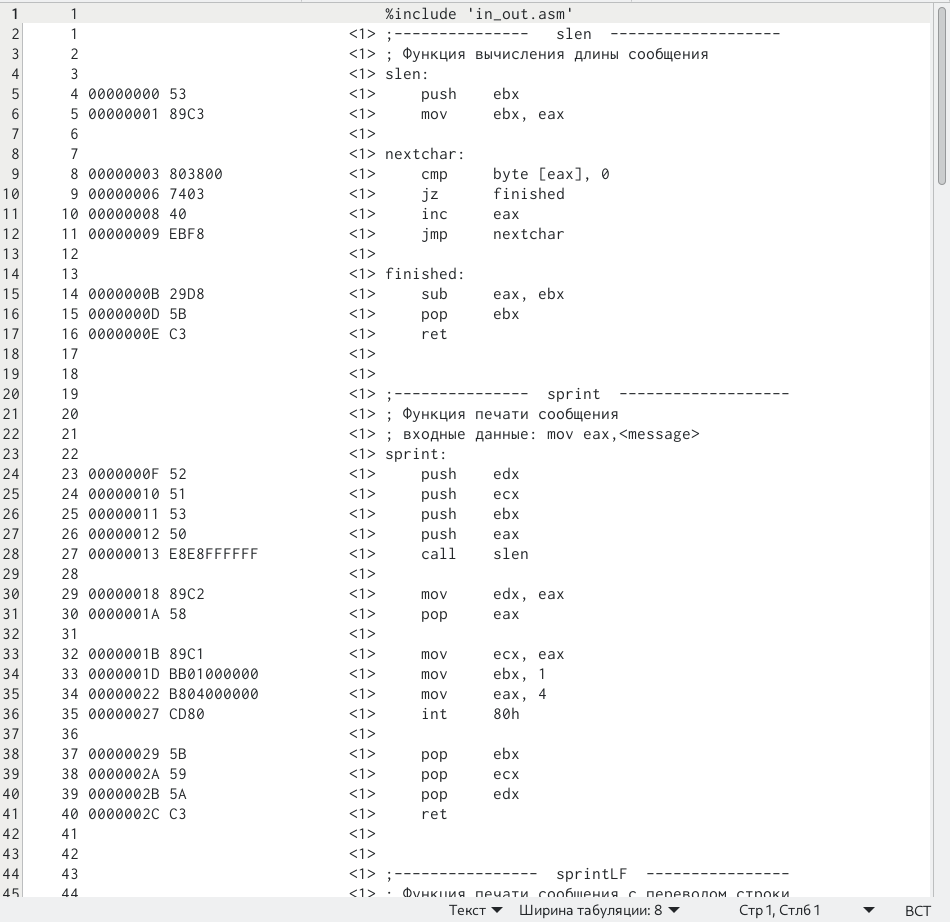
## 3.2 Изучение структуры файлы листинга

1. Обычно nasm создаёт в результате ассемблирования только объектный файл. Получить файл листинга можно, указав ключ -l и задав имя файла листинга в командной строке. Создайте файл листинга для программы из файла lab7-2.asm nasm -f elf -l lab7-2.lst lab7-2.asm

Создание

Создание

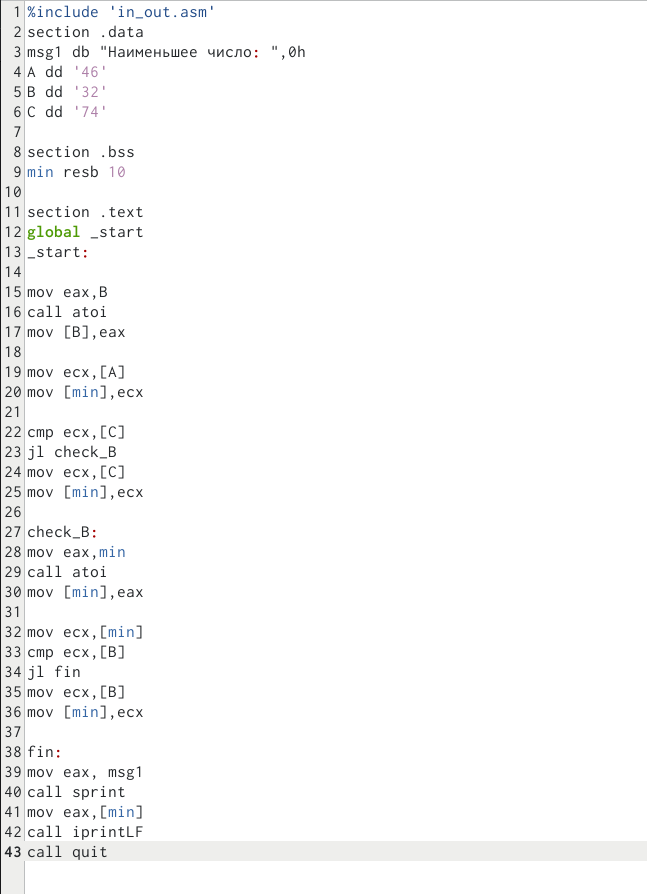
Откройте файл листинга lab7-2.lst с помощью любого текстового редактора, например mcedit: mcedit lab7-2.lst



Листинг

## 3.3 Задание для самостоятельной работы

1. Напишите программу нахождения наименьшей из 3 целочисленных переменных 𝑎,𝑏 и . Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы No 7. Создайте исполняемый файл и проверьте его работу.



Написание кода

Для выполнения задания нужно было создать 3 числа, вписать их в код и сравнить их между собой. В отличие от прошлого кода, для этого я использовала не JG, а JL, что значит, что там сравниваются числа и записывается наименьшее

Проверка

Проверка

1. Напишите программу, которая для введенных с клавиатуры значений 𝑥 и 𝑎 вычисляет значение заданной функции 𝑓(𝑥) и выводит результат вычислений. Вид функции 𝑓(𝑥) выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы No 7. Создайте исполняемый файл и проверьте его работу для значений 𝑥 и 𝑎 из 7.6.

Получившийся код:

%include ‘in\_out.asm’  
section .data  
msg1 db ‘Введите x:’,0h  
msg2 db ‘Введите a:’,0h  
msg3 db “x + a =”,0h  
msg4 db “x =”,0h  
section .bss  
B resb 10 ; x  
C resb 10 ; a  
A resb 10 ; ответ 1  
D resb 10 ; ответ 2

section .text  
global \_start  
\_start:

; ———- Вывод сообщения ‘Введите x:’ mov eax,msg1  
call sprint  
; ———- Ввод ‘x’  
mov ecx,B  
mov edx,10  
call sread  
; ———- Преобразование ‘x’ из символа в число  
mov eax,B  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [B],eax ; запись преобразованного числа в ‘x’  
  
; ———- Вывод сообщения ‘Введите a:’  
mov eax,msg2  
call sprint  
; ———- Ввод ‘a’  
mov ecx,C  
mov edx,10  
call sread  
; ———- Преобразование ‘a’ из символа в число  
mov eax,C  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [C],eax ; запись преобразованного числа в ‘B’  
  
; ———- Преобразование ‘a’ из символа в число  
mov eax,A  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [A],eax ; запись преобразованного числа в ‘B’  
; ———- Преобразование ‘a’ из символа в число  
mov eax,D  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [D],eax ; запись преобразованного числа в ‘B’  
  
mov eax,[B] ; запись х  
add eax,[C] ; добавление к х а  
mov [A],eax ; хапись в ответ 1 получившуюся сумму  
mov edx,[B] ; запись х  
mov [D],edx ; х в ответ 2  
  
; ———- Вывод результата  
fin:  
mov eax, msg3  
call sprint  
mov eax,[A]  
call iprintLF  
mov eax, msg4  
call sprint  
mov eax,[D]  
call iprintLF  
call quit



Проверка

# 4 Выводы

Изучили команды условного и безусловного переходов. Приобрели навыки написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

# Список литературы

https://esystem.rudn.ru/pluginfile.php/2089665/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%967.%20%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D1%8B%20%D0%B1%D0%B5%D0%B7%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B8%20%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%BF%D0%B5%D1%80%D0%B5%D1%85%D0%BE%D0%B4%D0%BE%D0%B2%20%D0%B2%20Nasm.%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%82%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9..pdf