РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ДОКЛАД

на тему «ПРОГРАМИРОВАНИЕ ЦИКЛОВ И ОБРАБОТКА ОРГУМЕНТОВ КОМАНДНОЙ СТРОКИ»

дисциплина: Архитектура компьютера

Студент : Зайцев А.А.

Группа: НКАбд-03-25

№ ст. билета: 1132255647

МОСКВА

2025г.

1. Цель работы

Изучить работу циклов, регистров, функций ввода-вывода в NASM, освоить работу с аргументами командной строки, закрепить навыки компиляции и отладки программ на ассемблере.

2. Ход выполнения работы

2.1. Задание 1 — программа с циклом

Создание рабочей директории и файла программы.

```
azaytsev@fedora:~$ mkdir ~/work/arch-pc/lab08
cd ~/work/arch-pc/lab08
touch lab8-1.asm
```

(команда mkdir, cd, touch lab8-1.asm)

Шаг 2. Открытие файла и попытка компиляции

Открываем файл в редакторе `nano` и пытаемся выполнить первую компиляцию. На данном этапе программа ещё не содержит полного кода.

```
azaytsev@fedora:~/work/arch-pc/lab08$ nano lab8-1.asm
azaytsev@fedora:~/work/arch-pc/lab08$
```

команда nano lab8-1.asm

Шаг 3. Копирование модуля и успешная сборка

```
 GNU nano 8.3
;-------------------------------------------------------------------
; Программа вывода значений регистра 'ecx'
;-------------------------------------------------------------------
%include 'in_out.asm'

SECTION .data
msg1 db 'Введите N: ',0h

SECTION .bss
N: resb 10

SECTION .text
global _start
_start:

; ----- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint

; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread

; ----- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax

; ------ Организация цикла
mov ecx,[N]      ; Счетчик цикла, ecx=N

label:
mov [N],ecx
mov eax,[N]
call iprintLF    ; Вывод значения N

loop label        ; ecx=ecx-1, если ecx !=0 → переход

call quit
```

После копирования файла `in_out.asm` в каталог `lab08` программа собирается без ошибок.

Шаг 6. Запуск программы

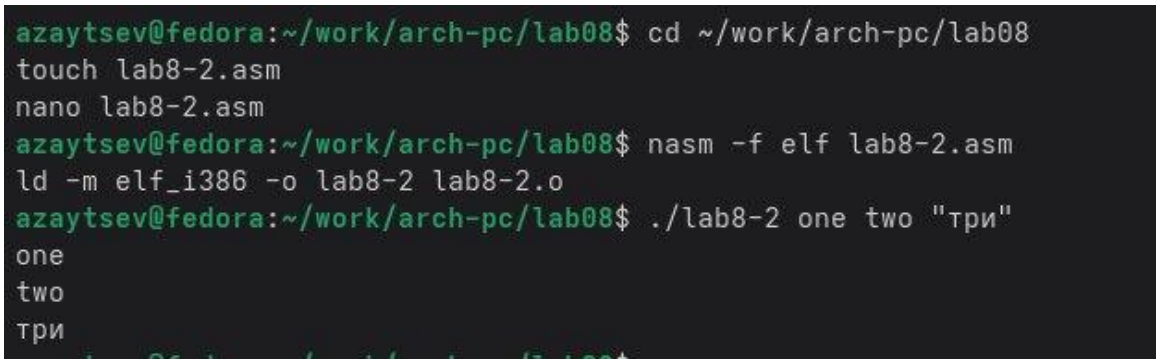
Программа корректно запрашивает число N и выводит значения от N до 1.

(пример работы: ввод 5 → вывод 5,4,3,2,1)

2.2. Задание 2 — вывод аргументов командной строки

Шаг 1. Создание файла программы lab8-2.asm

Создаём файл `lab8-2.asm`, вводим код, производим сборку и компоновку.



(команды touch, nano, nasm, ld для lab8-2)

2.3. Задание 3 — произведение числовых аргументов

Шаг 1. Создание файла lab8-3.asm и компиляция

Создаём программу, которая перемножает все числовые аргументы командной строки.

```
azaytsev@fedora:~/work/arch-pc/lab08$ cd ~/work/arch-pc/lab08
touch lab8-3.asm
nano lab8-3.asm
azaytsev@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
ld -m elf_i386 -o lab8-3 lab8-3.o
./lab8-3 2 3 4
Результат: 1725119000
```

(команды создания, редактирования и сборки lab8-3.asm)

2.4. Самостоятельная часть — реализация функции варианта f(x) = 7 + 2x

Шаг 1. Создание программы main.asm

Пишем программу для вычисления значений функции:

f(x)=7+2xf(x) = 7 + 2xf(x)=7+2x

```
azaytsev@fedora:~/work/arch-pc/lab08$ cd ~/work/arch-pc/lab08
touch main.asm
nano main.asm
azaytsev@fedora:~/work/arch-pc/lab08$
azaytsev@fedora:~/work/arch-pc/lab08$ nasm -f elf main.asm
ld -m elf_i386 -o main main.o
./main 1 2 3 4
Функция: f(x)=7+2x
Результат: 4273834032
```

(создание и сборка файла main.asm)

3. Выводы

В ходе выполнения лабораторной работы были изучены циклы в NASM, работа со стеком, обработка аргументов командной строки, арифметические операции на ассемблере. На практике были устранены ошибки компиляции, найден модуль `in_out.asm`, выполнены три задания и самостоятельная часть. Все программы успешно собраны и протестированы.