

Отчет по лабораторной работе №8

Простейший вариант

Чигладзе Майя Владиславовна

Содержание

1	Цель работы	5
2	Порядок выполнения лабораторной работы	6
2.1	Реализация переходов в NASM	6
2.1.1	Задание 1	6
2.1.2	Задание 2	9
3	Задание для самостоятельной работы	13
3.1	Задание 1	13
4	Выводы	16
	Список литературы	17

Список иллюстраций

2.1	Создание каталога	6
2.2	Проверка	7
2.3	Именить текст программы	8
2.4	Именить текст программы	9
2.5	Запуск программы с аргументами	10
2.6	Запуск программы с аргументами	11
2.7	Запуск программы с аргументами	12
3.1	Код программы	14
3.2	Запуск программы	15

Список таблиц

1 Цель работы

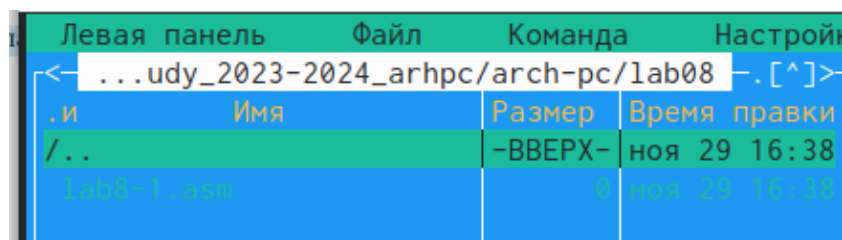
Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки

2 Порядок выполнения лабораторной работы

2.1 Реализация переходов в NASM

2.1.1 Задание 1

Создала каталог для программ лабораторной работы № 8, перешла в него и создала файл lab8-1.asm(рис. [2.1])



The screenshot shows a file manager window with a table of files and directories. The table has four columns: 'Левая панель' (Left panel), 'Файл' (File), 'Команда' (Command), and 'Настройка' (Settings). The first row shows a directory named '..' with the command '..' and the settings 'Имя' (Name), 'Размер' (Size), and 'Время правки' (Modification time). The second row shows a directory named 'lab8-1' with the command 'mkdir' and the settings 'Имя', 'Размер', and 'Время правки'. The third row shows a file named 'lab8-1.asm' with the command 'touch' and the settings 'Имя', 'Размер', and 'Время правки'.

Левая панель	Файл	Команда	Настройка
< ..	udy_2023-2024_arch-pc/lab08	..	[^]>
..	Имя	Размер	Время правки
lab8-1	Имя	Размер	Время правки
lab8-1.asm	Имя	Размер	Время правки

Рис. 2.1: Создание каталога

Проверила его работу (рис. [2.2]). Данный пример показывает, что использование регистра esx в теле цикла loop может привести к некорректной работе программы. Изменила текст программы добавив изменение значение регистра esx в цикле (рис. [2.3]). Регистр esx принимает нечетный значения, N не соответствует проходам цикла.

```
mvchigladze@dk3n35 ~/work/study/2023-20
$ ld -m elf_i386 -o lab8-1 lab8-1.o
mvchigladze@dk3n35 ~/work/study/2023-20
$ ./lab8-1
Введите N: 16
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
mvchigladze@dk3n35 ~/work/study/2023-20
```

Рис. 2.2: Проверка

```
mvchigladze@dk3n35 ~/work/study/2023-202
$ nasm -f elf lab8-1.asm
mvchigladze@dk3n35 ~/work/study/2023-202
$ ld -m elf_i386 -o lab8-1 lab8-1.o
mvchigladze@dk3n35 ~/work/study/2023-202
$ ./lab8-1
Введите N: 10
9
7
5
3
1
mvchigladze@dk3n35 ~/work/study/2023-202
$
```

Рис. 2.3: Именить текст программы

Для использования регистра `ecx` в цикле и сохранения корректности работы программы можно использовать стек. Внесу изменения в текст программы добавив команды `push` и `pop` (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла `loop` (рис. [2.4]). В данном случае `N` соответствует проходам цикла.


```
mvchigladze@dk3n35 ~/work/study/2023-202
$ nasm -f elf lab8-1.asm
mvchigladze@dk3n35 ~/work/study/2023-202
$ ld -m elf_i386 -o lab8-1 lab8-1.o
mvchigladze@dk3n35 ~/work/study/2023-202
$ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
mvchigladze@dk3n35 ~/work/study/2023-202
$
```

Рис. 2.4: Именить текст программы

2.1.2 Задание 2

Рассмотрим программу, которая выводит на экран аргументы командной строки. Создам файл lab8-2.asm в каталоге и введу в него текст программы из листинга 8.2. Создам исполняемый файл и запущу его, указав аргументы (рис. [2.5]). 3 аргумента было обработано программой.

```
mvchigladze@dk3n35 ~/work/study/2023-202
$ nasm -f elf lab8-2.asm
mvchigladze@dk3n35 ~/work/study/2023-202
$ ld -m elf_i386 -o lab8-2 lab8-2.o
mvchigladze@dk3n35 ~/work/study/2023-202
$ ./lab8-2
mvchigladze@dk3n35 ~/work/study/2023-202
$ ./lab8-2
mvchigladze@dk3n35 ~/work/study/2023-202
$ ./lab8-2 2 3 '6'
2
3
6
mvchigladze@dk3n35 ~/work/study/2023-202
$
```

Рис. 2.5: Запуск программы с аргументами

Рассмотрим еще один пример программы которая выводит сумму чисел, которые передаются в программу как аргументы. Создам файл lab8-3.asm в каталоге и введу в него текст программы из листинга 8.3 и запущу (рис. [2.6]).

```

touch lab8-3.asm
mvchigladze@dk3n35 ~/work/study/2023-20
$ nasm -f elf lab8-3.asm
lab8-3.asm:7: error: parser: instructio
mvchigladze@dk3n35 ~/work/study/2023-20
$ nasm -f elf lab8-3.asm
mvchigladze@dk3n35 ~/work/study/2023-20
$ ld -m elf_i386 -o lab8-3 lab8-3.o
mvchigladze@dk3n35 ~/work/study/2023-20
$ ./lab8-3
Результат: 0
mvchigladze@dk3n35 ~/work/study/2023-20
$ ./lab8-3 5 18 23
Результат: 46
mvchigladze@dk3n35 ~/work/study/2023-20
$ █

```

Рис. 2.6: Запуск программы с аргументами

Изменяю текст программы из листинга 8.3 для вычисления произведения аргументов командной строки (рис. [3.2]). Поменяла значение `esi` на 1, чтобы произведение не превращалось в 0, в `ebx` вписала значение аргумента, в `eax` значение того что умножается, умножила на `ebx` `eax`, записала получение в `esi`.

```

sub ecx,1 ; Уменьшаем 'ecx' на 1 (колич
; аргументов без названия программы)
mov esi, 1 ; Используем 'esi' для хране
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще арг
jz _end ; если аргументов нет выходим и
; (переход на метку '_end')
pop eax ; иначе извлекаем следующий арг
call atoi ; преобразуем символ в число
mov ebx,eax
mov eax,esi
mul ebx ; добавляем к промежуточной сум
mov esi,eax

```

Рис. 2.7: Запуск программы с аргументами

3 Задание для самостоятельной работы

3.1 Задание 1

Написала программу, которая находит сумму значений функции $f(x)$ для $x = x_1, x_2, \dots, x_n$, т.е. программа должна выводить значение $f(x_1) + f(x_2) + \dots + f(x_n)$. Значения x_i передаются как аргументы. Вид функции $f(x)$ выбрала из таблицы 8.1 В20. Создала исполняемый файл и проверьте его работу на нескольких наборах $x = x_1, x_2, \dots, x_n$. Мой вариант: $3(10+x)$ Код программы (рис. [??]) и исполнение программы (рис. [??]). Я создала промежуточную переменную еси, при итерации цикла, я добавляла 10, в ебикс записала 3, умножала еаикс на ебикс, добавляла полученное к нашей еси. В конце, в еаиск записала еси и вывела результат.

```

lab0-4.asm [C:\Program Files\Intel\Software\Binaries\IA32\227\31]
%include 'in_out.asm'
SECTION .data
msg db "Результат 20V: ",0
SECTION .text
global _start
_start:

pop ecx.
pop edx.
sub ecx, 1.
mov esi, 0

next:
cmp ecx, 0h
jz _end.

pop eax
call atoi
add eax,10
mov ebx, 3
mul ebx
add esi,eax
loop next.

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

```

Рис. 3.1: Код программы

```
./nasm -f elf lab8-4.asm
mvchigladze@dk3n35 ~/work/study/2023-2024
$ ld -m elf_i386 -o lab8-4 lab8-4.o
mvchigladze@dk3n35 ~/work/study/2023-2024
$ ./lab8-4 1 2 3
Результат 20V: 108
mvchigladze@dk3n35 ~/work/study/2023-2024
$
```

Рис. 3.2: Запуск программы

4 Выводы

В ходе лабораторной работы, я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки

Список литературы