

Отчет по лабораторной работе №8

Архитектура компьютера

Садыков Ильдар Ильфатович

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Программа с использованием циклов:	7
3.2	Исправление цикла для работы с esx :	8
3.3	Обработка аргументов командной строки:	9
3.4	Выполнение программы с значениями полученными из цикла:	11
4	Задание для самостоятельной работы	13
4.1	Программа вычисления суммы значений функции:	13
5	Выводы	15

Список иллюстраций

3.1	Создание файла и запуск программы lab8-1	8
3.2	Запуск программы lab8-1 с операциями с есх внутри цикла	9
3.3	Код программы lab8-2.asm для обработки аргументов	10
3.4	Код программы lab8-3.asm и его запуск	12
4.1	Код программы для вычисления суммы значений функции	14

Список таблиц

1 Цель работы

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

2 Задание

1. Изучить организацию стека и работу с ним в NASM.
2. Освоить инструкции организации циклов.
3. Научиться обрабатывать аргументы командной строки.
4. Написать программу вычисления суммы значений функции для заданных аргументов.

3 Выполнение лабораторной работы

3.1 Программа с использованием циклов:

Создаем каталог для программ лабораторной работы №8 и создаем файл lab8-1.asm. Вводим текст программы для вывода значений регистра esx (рис. 3.1).

```
sadykovii@fedora:~$ cd work/study/2025-2026/Архитектура\ компьютера/arch-pc/labs/lab08/
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ touch lab8-1.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$

12      _start:
13      ; ----- Вывод сообщения 'Введите N: '
14      mov eax,msg1
15      call sprint
16
17      ; ----- Ввод 'N'
18      mov ecx, N
19      mov edx, 10
20      call sread
21
```

Рисунок 3.1: Создание файла и запуск программы lab8-1

3.2 Исправление цикла для работы с ecx :

Следуя инструкциям создаем файл, который изначально выводит не правильно, далее исправляем его и запускаем(рис. 3.2).


```
96598442949659824294965980429496597842949659764294965974429496597242949659704294965
96842949659664294965964429496596242949659604294965958429496595642949659544294965952
42949659504294965948429496594642949659444294965942429496594042949659384294965936429
49659344294965932429496593042949659284294965926429496592442949659224294965920429496
59184294965916429496591442949659124294965910429496590842949659064294965904429496590
24294965900429496589842949658964294965894429496589242949658904294965888429496588642
94965884429496588242949658804294965878429496587642949658744294965872429496587042949
65868429496586642949658644294965862429496586042949658584294965856429496585442949658
52429496585042949658484294965846429496584442949658424294965840429496583842949658364
29496583442949658324294965830429496582842949658264294965824429496582242949658204294
96581842949658164294965814429496581242949658104294965808429496580642949658044294965
80242949658004294965798429496579642949657944294965792429496579042949657884294965786
42949657844294965782429496578042949657784294965776429496577442949657724294965770429
49657684294965766429496576442949657624294965760429496575842949657564294965754429496
575242949657504294965748429496574642949657444294965742429496574042^C
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
nasm -f elf lab8-1.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
ld -m elf_i386 -o lab8-1 lab8-1.o
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
./lab8-1
Введите N: 6
543210sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/l
ab08$
```

```
24     call atoi
25     mov [N],eax
26
27     ; ----- Организация цикла
28     mov ecx,[N] ; Счетчик цикла, `ecx=N`
29
30 label:
31     push ecx ; добавление значения ecx в стек
32     sub ecx,1
33     mov [N],ecx
34     mov eax,[N]
35     call iprint
36     pop ecx ; извлечение значения ecx из стека
37     loop label
38
39     call quit
```

Рисунок 3.2: Запуск программы lab8-1 с операциями с ecx внутри цикла

3.3 Обработка аргументов командной строки:

Создаем файл lab8-2.asm для вывода аргументов командной строки (рис. 3.3).

```

sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
touch lab8-2.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
nasm -f elf lab8-2.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
ld -m elf_i386 -o lab8-2 lab8-2.o
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
./lab8-2 123 57 'II'
123
57
II
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$

```

```

12      ; (второе значение в стеке)
13
14      sub ecx, 1 ; Уменьшаем `ecx` на 1 (количество
15      ; аргументов без названия программы)
16
17      next:
18      cmp ecx, 0 ; проверяем, есть ли еще аргументы
19
20      jz _end ; если аргументов нет выходим из цикла
21      ; (переход на метку `_end`)
22
23      pop eax ; иначе извлекаем аргумент из стека
24      call sprintLF ; вызываем функцию печати
25
26      loop next ; переход к обработке следующего
27      ; аргумента (переход на метку `next`)
28

```

Рисунок 3.3: Код программы lab8-2.asm для обработки аргументов

3.4 Выполнение программы с значениями полученными из цикла:

Создаем файл lab8-3.asm для вывода суммы аргументов командной строки (рис. 3.3).

```
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ touch lab8-3.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ nasm -f elf lab8-3.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$ ./lab8-3 7 4 21 9
Результат: 41
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
```

~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08/lab8-3.asm - Sublime T

```
File Edit Selection Find View Goto Tools Project Preferences Help
in_out.asm x lab8-1.asm x lab8-2.asm x lab8-3.asm x
1 %include '/home/sadykovii/Test/in_out.asm'
2
3 SECTION .data
4     msg db "Результат: ",0
5
6 SECTION .text
7     global _start
8
9     _start:
10    pop ecx ; Извлекаем из стека в `ecx` количество
11           ; аргументов (первое значение в стеке)
12    pop edx ; Извлекаем из стека в `edx` имя программы
13           ; (второе значение в стеке)
14    sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
15           ; аргументов без названия программы)
16
17    mov esi, 0 ; Используем `esi` для хранения
18           ; промежуточных сумм
19
20    next:
21    cmp ecx,0h ; проверяем, есть ли еще аргументы
22    jz _end ; если аргументов нет выходим из цикла
23           ; (переход на метку `_end`)
24
25    pop eax ; иначе извлекаем следующий аргумент из стека
```

Рисунок 3.4: Код программы lab8-3.asm и его запуск

4 Задание для самостоятельной работы

4.1 Программа вычисления суммы значений функции:

Написана программа, которая вычисляет сумму значений функции суммы $f(x) = 3(x + 2)$ для аргументов, переданных через командную строку.

```
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
nasm -f elf lab8-f.asm
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
ld -m elf_i386 -o lab8-f lab8-f.o
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
./lab8-f 1 2 3 4
f(x) = 3(x+2)
Результат: 54
sadykovii@fedora:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08$
```

~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab08/lab8-f.asm - Sublime T

File Edit Selection Find View Goto Tools Project Preferences Help

in_out.asm x lab8-1.asm x lab8-2.asm x lab8-3.asm x lab8-f.asm

```
1 ;-----Variant_7-----
2 %include '/home/sadykovii/Test/in_out.asm'
3
4 SECTION .data
5     f: DB 'f(x) = 3(x+2)',0
6     ans: DB 'Результат: ',0
7
8 SECTION .bss
9     x: RESB 80
10
11 SECTION .text
12     GLOBAL _start
13
14     _start:
15     pop ecx ; Извлекаем из стека в `ecx` количество
16             ; аргументов (первое значение в стеке)
17     pop edx ; Извлекаем из стека в `edx` имя программы
18             ; (второе значение в стеке)
19     sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
20             ; аргументов без названия программы)
21
22     mov esi, 0 ; Используем `esi` для хранения
23             ; промежуточных сумм
24
25
26     mov eax, f
```

Рисунок 4.1: Код программы для вычисления суммы значений функции

5 Выводы

В ходе выполнения лабораторной работы были приобретены навыки написания программ с использованием циклов и обработки аргументов командной строки в NASM. Изучена организация стека и принципы работы с ним. Освоены инструкции организации циклов, в частности инструкция Loop. Написаны программы для обработки аргументов командной строки и вычисления суммы значений функции для заданных аргументов. Полученные знания позволяют создавать более сложные и функциональные программы на языке ассемблера.