

Лабораторная работа №8

Дисциплина: Архитектура компьютера

Жибицкая Евгения Дмитриевна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Задание для самостоятельной работы	9
4	Выводы	11

Список иллюстраций

2.1	Создание каталога и файлов	6
2.2	Файл lab8-1	6
2.3	Третий запуск lab8-1	7
2.4	Запуск lab8-2	7
2.5	Запуск lab8-3	7
2.6	Запуск измененного файла lab8-3	8
3.1	Код программы	9
3.2	Запуск программы	10

Список таблиц

1 Цель работы

Продолжение освоения языка ассемблера. Изучение написания программ с использованием циклов и обработкой аргументов командной строки.

2 Выполнение лабораторной работы

Для начала создадим все необходимые файлы и каталоги, также не забудем скопировать файл `in_out.asm`(рис. 2.1).

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

Рис. 2.1: Создание каталога и файлов

Затем создадим файл `lab8-1.asm`, заполним его текстом из Лиситнга 8.1, создадим исполняемый файл и запустим(рис. 2.2).

```
edzhbitskaya@edzhbitskaya: ~/work/arch-pc/lab08
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 5
5
4
3
2
1
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$
```

Рис. 2.2: Файл lab8-1

Немного изменим содержимое кода и еще раз запустим(при этом запуске программа выводит бесконечный набор чисел).

Еще раз изменим код, добавив команды `push` и `pop` (рис. 2.3).

```
edzhbitskaya@edzhbitskaya: ~/work/arch-pc/lab08
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 3
2
1
0
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$
```

Рис. 2.3: Третий запуск lab8-1

Для изучения работы с аргументами создадим lab8-2.asm, введем текст Листинга 8.2, создадим исполняемый файл и запустим программу(рис. 2.4).

```
edzhbitskaya@edzhbitskaya: ~/work/arch-pc/lab08
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
2
аргумент 3
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$
```

Рис. 2.4: Запуск lab8-2

Наконец создадим файл lab8-3.asm, заполним его текстом Листинга 8.3, создадим исполняемый файл и запустим(рис. 2.5). В конце изменим код так, чтобы вместо суммы програама считала произведение введенных значений(рис. 2.6)

```
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ touch lab8-3.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ gedit lab8-3.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ./lab8-3 5 67 8 92
Результат: 172
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$
```

Рис. 2.5: Запуск lab8-3

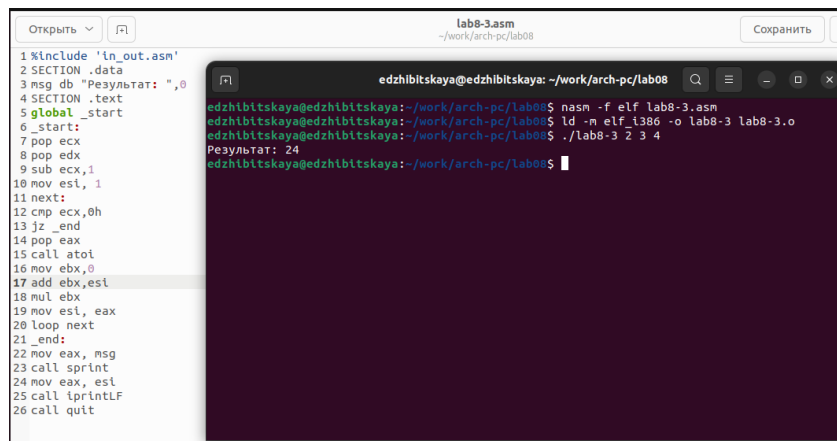
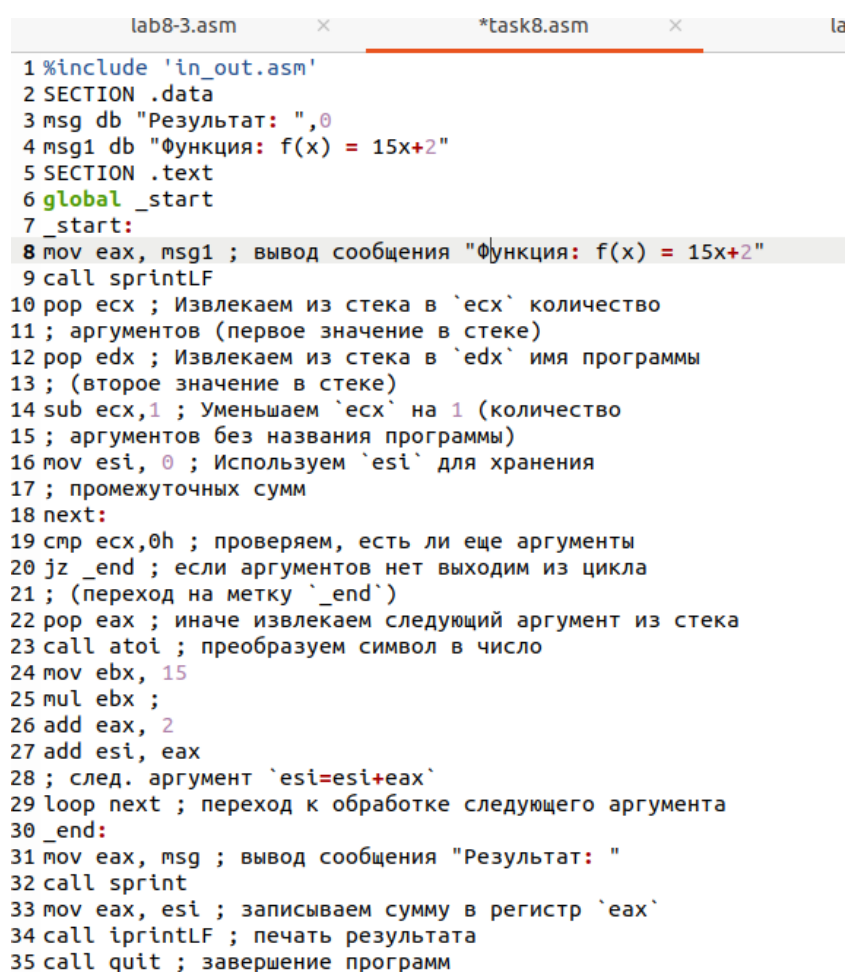


Рис. 2.6: Запуск измененного файла lab8-3

3 Задание для самостоятельной работы

Напишем программу, которая находит сумму значений функции из 11 варианта(рис. 3.1),(рис. 3.2)



```
lab8-3.asm x *task8.asm x la
1 %include 'in_out.asm'
2 SECTION .data
3 msg db "Результат: ",0
4 msg1 db "Функция: f(x) = 15x+2"
5 SECTION .text
6 global _start
7 _start:
8 mov eax, msg1 ; вывод сообщения "Функция: f(x) = 15x+2"
9 call sprintf
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 mov esi, 0 ; Используем `esi` для хранения
17 ; промежуточных сумм
18 next:
19 cmp ecx,0h ; проверяем, есть ли еще аргументы
20 jz _end ; если аргументов нет выходим из цикла
21 ; (переход на метку `_end`)
22 pop eax ; иначе извлекаем следующий аргумент из стека
23 call atoi ; преобразуем символ в число
24 mov ebx, 15
25 mul ebx ;
26 add eax, 2
27 add esi, eax
28 ; след. аргумент `esi=esi+eax`
29 loop next ; переход к обработке следующего аргумента
30 _end:
31 mov eax, msg ; вывод сообщения "Результат: "
32 call sprintf
33 mov eax, esi ; записываем сумму в регистр `eax`
34 call iprintf ; печать результата
35 call quit ; завершение программ
```

Рис. 3.1: Код программы

```
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ touch task8.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ gedit task8.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ nasm -f elf task.asm
nasm: fatal: unable to open input file 'task.asm' No such file or directory
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ nasm -f elf task8.asm
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ld -m elf_i386 -o task8 task8.o
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$ ./task8 1 2 3
Функци:  $f(x) = 15x+2$ 
Результат: 96
edzhbitskaya@edzhbitskaya:~/work/arch-pc/lab08$
```

Рис. 3.2: Запуск программы

4 Выводы

В ходе работы было произведено знакомство с принципами работы циклов и использовании аргументов, написана небольшая программа

...