

Отчёт по лабораторной работе №8

Архитектура компьютера

Андреева Софья Владимировна

Содержание

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

Список иллюстраций

2.1	Результат работы файла lab8-1.asm	5
2.2	Запуск измененного файла.	6
2.3	Запуск измененного файла.	6
2.4	Результат работы файла lab8-1.asm.	7
2.5	Работа файла lab7-3.asm.	7
2.6	Измененный файл.	8
2.7	Запуск измененного файла.	8
3.1	Текст программы	10
3.2	Запуск файла	10

1 Цель работы

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

2 Выполнение лабораторной работы

Создадим каталог для программ лабораторной работы № 8, перейдем в него и создадим файл lab8-1.asm. Внимательно изучим текст программы из листинга 8.1. Введем в файл lab8-1.asm текст программы. Создадим исполняемый файл и запустим его. (рис. 2.1).

```
svandreeva@svandreeva-VirtualBox:~$ mkdir ~/work/arch-pc/lab08
svandreeva@svandreeva-VirtualBox:~$ cd ~/work/arch-pc/lab08
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ touch lab8-1.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ mc

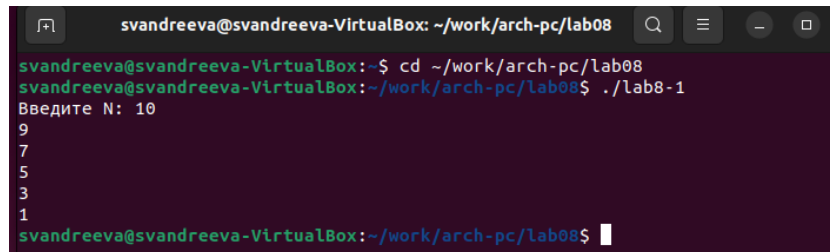
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 8
8
7
6
5
4
3
2
1
```

Рис. 2.1: Результат работы файла lab8-1.asm

Изменим текст программы добавив изменение в значение регистра ecx в цикле. Добавим строчку с уменьшением ecx на 1:

```
label:
sub ecx,1 ; `ecx=ecx-1`
mov [N],ecx
mov eax,[N]
call iprintLF
loop label
```

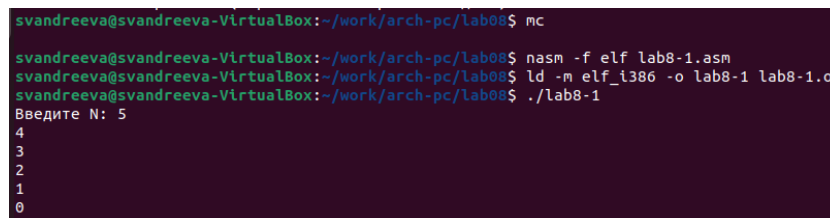
Создадим исполняемый файл и запустим его. В результате в ходе выполнения одной итерации цикла регистр уменьшается на 2, и общее количество итерации становится меньше, при этом в зависимости от ввода N, проверка $ecx = 0$ может не наступить, что приведет к бесконечному выполнению программы (рис. 2.2).



```
svandreeva@svandreeva-VirtualBox: ~/work/arch-pc/lab08
svandreeva@svandreeva-VirtualBox:~$ cd ~/work/arch-pc/lab08
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
7
5
3
1
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$
```

Рис. 2.2: Запуск измененного файла.

Внесем изменения в текст программы добавив команды push и pop для сохранения значения счетчика цикла loop. Создадим исполняемый файл и проверим его работу. Количество итерации цикла совпадает со значением N. (рис. 2.3)



```
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ mc
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 5
4
3
2
1
0
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$
```

Рис. 2.3: Запуск измененного файла.

Рассмотрим программу, которая выводит на экран аргументы командной строки. Внимательно изучим текст программы (Листинг 8.2). Создадим файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и введем в него текст программы из листинга 8.2. Создадим исполняемый файл и запустим его, указав аргументы. Программа обработала 4 аргумента, разделенных пробелами. (рис. 2.4).

```

svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент3'
аргумент1
аргумент
2
аргумент3

```

Рис. 2.4: Результат работы файла lab8-1.asm.

Рассмотрим еще один пример программы которая выводит сумму чисел, кото-
рые передаются в программу как аргументы. Создадим файл lab8-3.asm в катало-
ге ~/work/archpc/lab08 и введем в него текст программы из листинга 8.3.Создадим
исполняемый файл и запустим его.(рис. 2.5).

```

svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nc
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47

```

Рис. 2.5: Работа файла lab7-3.asm.

Изменим текст программы из листинга 8.3 для вычисления произведения
аргументов командной строки.(рис. 2.6).

```

_start:
pop ecx ; Извлекаем из стека в `ecx` количество
; аргументов (первое значение в стеке)
pop edx ; Извлекаем из стека в `edx` имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
; аргументов без названия программы)
mov esi, 1 ; Используем `esi` для хранения
; промежуточных произведений
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку `_end`)
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mov ebx, eax ; сохраняем значений аргумент в ebx для умножения
mov eax, esi ; сохраняем esi в eax, чтобы домножить на аргумент
mul ebx ; умножаем eax*ebx == промежуточное произведение на аргумент
mov esi, eax ; сохраняем значение получившейся суммы обратно в esi
loop next ; переход к обработке следующего аргумента
_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint
mov eax, esi ; записываем сумму в регистр `eax`
call iprintLF ; печать результата
call quit ; завершение программы

```

Рис. 2.6: Измененный файл.

Создадим исполняемый файл и запустим его. Всё получилось, $6510 \cdot 3 = 900$. (рис. 2.7).

```

svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ mc
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-3 6 5 10 3
Результат: 900

```

Рис. 2.7: Запуск измененного файла.

3 Задание для самостоятельной работы.

Для выполнения заданий выбран вариант 12, полученный при выполнении лабораторной работы №6. Напишем программу в файле samr12.asm для нахождения суммы значений функции $f(x)=15x-9$ (рис. 3.1).

```

GNU nano 6.2 /home/svandreev
msg2 db "Результат: ",0
SECTION .text
global _start

_start:

mov eax, msg1
call sprintf

mov ebx, 15 ; регистр ebx используем для умножения

por ecx ; Извлекаем из стека в `ecx` количество
; аргументов (первое значение в стеке)
por edx ; Извлекаем из стека в `edx` имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
; аргументов без названия программы)
mov esi, 0 ; Используем `esi` для хранения
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку `_end`)
por eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mul ebx ; eax=eax*ebx
sub eax,9 ; eax=eax-9
add esi,eax ; добавляем к промежуточной сумме
; след. аргумент `esi=esi+eax`
loop next ; переход к обработке следующего аргумента
_end:
mov eax, msg2 ; вывод сообщения "Результат: "
call sprintf
mov eax, esi ; записываем сумму в регистр `eax`
call iprintfLF ; печать результата
call quit ; завершение программы

```

Рис. 3.1: Текст программы

Создадим исполняемый файл и запустим его. Проверим вычисления. Все исполнилось корректно (рис. 3.2).

```

svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ touch samr12.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nc

svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf samr12.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o samr12 samr12.o
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./samr12 1 2 3
Функция: f(x)=15*x-9
Результат: 63
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab08$ ./samr12 10 10 10
Функция: f(x)=15*x-9
Результат: 423

```

Рис. 3.2: Запуск файла

4 Выводы

Я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.