

Отчёт по лабораторной работе №8

Дисциплина: архитектура компьютера

Горобцова Арина Романовна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	17

Список иллюстраций

3.1	Программа lab8-1.asm	7
3.2	Запуск программы lab8-1.asm	8
3.3	Программа lab8-1.asm	9
3.4	Запуск программы lab8-1.asm	9
3.5	Программа lab8-1.asm	10
3.6	Запуск программы lab8-1.asm	11
3.7	Программа lab8-2.asm	12
3.8	Запуск программы lab8-2.asm	12
3.9	Программа lab8-3.asm	13
3.10	Запуск программы lab8-3.asm	13
3.11	Программа lab8-3.asm	14
3.12	Запуск программы lab8-3.asm	14
3.13	Программа lab8-4.asm	15
3.14	Запуск программы lab8-4.asm	16

Список таблиц

1 Цель работы

Целью работы является приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

2 Задание

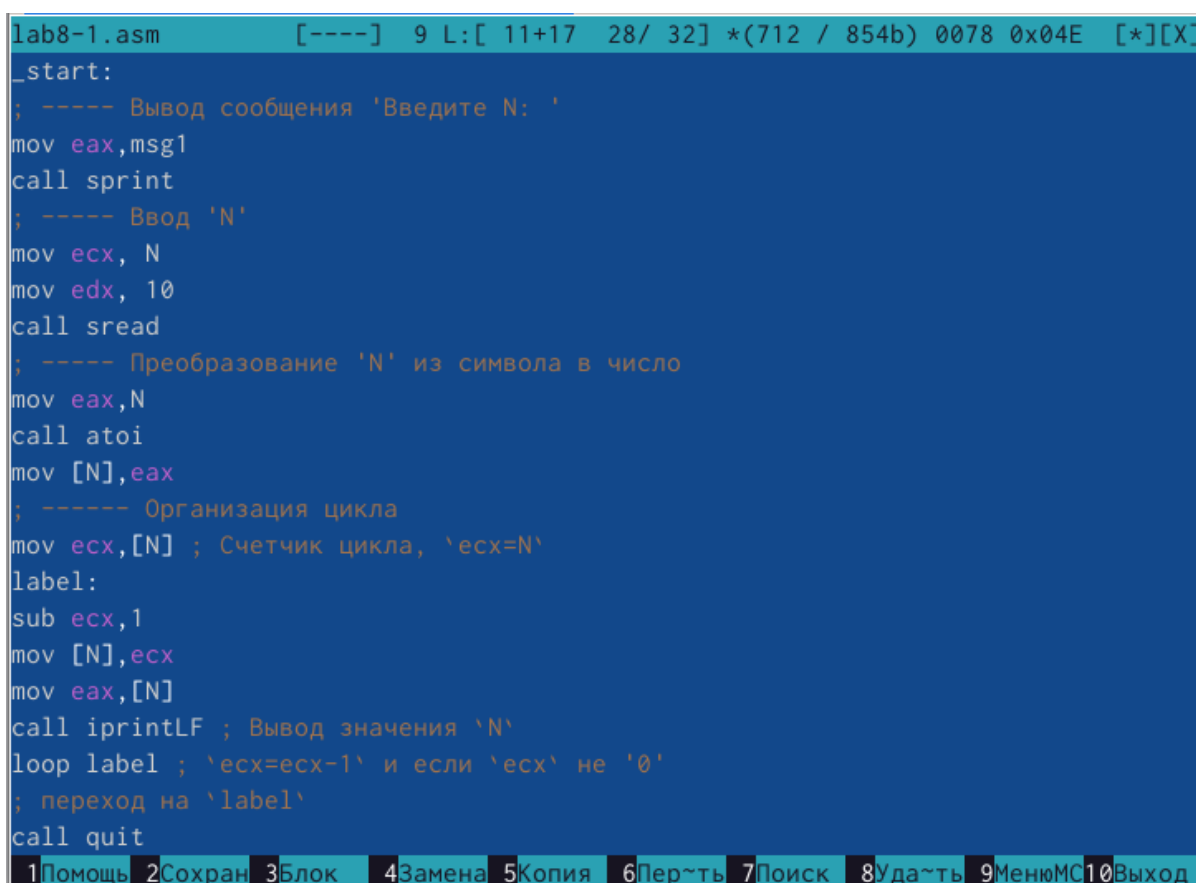
1. Реализация циклов в NASM
2. Обработка аргументов командной строки
3. Задание для самостоятельной работы

3 Выполнение лабораторной работы

1. Реализация циклов в NASM

Создали каталог для программ лабораторной работы № 8, переходим в него и создаем файл lab8-1.asm

Написали в файл lab8-1.asm текст программы из листинга 8.1. (рис. 3.1). Создал исполняемый файл и проверил его работу. (рис. 3.2).



```
lab8-1.asm [----] 9 L:[ 11+17 28/ 32] *(712 / 854b) 0078 0x04E [*][X]
_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:
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Переть 7Поиск 8Удалить 9МенюМС10Выход
```

Рис. 3.1: Программа lab8-1.asm

```

argorobcova@dk3n35 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 6
6
5
4
3
2
1
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 5
5
4
3
2
1
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ █

```

Рис. 3.2: Запуск программы lab8-1.asm

Данный пример показывает, что использование регистра `ecx` в теле цикла `loop` может привести к некорректной работе программы (рис. 3.3). Изменяем текст программы добавив изменение значение регистра `ecx` в цикле: Создаем исполняемый файл и проверяем его работу (рис. 3.4). Какие значения принимает регистр `ecx` в цикле? Соответствует ли число проходов цикла значению `N`, введенному с клавиатуры?

Программа запускает бесконечный цикл при нечетном `N` и выводит только нечетные числа при четном `N`.


```
lab8-1.asm      [----]  9 L:[ 11+17  28/ 32] *(712 / 854b) 0078 0x04E  [*][X]
_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:
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Переть 7Поиск 8Удасть 9МенюМС10Выход
```

Рис. 3.3: Программа lab8-1.asm

```
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 6
5
3
1
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ █
```

Рис. 3.4: Запуск программы lab8-1.asm

Для использования регистра ecx в цикле и сохранения корректности работы программы можно использовать стек. Вносим изменения в текст программы добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop. (рис. 3.5). Создаем исполняемый файл и

проверяем его работу. (рис. 3.6). Соответствует ли в данном случае число проходов цикла значению N введенному с клавиатуры?

Программа выводит числа от N-1 до 0, число проходов цикла соответствует N.

```
lab8-1.asm      [-M--] 10 L:[ 11+21  32/ 33] *(872 / 882b) 0010 0x00A  [*][X]
_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:
push ecx ; добавление значения ecx в стек
sub  ecx,1
mov  [N],ecx
mov  eax,[N]
call iprintLF
pop  ecx ; извлечение значения ecx из стека
loop label
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюМС10Выход
```

Рис. 3.5: Программа lab8-1.asm

```

argorobcova@dk3n35 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 6
5
4
3
2
1
0
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 5
4
3
2
1
0
argorobcova@dk3n35 ~/work/arch-pc/lab08 $

```

Рис. 3.6: Запуск программы lab8-1.asm

2. Обработка аргументов командной строки

Создаем файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и вводим в него текст программы из листинга 8.2 (рис. 3.7). Создаем исполняемый файл и запускаем его, указав аргументы (рис. 3.8). Сколько аргументов было обработано программой?

Программа обработала 5 аргументов.

```
lab8-2.asm [----] 0 L:[ 1+ 8 9/ 20] *(330 / 943b) 0115 0x073 [*][X]
#include 'in_out.asm'
SECTION .text
global _start
_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)
next:
    cmp ecx, 0 ; проверяем, есть ли еще аргументы
    jz _end ; если аргументов нет выходим из цикла
             ; (переход на метку '_end')
    pop eax ; иначе извлекаем аргумент из стека
    call sprintf ; вызываем функцию печати
    loop next ; переход к обработке следующего
             ; аргумента (переход на метку 'next')
_end:
    call quit
```

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Переть 7Поиск 8Удалить 9МенюМС10Выход

Рис. 3.7: Программа lab8-2.asm

```
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-2
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-2 A r i n a
A
r
i
n
a
argorobcova@dk3n35 ~/work/arch-pc/lab08 $
```

Рис. 3.8: Запуск программы lab8-2.asm

Рассмотрим еще один пример программы которая выводит сумму чисел, которые передаются в программу как аргументы (рис. 3.9) и (рис. 3.10).

```

lab8-3.asm      [-M--] 30 L:[ 12+19 31/ 33] *(1402/1461b) 1072 0x430  [*][X]
; 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')
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mov ebx,eax
mov eax,esi
mul ebx
mov esi,eax ; добавляем к промежуточной сумме
; след. аргумент 'esi=esi+eax'
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
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюМС10Выход

```

Рис. 3.9: Программа lab8-3.asm

```

argorobcova@dk3n35 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-3
Результат: 0
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-3 2 6 8
Результат: 16
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-3 2 6 8
Результат: 16
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-3 2 2 2
Результат: 6
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ █

```

Рис. 3.10: Запуск программы lab8-3.asm

Изменяем текст программы из листинга 8.3 для вычисления произведения аргументов командной строки (рис. 3.11) и (рис. 3.12).

```

lab8-3.asm      [-M--] 30 L:[ 12+19  31/ 33] *(1402/1461b) 1072 0x430  [*][X]
; 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`)
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mov ebx,eax
mov eax,esi
mul ebx
mov esi,eax ; добавляем к промежуточной сумме
; след. аргумент `esi=esi+eax`
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

```

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюМС10Выход

Рис. 3.11: Программа lab8-3.asm

```

argorobcova@dk3n35 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-3 2 6 8
Результат: 96
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-3 2 2 2
Результат: 8
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ █

```

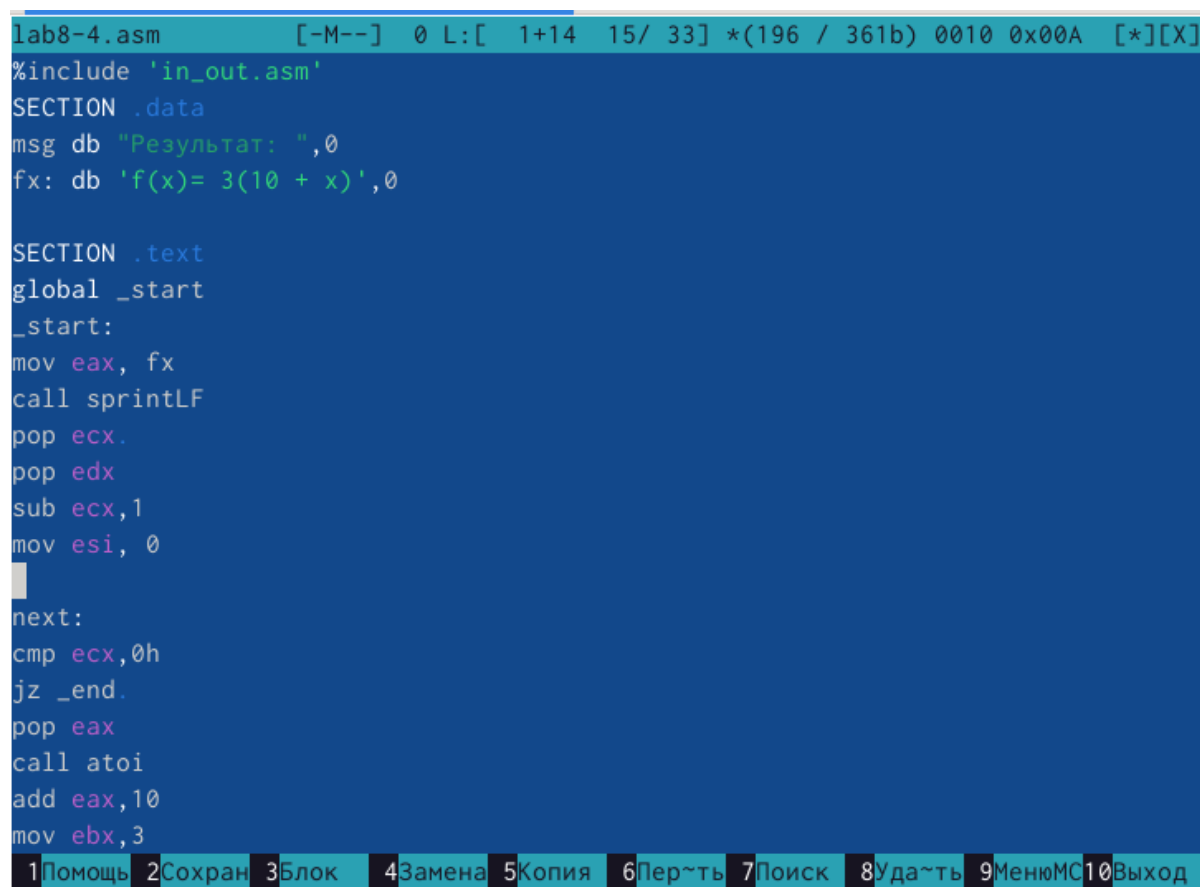
Рис. 3.12: Запуск программы lab8-3.asm

3. Задание для самостоятельной работы

Напишем 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 передаются как аргументы. Вид функции $f(x)$ выбрать из таблицы 8.1 вариантов заданий в соответствии с вариантом, полученным при выполнении

лабораторной работы № 7. (рис. 3.13). Создайте исполняемый файл и проверьте его работу на нескольких наборах x . (рис. 3.14).

для варианта 20 $f(x) = 3(10 + x)$



```
lab8-4.asm      [-M--]  0 L:[ 1+14 15/ 33] *(196 / 361b) 0010 0x00A  [*][X]
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
fx: db 'f(x)= 3(10 + x)',0

SECTION .text
global _start
_start:
mov eax, fx
call sprintLF
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
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюМС10Выход
```

Рис. 3.13: Программа lab8-4.asm

```
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-4 1
f(x)= 3(10 + x)
Результат: 33
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-4 2
f(x)= 3(10 + x)
Результат: 36
argorobcova@dk3n35 ~/work/arch-pc/lab08 $ ./lab8-4 2 4 6 8
f(x)= 3(10 + x)
Результат: 180
argorobcova@dk3n35 ~/work/arch-pc/lab08 $
```

Рис. 3.14: Запуск программы lab8-4.asm

4 Выводы

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