

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

дисциплина: Архитектура компьютера

Холопов Илья Алексеевич

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	11

Список иллюстраций

2.1	Создание каталога и файла в нем lab8-1.asm	6
2.2	Содержимое файла lab8-1.asm	6
2.3	Создание и запуск исполняемого файла	7
2.4	Результат работы измененной программы	7
2.5	Результат работы программы со стеком	7
2.6	Содержимое файла lab8-2.asm	8
2.7	Результат программы, обрабатывающей аргументы	8
2.8	Текст программы lab8-3.asm	9
2.9	Результат работы программы, выводящей сумму чисел	9
2.10	Результат программы, выводящей произведение чисел	9
2.11	Текст программы task.asm	10
2.12	Результат работы программы, вычисляющей выражение	10

Список таблиц

1 Цель работы

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

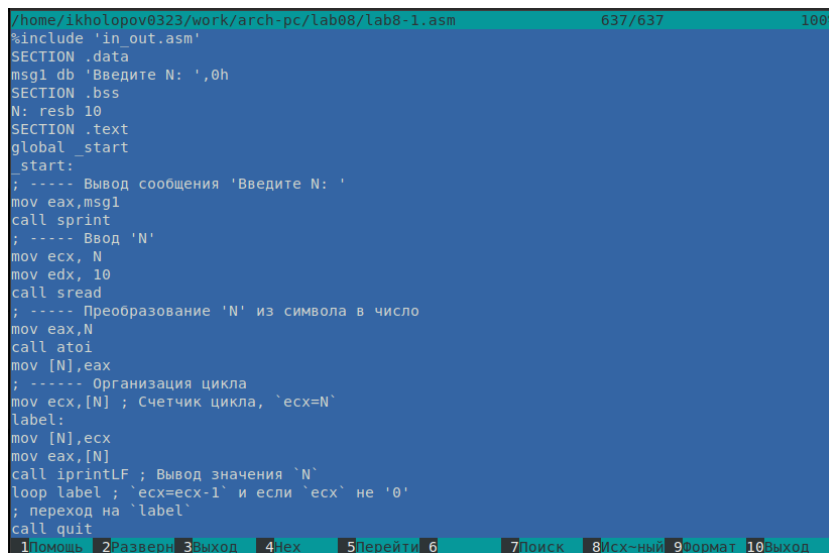
2 Выполнение лабораторной работы

Создадим каталог для лабораторной работы № 8, перейдем в него и создадим файл lab8-1.asm (рис. 2.1).

```
ikholopov0323@ikholopov0323:~$ mkdir work/arch-pc/lab0
lab04/ lab05/ lab06/ lab07/
ikholopov0323@ikholopov0323:~$ mkdir work/arch-pc/lab08
ikholopov0323@ikholopov0323:~$ cd work/arch-pc/lab08
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ touch lab8-1.asm
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$
```

Рис. 2.1: Создание каталога и файла в нем lab8-1.asm

Введем в файл lab8-1.asm текст программы с использованием цикла (рис. 2.2).



```
/home/ikholopov0323/work/arch-pc/lab08/lab8-1.asm 637/637 100%
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_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:
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 2Заверш 3Выход 4lex 5Перейти 6 7Поиск 8Иск-ный 9Формат 10Выход
```

Рис. 2.2: Содержимое файла lab8-1.asm

Создадим исполняемый файл и запустим его (рис. 2.3).

```

ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 3
3
2
1
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$

```

Рис. 2.3: Создание и запуск исполняемого файла

Изменим программу таким образом, чтобы в цикле loop использовался регистр ехх. На рис. 2.4 видно, что вывод программы не соответствует введенному числу N.

```

ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 3
2
0
4294967294
4294967292
4294967290
4294967288

```

Рис. 2.4: Результат работы измененной программы

Снова изменим программу, чтобы она использовала стек для ехх (рис. 2.5). В данном случае вывод программы соответствует введенному числу N.

```

ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 3
3
2
1
0
ikholopov0323@ikholopov0323:~/work/arch-pc/lab08$

```

Рис. 2.5: Результат работы программы со стеком

Создадим файл lab8-2.asm и введем в него текст программы, использующей аргументы командной строки (рис. 2.6).

```

/home/ikhologov0323/work/arch-pc/lab08/lab8-2.asm 944/944 100%
#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 2Разверн 3Выход 4Нех 5Перейти 6 7Поиск 8Исх-ный 9Формат 10Выход

Рис. 2.6: Содержимое файла lab8-2.asm

Создадим исполняемый файл и запустим его, указав аргументы (2.7). Программа обработала 5 аргументов: имя программы, аргумент1, аргумент, 2, 'аргумент3'.

```

ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$

```

Рис. 2.7: Результат программы, обрабатывающей аргументы

Создадим файл lab8-3.asm и введем в него текст программы, выводящей сумму чисел командной строки (рис. 2.8).


```

/home/ikhlopov0323/work/arch-pc/lab08/lab8-3.asm 1387/1387 100%
#include 'in out.asm'
SECTION .data
msg db "Результат: ",0
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)
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
add 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 iprintf ; печать результата
call quit
1Помощь 2Разверн 3Выход 4Нех 5Перейти 6 7Поиск 8Сх-ный 9Формат 10Выход

```

Рис. 2.8: Текст программы lab8-3.asm

Создайте исполняемый файл и запустите его, указав аргументы (рис. 2.9).

```

ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ ./lab8-3 1 2 3 4 5 10
Результат: 25
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$

```

Рис. 2.9: Результат работы программы, выводящей сумму чисел

Изменим текст программы lab8-3.asm для нахождения произведения чисел. Создадим и запустим исполняемый файл, указав аргументы (рис. 2.10).

```

ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ ./lab8-3 1 2 3
Результат: 6
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ ./lab8-3 1 2 3 4
Результат: 24
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$ ./lab8-3 1 2 3 4 5
Результат: 120
ikhlopov0323@ikhlopov0323:~/work/arch-pc/lab08$

```

Рис. 2.10: Результат программы, выводящей произведение чисел

Создадим файла task.asm и введем в него текст программы, выводящей значение выражения $f(x_1) + f(x_2) + f(x_n)$, где $f(x) = 4x + 3$ (рис. 2.11).

```
/home/ikhologov0323/work/arch-pc/lab08/task.asm 1134/1134 100%
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx ; Извлекаем из стека в `ecx` количество
pop edx ; Извлекаем из стека в `edx` имя программы
sub ecx,1 ; Уменьшаем `ecx` на 1 (количество)
mov esi, 0 ; Используем `esi` для хранения

next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mov ebx,eax
mov eax,4
mul ebx ; добавляем к промежуточной сумме
add eax,3
add eax,esi
mov 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 iprintf ; печать результата
call quit

1Помощь 2Разверн 3Выход 4Чех 5Перейти 6 7Поиск 8Исх-ный 9Формат 10Выход
```

Рис. 2.11: Текст программы task.asm

Создадим и запустим исполняемый файл программы, указав аргументы (рис. 2.12).

```
ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$ nasm -f elf task.asm
ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$ ld -m elf_i386 -o task task.o
ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$ ./task 1 2
Результат: 18
ikhologov0323@ikhologov0323:~/work/arch-pc/lab08$ ./task 1 2 3
Результат: 33
```

Рис. 2.12: Результат работы программы, вычисляющей выражение

3 Выводы

В результате выполнения работы были приобретены навыки написания программ с использованием аргументов командной строки.