Отчёт по лабораторной работе №8

По теме: Программирование цикла. Обработка аргументов командной строки.

Выполнил: Чубаев Кирилл Евгеньевич, НММбд-04-24.

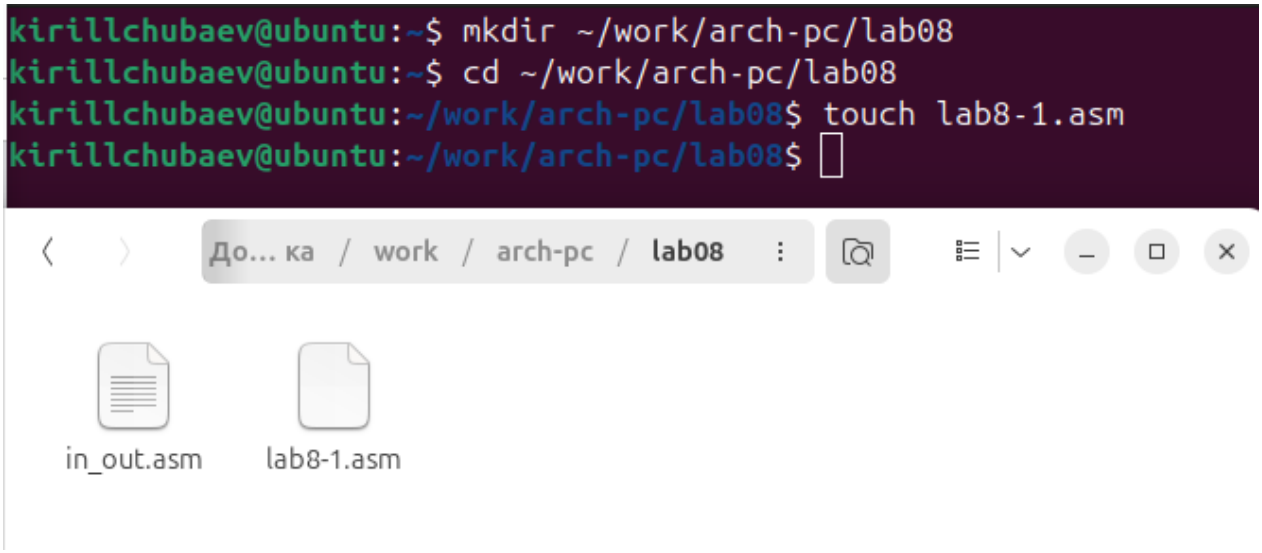
Содержание

Цель работы

Целью данной лабораторной работы является приобретение навыков написания программ с использованием циклов, а также обработки аргументов командной строки.

Ход выполнения лабораторной работы

1. Сначала я создал каталог lab08 и файл lab8-1.asm:



2. Далее я ввёл текст первой программы с помощью листинга 8.1. Потом создал исполняемый файл и запустил его:

```
%include 'in_out.asm'

SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:

mov eax,msg1
call sprint

mov ecx, N
mov edx, 10
call sread

mov eax,N
call atoi
mov [N],eax

mov ecx,[N]
label:
mov [N],ecx
mov eax,[N]
call iprintLF
loop label

call quit
```

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 35
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
```

3. Я изменил текст программы, в теле цикла label добавил строку “sub eax, 1”. Цикл закольцевался и стал бесконечным:

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

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 20
19
17
15
13
11
9
7
5
3
1
kirillchubaev@ubuntu:~/work/arch-pc/lab08$
```

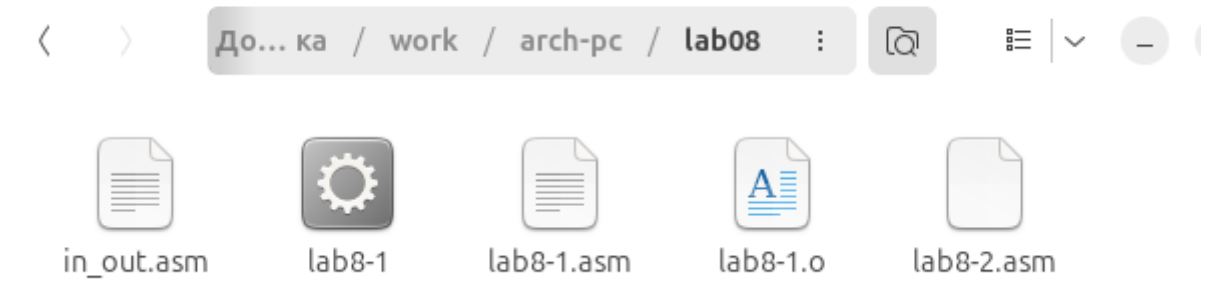
4. Далее изменил часть программы так, чтобы цикл и счетчик работал правильно. В итоге, после изменения кода программы, число проходки цикла стало соответствовать числу, введенному с клавиатуры:

```
mov ecx,[N]
label:
push ecx
sub ecx, 1
mov [N],ecx
mov eax,[N]
call iprintLF
pop ecx
loop label
```

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
kirillchubaev@ubuntu:~/work/arch-pc/lab08$
```

5. Я создал файл lab8-2.asm и с помощью листинга 8.2 написал код программы, которая выводит в терминал все введённые ранее аргументы:

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ touch lab8-2.asm
kirillchubaev@ubuntu:~/work/arch-pc/lab08$
```



```
%include 'in_out.asm'
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx, 1
next:
cmp ecx, 0
jz _end
pop eax
call sprintLF
loop next
_end:
call quit
```

6. Потом создал исполняемый файл и запустил его. В результате программа вывела все 3 аргумента, которые были введены, но в разной вариации:

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

7. Я создал файл lab8-3.asm. Ввел текст программы с помощью листинга 8.3. Я создал исполняемый файл и запустил ее. Программа вывела в терминал сумму чисел, которые я написал ранее:

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ touch lab8-3.asm
```

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

8. Далее изменил программу так, чтобы она выводила произведение введенных чисел. Затем я создал исполняемый файл и запустил его. В качестве проверки я ввел несколько комбинаций чисел. Программа работает корректно:

```
%include 'in_out.asm'

SECTION .data
msg DB "Результат: ",0

SECTION .text
GLOBAL _start

_start:

pop ecx

pop edx

sub ecx,1

mov esi,1
mov eax,1

next:
cmp ecx,0
jz _end

pop eax
call atoi
mov ebx,eax
mov eax,esi
mul ebx
mov esi,eax
loop next
```

```
_end:
mov eax,msg
call sprint
mov eax,esi
call iprintLF

call quit
```

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-3
Результат: 1
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-3 1 2 3 4
Результат: 24
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 54600
```

Выполнение самостоятельной работы

1. Для выполнения самостоятельной работы сначала я создал файл lab8-test.asm:

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ touch lab8-test.asm
```

2. Далее я написал программу, которая выводит сумму всех решений определённого выражения. В лабораторной работе №6 я получил 14 вариант, поэтому я написал код программы для функции f(x)=7*(x+1):

```
SECTION .text
GLOBAL _start

_start:
    pop ecx             ; Получаем количество аргументов (argc)
    pop edx             ; Пропускаем имя программы (argv[0])
    sub ecx, 1          ; Уменьшаем счетчик аргументов (не учитываем имя программы)

    mov esi, 0          ; Инициализируем сумму результатом 0

    mov eax, prim       ; Выводим строку с описанием функции
    call sprintLF

next:
    cmp ecx, 0          ; Если больше нет аргументов, завершаем цикл
    jz _end

    pop eax             ; Получаем следующий аргумент (в ASCII)
    call atoi           ; Преобразуем его в число

    add eax, 1          ; Вычисляем x + 1
    mov ebx, 7          ; Умножаем на 7
    mul ebx

    add esi, eax        ; Добавляем результат к общей сумме

    loop next
```

```
_end:
    mov eax, otv        ; Выводим строку с результатом
    call sprint

    mov eax, esi        ; Выводим итоговую сумму
    call iprintLF

    call quit
```

3. Затем я создал исполняемый файл и запустил программу. В качестве проверки работоспособности программы я ввел числа из примера (1, 2, 3, 4), а затем еще несколько комбинаций чисел. Программа работает исправно:

```
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ nasm -f elf lab8-test.asm
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-test lab8-test.o
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-test 1 2 3 4
f(x)=7*(x+1)
Результат: 98
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-test 1 2 3 4 5
f(x)=7*(x+1)
Результат: 140
kirillchubaev@ubuntu:~/work/arch-pc/lab08$ ./lab8-test 1 2 3 4 5 6
f(x)=7*(x+1)
Результат: 189
kirillchubaev@ubuntu:~/work/arch-pc/lab08$
```

Вывод

Во время выполнения данной лабораторной работы я приобрел полезные навыки написания программ с использованием цикла, а также обработки аргументов командной строки.

Список литературы

1. GDB: The GNU Project Debugger. — URL: https://www.gnu.org/software/gdb/.

2. GNU Bash Manual. — 2016. — URL: https://www.gnu.org/software/bash/manual/.

3. Midnight Commander Development Center. — 2021. — URL: https://midnight-commander.org/.

4. NASM Assembly Language Tutorials. — 2021. — URL: https://asmtutor.com/.

5. *Newham C.* Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658.

6. *Robbins A.* Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.

7. The NASM documentation. — 2021. — URL: https://www.nasm.us/docs.php.

8. *Zarrelli G.* Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.

9. *Колдаев В. Д., Лупин С. А.* Архитектура ЭВМ. — М. : Форум, 2018.

10. *Куляс О. Л., Никитин К. А.* Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.

11. *Новожилов О. П.* Архитектура ЭВМ и систем. — М. : Юрайт, 2016.

12. Расширенный ассемблер: NASM. — 2021. — URL: https://www.opennet.ru/docs/RUS/nasm/.

13. *Робачевский А., Немнюгин С., Стесик О.* Операционная система UNIX. — 2-е изд. — БХВ-Петербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.

14. *Столяров А.* Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.

15. *Таненбаум Э.* Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).

16. *Таненбаум Э.*, *Бос Х.* Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).