

# **Шаблон отчёта по лабораторной работе**

**Простейший вариант**

Лупупа Чилеще

# Содержание

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Цель работы</b>                           | <b>5</b>  |
| <b>2</b> | <b>Задание</b>                               | <b>6</b>  |
| 2.1      | Задание для самостоятельной работы . . . . . | 10        |
| <b>3</b> | <b>Выводы</b>                                | <b>12</b> |

## Список иллюстраций

|     |                                  |    |
|-----|----------------------------------|----|
| 2.1 | lab8-1.asm . . . . .             | 6  |
| 2.2 | листинг 8.1 . . . . .            | 7  |
| 2.3 | исполняемый файл . . . . .       | 7  |
| 2.4 | есх=есх-1' . . . . .             | 8  |
| 2.5 | есх=есх-1' . . . . .             | 8  |
| 2.5 | исполняемый файл . . . . .       | 9  |
| 2.6 | листинг 8.2 . . . . .            | 9  |
| 2.7 | lab8-3.asm . . . . .             | 10 |
| 2.8 | Самостоятельная работа . . . . . | 11 |
| 2.9 | Самостоятельная работа . . . . . | 11 |

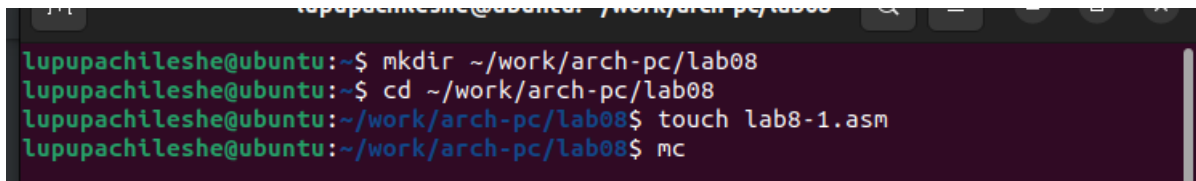
## Список таблиц

# 1 Цель работы

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

## 2 Задание

1. Создайте каталог для программ лабораторной работы № 8, перейдите в него и создайте файл lab8-1.asm:

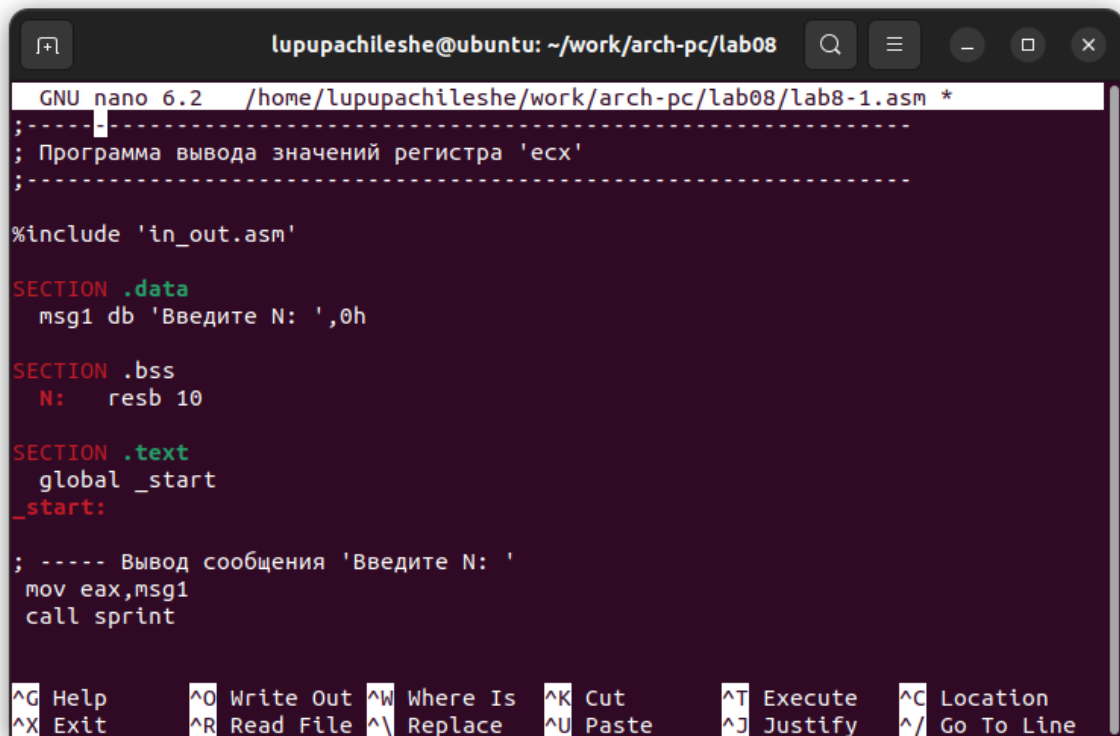


```
lupupachileshe@ubuntu:~$ mkdir ~/work/arch-pc/lab08
lupupachileshe@ubuntu:~$ cd ~/work/arch-pc/lab08
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ touch lab8-1.asm
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ mc
```

Рис. 2.1: lab8-1.asm

**комментарий:** Создал каталог для lab 8 и в нем создал файл lab8-1.asm.

2. Введите в файл lab8-1.asm текст программы из листинга 8.1. Создайте исполняемый файл и проверьте его работу.



```
GNU nano 6.2 /home/lupupachileshe/work/arch-pc/lab08/lab8-1.asm *
;-----
; Программа вывода значений регистра 'ecx'
;-----

#include 'in_out.asm'

SECTION .data
    msg1 db 'Введите N: ',0h

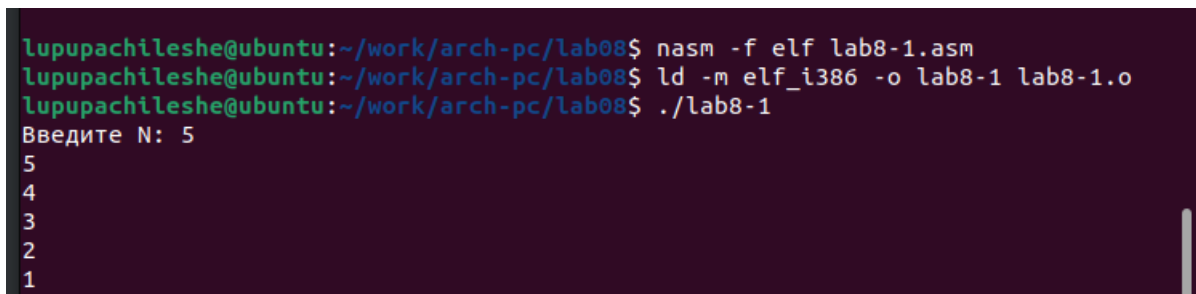
SECTION .bss
    N:    resb 10

SECTION .text
    global _start
_start:

; ----- Вывод сообщения 'Введите N: '
    mov eax,msg1
    call sprint

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

Рис. 2.2: листинг 8.1



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

Рис. 2.3: исполняемый файл

**комментарий:** Я внимательно изучил текст в листинге 8.1, вложил его в созданный файл и сделал исполняемый файл.

3. Измените текст программы добавив изменение значение регистра ecx в цикле:

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

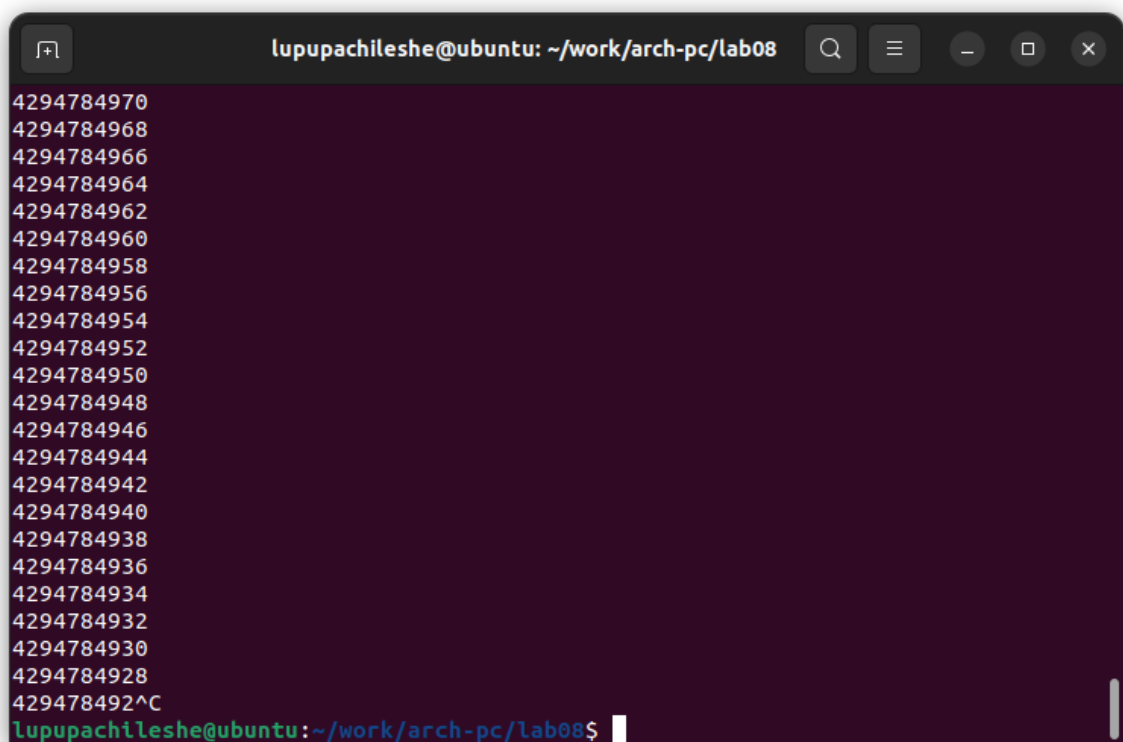
```

; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, `ecx=N`
Label:
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
loop label
call quit

```

Рис. 2.4:  $ecx=ecx-1$

4. Создайте исполняемый файл и проверьте его работу



The screenshot shows a terminal window titled 'lupupachileshe@ubuntu: ~/work/arch-pc/lab08'. The terminal output consists of a list of memory addresses, starting from 4294784970 and decreasing by 2 up to 4294784928, followed by a control character ^C. The prompt 'lupupachileshe@ubuntu:~/work/arch-pc/lab08\$' is visible at the bottom.

```

4294784970
4294784968
4294784966
4294784964
4294784962
4294784960
4294784958
4294784956
4294784954
4294784952
4294784950
4294784948
4294784946
4294784944
4294784942
4294784940
4294784938
4294784936
4294784934
4294784932
4294784930
4294784928
429478492^C
lupupachileshe@ubuntu:~/work/arch-pc/lab08$

```

Рис. 2.5:  $ecx=ecx-1$

**комментарий:** Я изменил текст программы, изменив в цикле значение регистра `ecx`

5. Внесите изменения в текст программы добавив команды `push` и `pop` (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла `loop`:



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

```
mov ecx,[N] ; Счетчик цикла, `ecx=N`  
label:  
push ecx  
sub ecx,1  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
loop label  
call _exit
```

**комментарий:** Я внес изменения в текст, включив команды push и pop для сохранения значения счетчика цикла.

6. Создайте исполняемый файл и проверьте его работу

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

Рис. 2.5: исполняемый файл

**комментарий:** создал исполняемый файл для файла после изменения текста

7. Создайте файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и введите в него текст программы из листинга 8.2. Создайте исполняемый файл и запустите его, указав аргументы:

user@dk4n31:~\$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'

```
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'  
аргумент1  
аргумент  
2  
аргумент 3  
lupupachileshe@ubuntu:~/work/arch-pc/lab08$
```

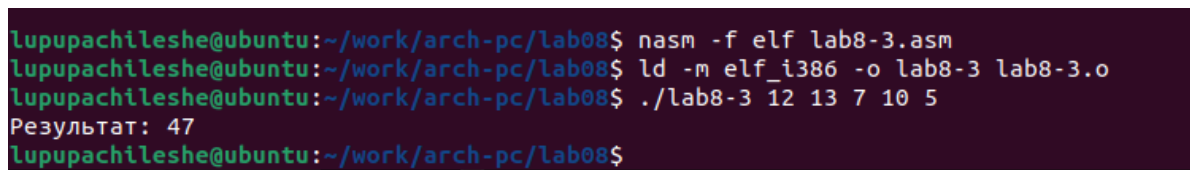
Рис. 2.6: листинг 8.2

**комментарий:** Создал файл с именем lab8-2.asm. Я внимательно изучил содержимое листинга 8.2 и поместил его в созданный файл. Далее я создал исполняемый файл

8. Создайте файл lab8-3.asm в каталоге ~/work/archpc/lab08 и введите в него текст программы из листинга 8.3.

Создайте исполняемый файл и запустите его, указав аргументы. Пример результата работы программы:

user@dk4n31:~\$ ./main 12 13 7 10 5 Результат: 47 user@dk4n31:~\$



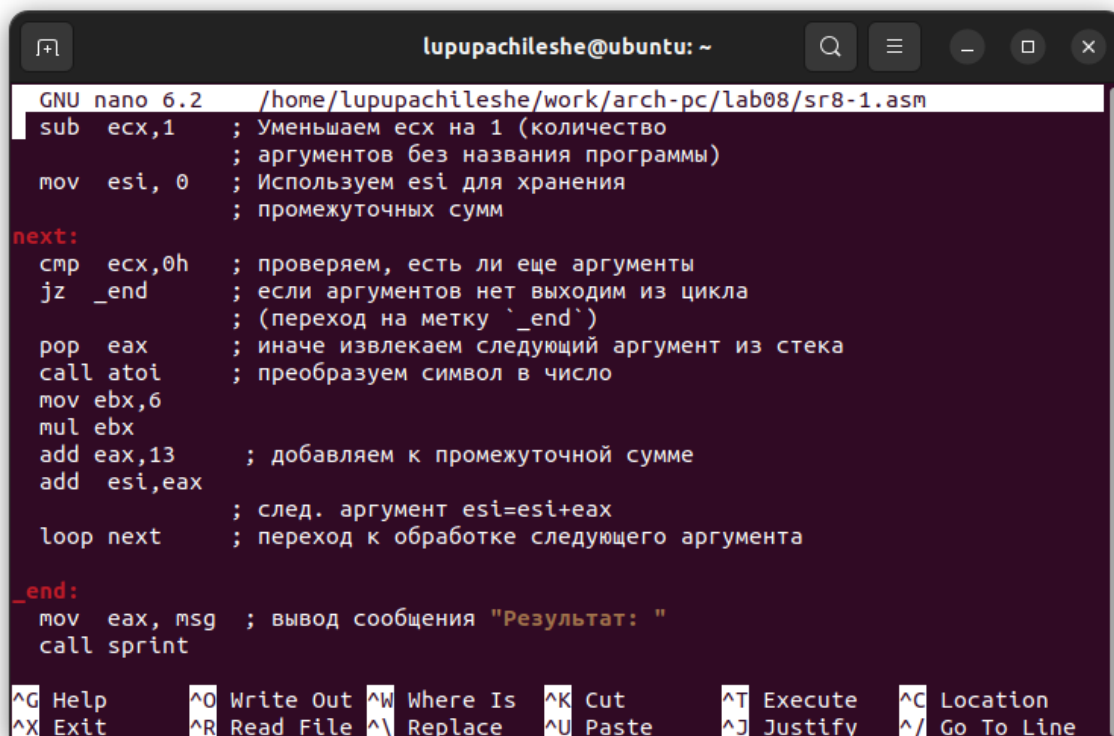
```
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
lupupachileshe@ubuntu:~/work/arch-pc/lab08$
```

Рис. 2.7: lab8-3.asm

**комментарий:** Я создал файл lab8-3.asm и поместил в него содержимое листинга 8.3. Далее я создал для созданного файла исполняемый файл

## 2.1 Задание для самостоятельной работы

Вариант 15:  $6\boxed{x} + 13$

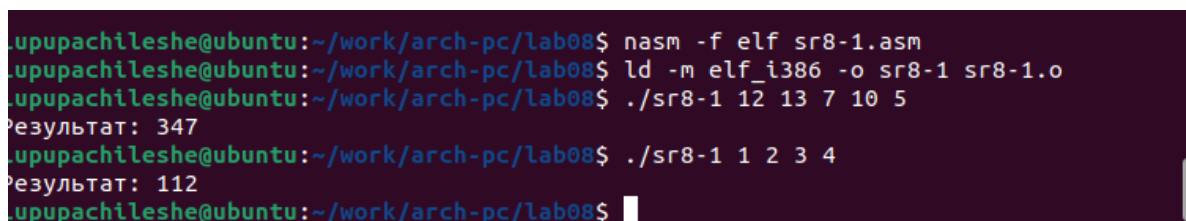


```
GNU nano 6.2 /home/lupupachileshe/work/arch-pc/lab08/sr8-1.asm
sub ecx,1 ; Уменьшаем ecx на 1 (количество
; аргументов без названия программы)
mov esi, 0 ; Используем esi для хранения
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку `_end`)
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mov ebx,6
mul ebx
add eax,13 ; добавляем к промежуточной сумме
add esi,eax
; след. аргумент esi=esi+eax
loop next ; переход к обработке следующего аргумента

_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line
```

Рис. 2.8: Самостоятельная работа



```
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ nasm -f elf sr8-1.asm
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ ld -m elf_i386 -o sr8-1 sr8-1.o
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ ./sr8-1 12 13 7 10 5
Результат: 347
lupupachileshe@ubuntu:~/work/arch-pc/lab08$ ./sr8-1 1 2 3 4
Результат: 112
lupupachileshe@ubuntu:~/work/arch-pc/lab08$
```

Рис. 2.9: Самостоятельная работа

**комментарий:** Я создал файл для самостоятельной работы. В нем я написал текст для ответа на 15 вопрос.

## 3 Выводы

Я научился писать программы с использованием циклов и обработки. аргументы командной строки.