

Отчёта по лабораторной работе №3

Команды безусловного и условного переходов в Nasm

Хохлачева Полина Дмитриевна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	15

Список иллюстраций

2.1	С помощью команды mkdir	6
2.2	Заполняем файл	6
2.3	Смотрим на работу файла	7
2.4	Редактируем файл	7
2.5	Смотрим на работу файла	7
2.6	Редактируем файл	8
2.7	Проверяем работу	8
2.8	С помощью команды touch	8
2.9	Заполняем файл	9
2.10	Смотрим на работу команды	9
2.11	Файл листинга	9
2.12	Изучаем файл	10
2.13	Удаляем операндум из файла	11
2.14	Транслируем файл	11
2.15	Изучаем файл с ошибкой	12
2.16	Новый файл	12
2.17	Пишем программу	13
2.18	Новый файл	13
2.19	Новый файл	14

Список таблиц

1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга

2 Выполнение лабораторной работы

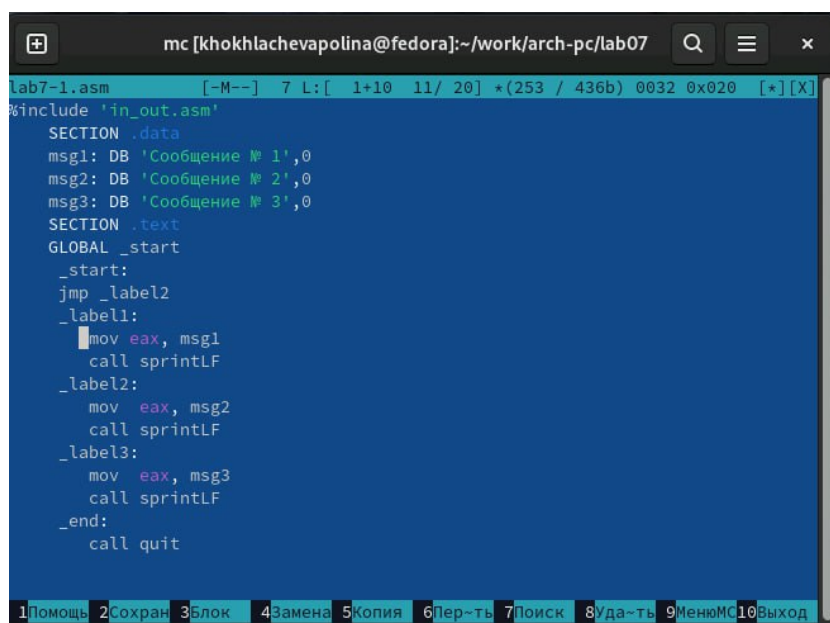
Создаём каталог и в нём создаём файл



```
khokhlachevapolina@fedora:~/work/arch-pc/lab07
khokhlachevapolina@fedora:~$ mkdir ~/work/arch-pc/lab07
khokhlachevapolina@fedora:~$ cd ~/work/arch-pc/lab07
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ touch lab7-1.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.1: С помощью команды mkdir

Открываем файл и заполняем его в соответствии с листингом



```
lab7-1.asm [-M--] 7 L:[ 1+10 11/ 20] *(253 / 436b) 0032 0x020 [*][X]
%include 'in_out.asm'
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1
call sprintLF
_label2:
mov eax, msg2
call sprintLF
_label3:
mov eax, msg3
call sprintLF
_end:
call quit
```

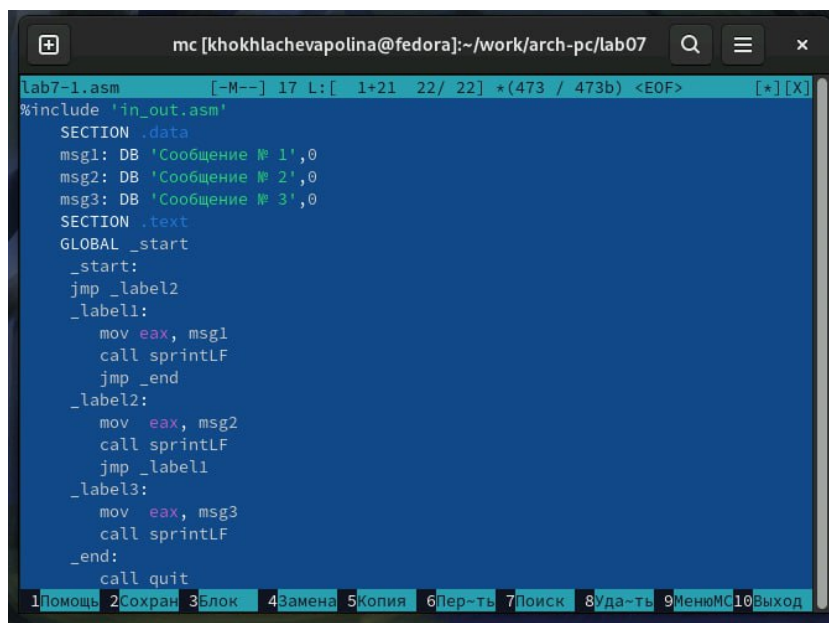
Рис. 2.2: Заполняем файл

Создаём файл и запускаем его

```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.3: Смотрим на работу файла

Открываем файл и редактируем его в соответствии с листингом



```
lab7-1.asm [-M--] 17 L: [ 1+21 22/ 22] *(473 / 473b) <EOF> [*] [X]
#include 'in_out.asm'
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1
call sprintf
jmp _end
_label2:
mov eax, msg2
call sprintf
jmp _label1
_label3:
mov eax, msg3
call sprintf
_end:
call quit
```

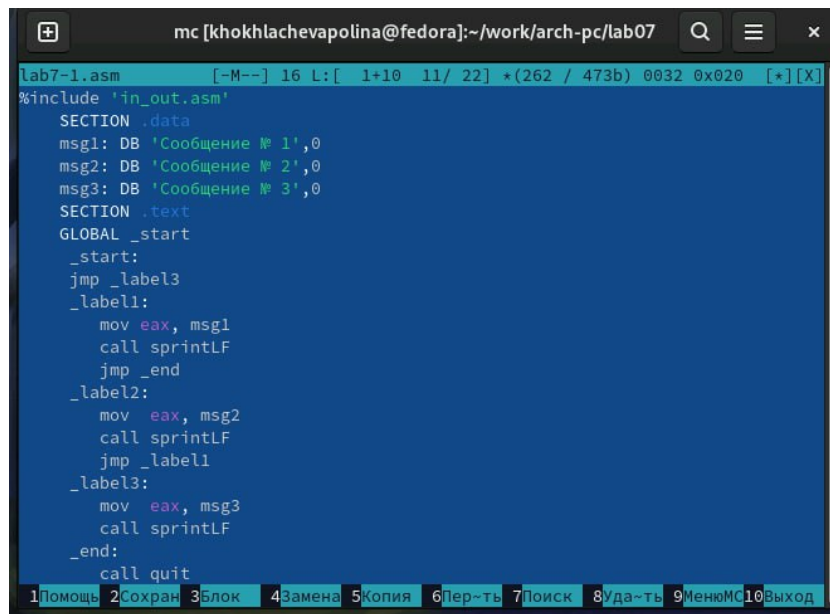
Рис. 2.4: Редактируем файл

Создаём файл и запускаем его

```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.5: Смотрим на работу файла

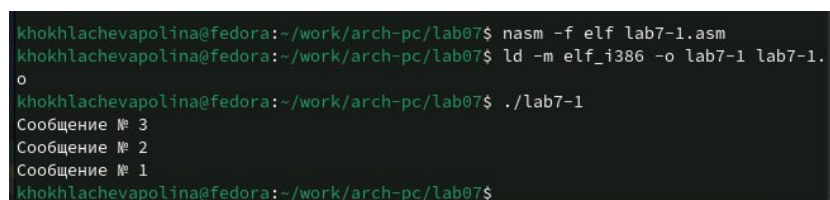
Открываем файл и редактируем его



```
lab7-1.asm [-M--] 16 L: [ 1+10 11/ 22] +(262 / 473b) 0032 0x020 [*][X]
#include 'in_out.asm'
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1
call sprintf
jmp _end
_label2:
mov eax, msg2
call sprintf
jmp _label1
_label3:
mov eax, msg3
call sprintf
_end:
call quit
```

Рис. 2.6: Редактируем файл

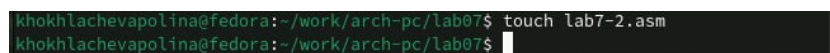
Создаём файл и запускаем его



```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.7: Проверяем работу

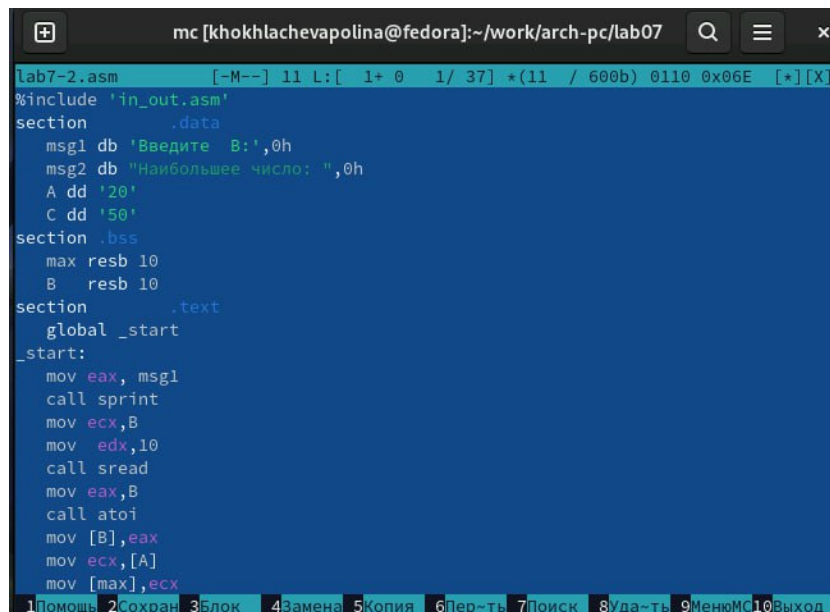
Создаём новый файл



```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ touch lab7-2.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.8: С помощью команды touch

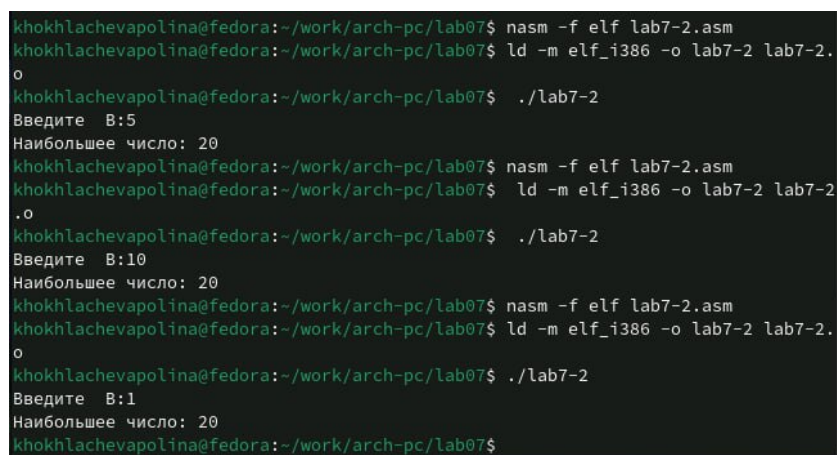
Открываем файл и заполняем в соответствии с листингом



```
lab7-2.asm [-M--] 11 L: [ 1+ 0 1/ 37] *(11 / 600b) 0110 0x06E [*][X]
#include 'in_out.asm'
section .data
    msg1 db 'Введите B:',0h
    msg2 db "Наибольшее число: ",0h
    A dd '20'
    C dd '50'
section .bss
    max resb 10
    B resb 10
section .text
    global _start
_start:
    mov eax, msg1
    call sprint
    mov ecx, B
    mov edx, 10
    call sread
    mov eax, B
    call atoi
    mov [B], eax
    mov ecx, [A]
    mov [max], ecx
```

Рис. 2.9: Заполняем файл

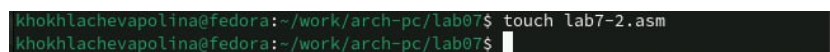
Создаём файл проверяем работу



```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B:5
Наибольшее число: 20
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B:10
Наибольшее число: 20
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B:1
Наибольшее число: 20
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.10: Смотрим на работу команды

Создаём файл



```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ touch lab7-2.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.11: Файл листинга

Открываем файл и изучаем его

```

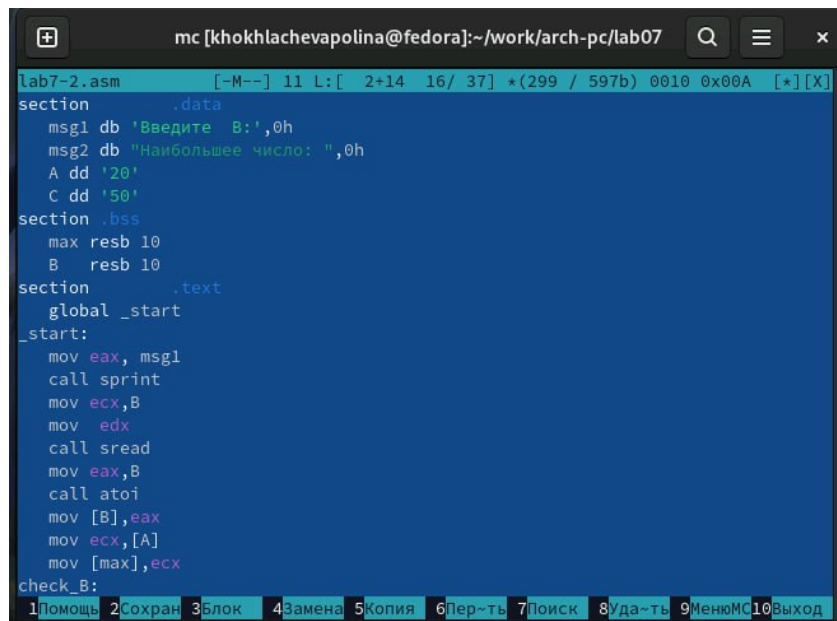
khokhlachevapolina@fedora:~/work/arch-pc/lab07
lab7-2.lst  [----]  0 L: [ 1+ 0 1/213] *(0 /12835b) 0032 0x020 [*][X]
1          %include 'in_out.asm'
1          <1> ;----- slen -----
2          <1> ; Функция вычисления длины сообщения
3          <1> slen:.....
4 00000000 53          <1> push ebx.....
5 00000001 89C3        <1> mov ebx, eax.....
6          <1> .....
7          <1> nextchar:.....
8 00000003 803800      <1> cmp byte [eax], 0...
9 00000006 7403        <1> jz finished.....
10 00000008 40          <1> inc eax.....
11 00000009 EBF8       <1> jmp nextchar.....
12          <1> .....
13          <1> finished:
14 0000000B 29D8       <1> sub eax, ebx
15 0000000D 5B          <1> pop ebx.....
16 0000000E C3         <1> ret.....
17          <1> .
18          <1> .
19          <1> ;----- sprint -----
20          <1> ; Функция печати сообщения
21          <1> ; входные данные: mov eax,<message>
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход

```

Рис. 2.12: Изучаем файл

Строка 33: 0000001D-адрес в сегменте кода, B1000000-машинный код, mov ebx, 1-присвоение переменной ebx 1. Строка 34: 00000022-адрес в сегменте кода, B804000000-машинный код, mov eax, 4-присвоение переменной eax значение 4. Строка 3500000027-адрес в сегменте кода, CD80-машинный код, int 80h-вызов ядра

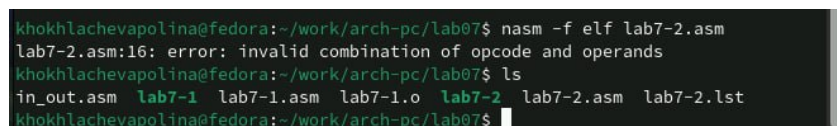
Открываем файл и удаляем один операндум



```
lab7-2.asm [-M--] 11 L: [ 2+14 16/ 37] *(299 / 597b) 0010 0x00A [*][X]
section .data
    msg1 db 'Введите B:',0h
    msg2 db "Наибольшее число: ",0h
    A dd '20'
    C dd '50'
section .bss
    max resb 10
    B resb 10
section .text
    global _start
_start:
    mov eax, msg1
    call sprint
    mov ecx, B
    mov edx,
    call sread
    mov eax, B
    call atoi
    mov [B], eax
    mov ecx, [A]
    mov [max], ecx
check_B:
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход
```

Рис. 2.13: Удаляем операндум из файла

Транслируем с получение файла листинга



```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
lab7-2.asm:16: error: invalid combination of opcode and operands
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ ls
in_out.asm lab7-1 lab7-1.asm lab7-1.o lab7-2 lab7-2.asm lab7-2.lst
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.14: Транслируем файл

Выдаётся ошибка, но создаются исполнительные файлы lab7-2 и lab7-2.lst

Открываем файл и изучаем его

```

khokhlachevapolina@fedora:~/work/arch-pc/lab07
lab7-2.lst [----] 55 L:[ 26+10 36/213] *(2260/12835b) 0010 0x00A [*][X]
25 00000011 53 <1> push ebx
26 00000012 50 <1> push eax
27 00000013 E8E8FFFFFF <1> call slen
28 <1> .....
29 00000018 89C2 <1> mov edx, eax
30 0000001A 58 <1> pop eax
31 <1> .....
32 0000001B 89C1 <1> mov ecx, eax
33 0000001D BB01000000 <1> mov ebx, 1
34 00000022 B804000000 <1> mov eax, 4
35 00000027 CD80 <1> int 80h
36 <1> .
37 00000029 5B <1> pop ebx
38 0000002A 59 <1> pop ecx
39 0000002B 5A <1> pop edx
40 0000002C C3 <1> ret
41 <1> .
42 <1> .
43 <1> ;----- sprintLF -----
44 <1> ; Функция печати сообщения с переводом с
45 <1> ; входные данные: mov eax,<message>
46 <1> sprintLF:
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Вда-ть 9МенюMC10Выход

```

Рис. 2.15: Изучаем файл с ошибкой

#Задание для самостоятельной работы

Напишите программу нахождения наименьшей из 3 целочисленных переменных a , b и c . Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу

Создаём новый файл

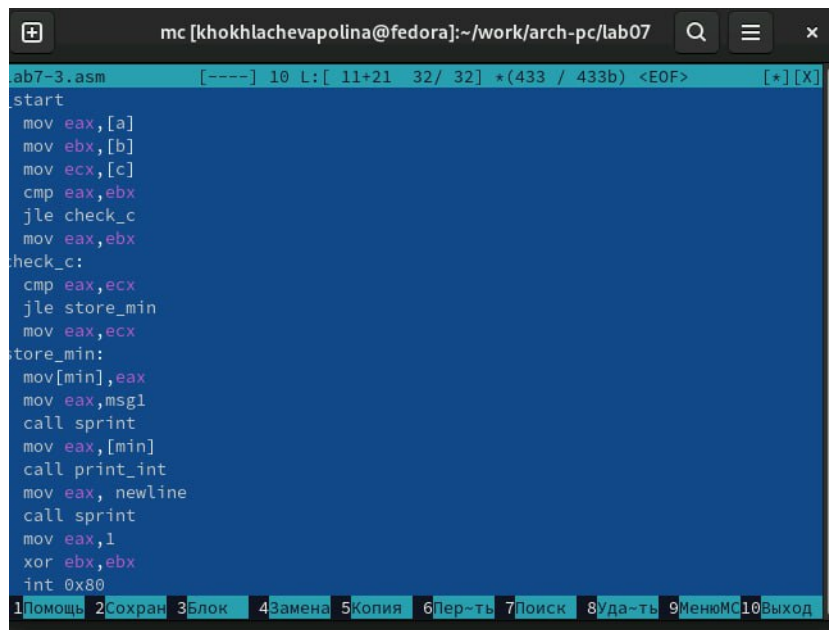
```

khokhlachevapolina@fedora:~/work/arch-pc/lab07$ touch lab7-3.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$

```

Рис. 2.16: Новый файл

Открываем файл и пишем программу для решения задачи

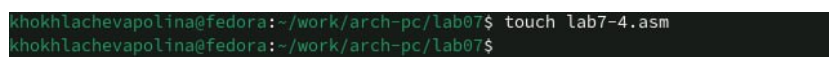


```
ab7-3.asm [----] 10 L: [ 11+21 32/ 32] *(433 / 433b) <EOF> [*][X]
start
mov eax,[a]
mov ebx,[b]
mov ecx,[c]
cmp eax,ebx
jle check_c
mov eax,ebx
check_c:
cmp eax,ecx
jle store_min
mov eax,ecx
store_min:
mov [min],eax
mov eax,msg1
call sprint
mov eax,[min]
call print_int
mov eax,newline
call sprint
mov eax,1
xor ebx,ebx
int 0x80
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюМС10Выход
```

Рис. 2.17: Пишем программу

Напишите программу, которая для введенных с клавиатуры значений a и b вычисляет значение заданной функции $f(a, b)$ и выводит результат вычислений. Вид функции $f(a, b)$ выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений a и b из 7.6

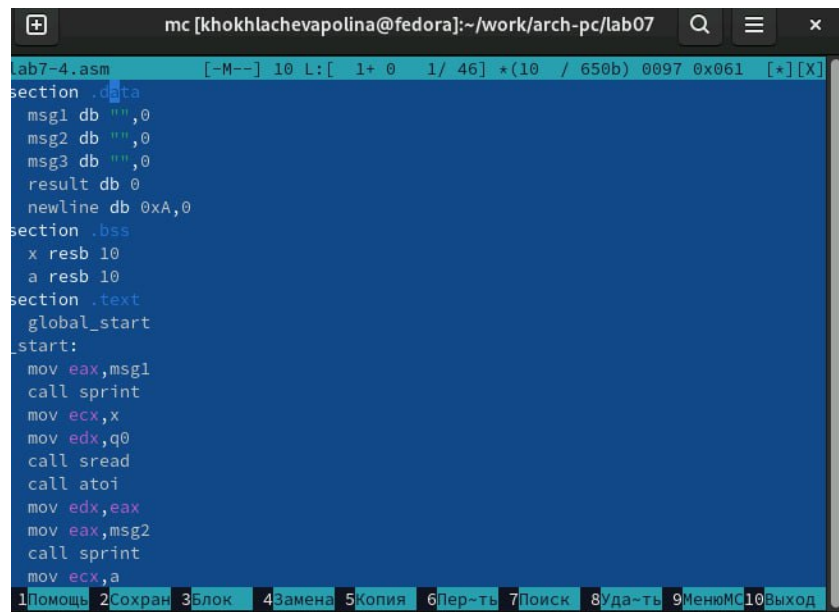
Создаём новый файл



```
khokhlachevapolina@fedora:~/work/arch-pc/lab07$ touch lab7-4.asm
khokhlachevapolina@fedora:~/work/arch-pc/lab07$
```

Рис. 2.18: Новый файл

Открываем его и пишем программу для решения задачи



```
lab7-4.asm [-M--] 10 L:[ 1+ 0 1/ 46] *(10 / 650b) 0097 0x061 [*][X]
section .data
msg1 db "",0
msg2 db "",0
msg3 db "",0
result db 0
newline db 0xA,0
section .bss
x resb 10
a resb 10
section .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx,x
mov edx,q0
call sread
call atoi
mov edx,eax
mov eax,msg2
call sprint
mov ecx,a
```

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

Рис. 2.19: Новый файл

3 Выводы

Мы познакомились с структурой файла листинга, освоили условного и безусловного перехода.