РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

дисциплина: Архитектура компьютера

Студент: Макаренко Александра Александровна

1132250434

Группа:НКАбд-07-25

МОСКВА

2025г.

Содержание

1 Цель работы

Цель данной лабораторной работы - освоение арифметческих инструкций языка ассемблера NASM.

2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. - Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: mov ax,bx. - Непосредственная адресация – значение операнда задается непосредственно в команде, Например: mov ax,2. - Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется

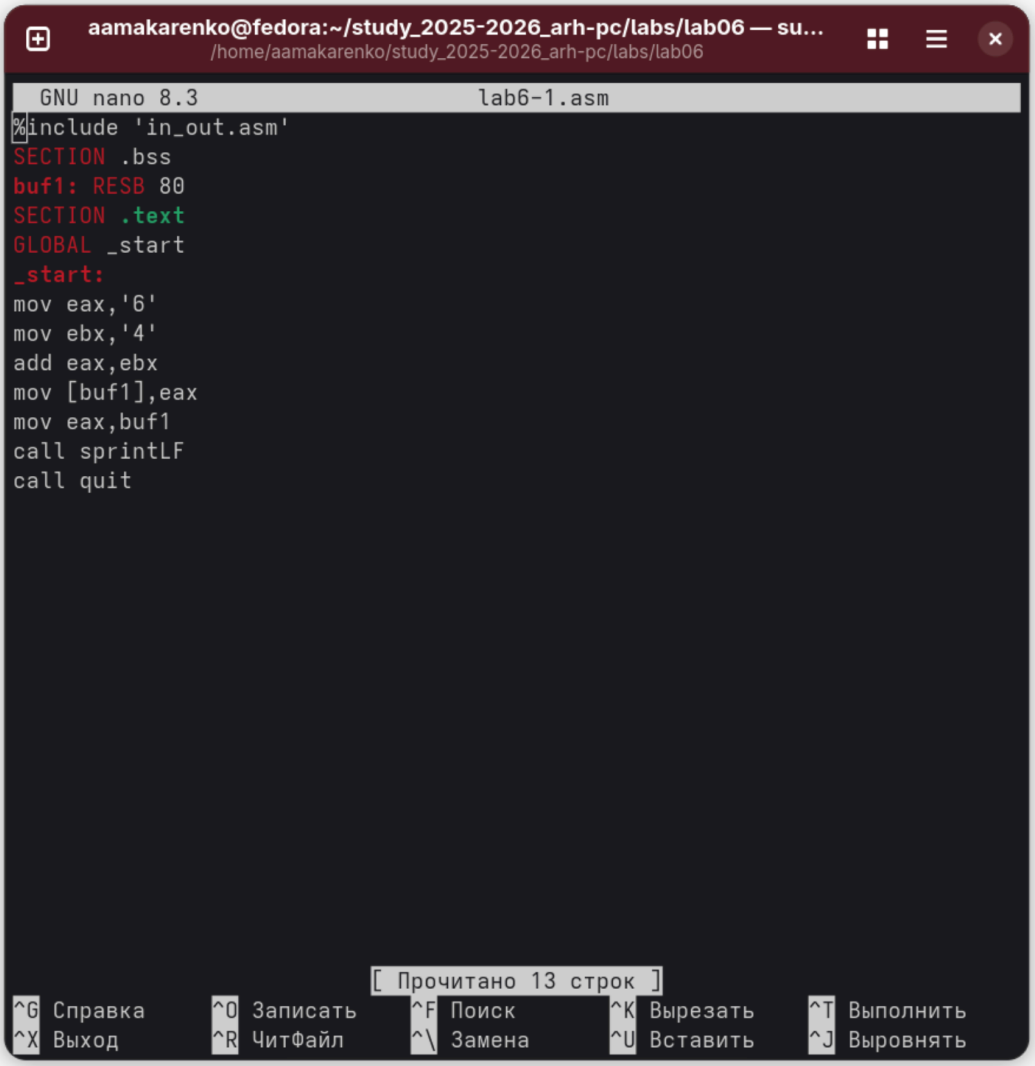
выполнить операцию. Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от American Standard Code for Information Interchange (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). Поэтому, например, чтобы вывести число, надо предварительно преобразовать его цифры в ASCII-коды этих цифр и выводить на экран эти коды, а не само число. Если же выводить число на экран непосредственно, то экран воспримет его не как число, а как последовательность ASCII-символов – каждый байт числа будет воспринят как один ASCII-символ – и выведет на экран эти символы. Аналогичная ситуация происходит и при вводе данных с клавиатуры. Введенные данные будут представлять собой символы, что сделает невозможным получение корректного результата при выполнении над ними арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно.

4 Выполнение лабораторной работы

4.1 Символьные и численные данные в NASM

Создаю каталог для программ лабораторной работы №6 и перехожу в него, создаю там файл

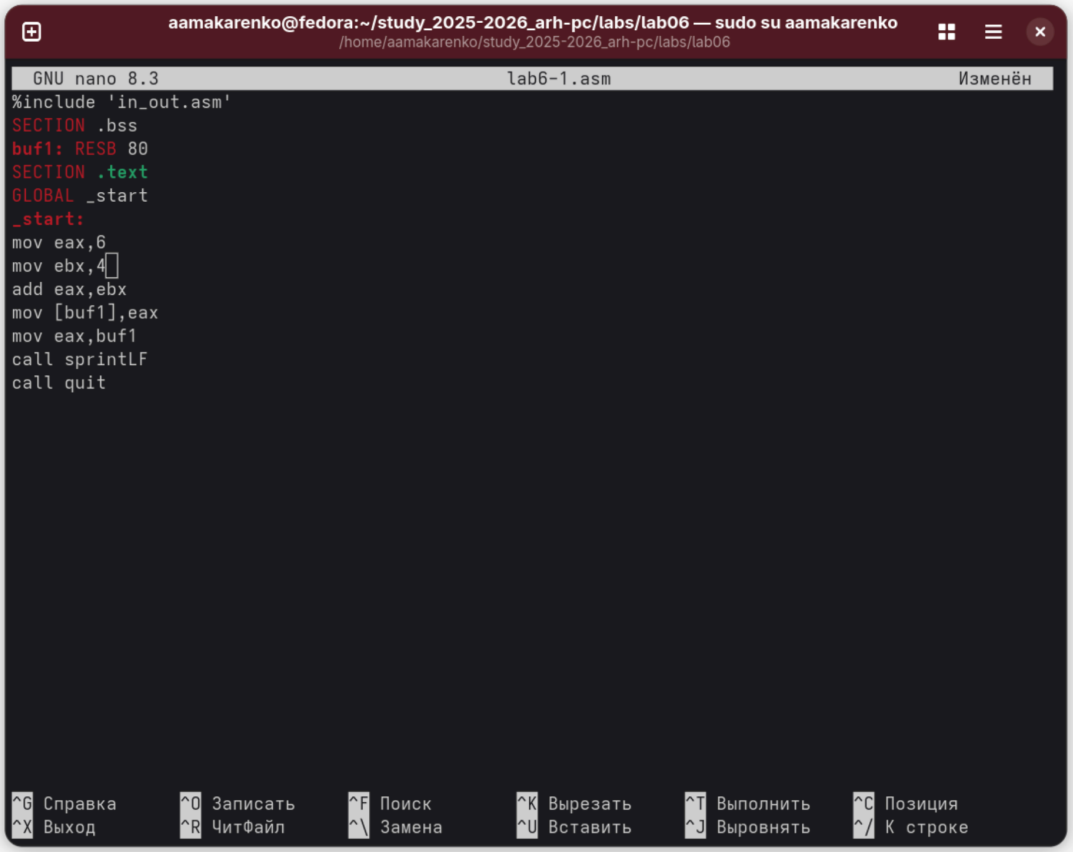
В созданном файле ввожу программу из листинга



Создаю исполняемый файл и запускаю его, вывод программы отличается от предполагаемого изначально, ибо коды символов в сумме дают символ j по таблице ASCII.

```
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ nano lab6-1.asm
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ nasm -f elf lab6-1.asm
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ./lab6-1
j
```

Изменяю текст изначальной программы, убрав кавычки



На этот раз программа выдала пустую строчку, это связано с тем, что символ 10 означает переход на новую строку

Создаю новый файл для будущей программы и записываю в нее код из листинга

```
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ touch lab6-2.asm
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$
```

```
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ./lab6-2
106
```

```
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ./lab6-2
106
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ cat lab6-2.asm
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ nano lab6-2.asm
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ./lab6-2
106
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ cat lab6-2.asm
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, 6
mov ebx, 4
add eax,ebx
call iprintLF
call quit
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$
```

Создаю исполняемый файл и запускаю его, теперь отображается результат 106, программа, как и в первый раз, сложила коды символов, но вывела само число, а не его символ, благодаря замене функции вывода на iprintLF

Убрав кавычки в программе, я снова ее запускаю и получаю предполагаемый 12 изначально результат.

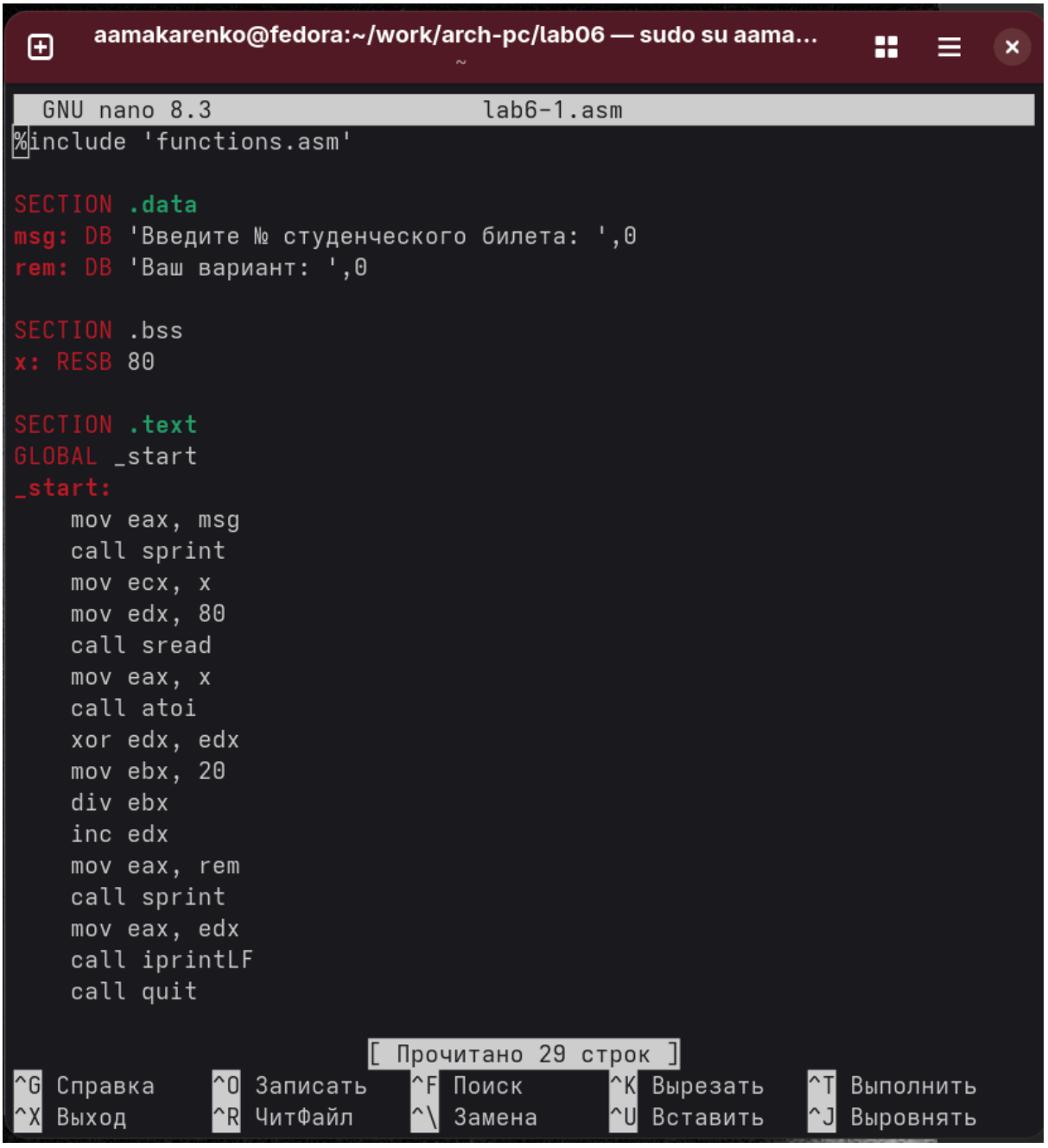
4.2 Выполнение арифметических операций в NASM Создаю новый файл и копирую в него содержимое листинга

Создаю новый файл и копирую в него содержимое листинга

```
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ cat lab6-3.asm
%include 'in_out.asm'
SECTION .data
div: DB 'Результат: ', 0
rem: DB 'Остаток от деления: ', 0
SECTION .text
GLOBAL _start
_start:
mov eax, 4
mov ebx, 6
mul ebx
add eax, 2
xor edx, edx
mov ebx, 5
div ebx
mov edi, eax
mov eax, div
call sprint
mov eax, edi
call iprintLF
mov eax, rem
call sprint
mov eax, edx
call iprintLF
call quit
```

```
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ nasm -f elf32 lab6-3.asm -o lab6-3.o
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ld -m elf_i386 lab6-3.o -o lab6-3
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$
```

Создаю новый файл и помещаю текст из листинг




```
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ nano lab6-31.asm
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ nasm -f elf32 lab6-31.asm -o lab6-31.o
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ld -m elf_i386 lab6-31.o -o lab6-31
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$ ./lab6-31
Введите № студенческого билета:
1132250434
Ваш вариант: 15
aamakarenko@fedora:~/study_2025-2026_arh-pc/labs/lab06$
```

4.3 Ответы на контрольные вопросы

1. За вывод сообщения “Ваш вариант” отвечают строки кода: mov eax,rem call sprint

2. Инструкция mov ecx, x используется, чтобы положить адрес вводимой строки x в регистр ecx mov edx, 80 - запись в регистр edx длины вводимой строки call sread - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры.

3. call atoi используется для вызова подпрограммы из внешнего файла, которая преобразует ascii-код символа в целое число и записывает результат в регистр eax.

4. За вычисления варианта отвечают строки:
xor edx,edx ; обнуление edx для корректной работы
div mov ebx,20 ;
ebx = 20 div ebx ;
eax = eax/20, edx - остаток от деления
inc edx ; edx = edx 1

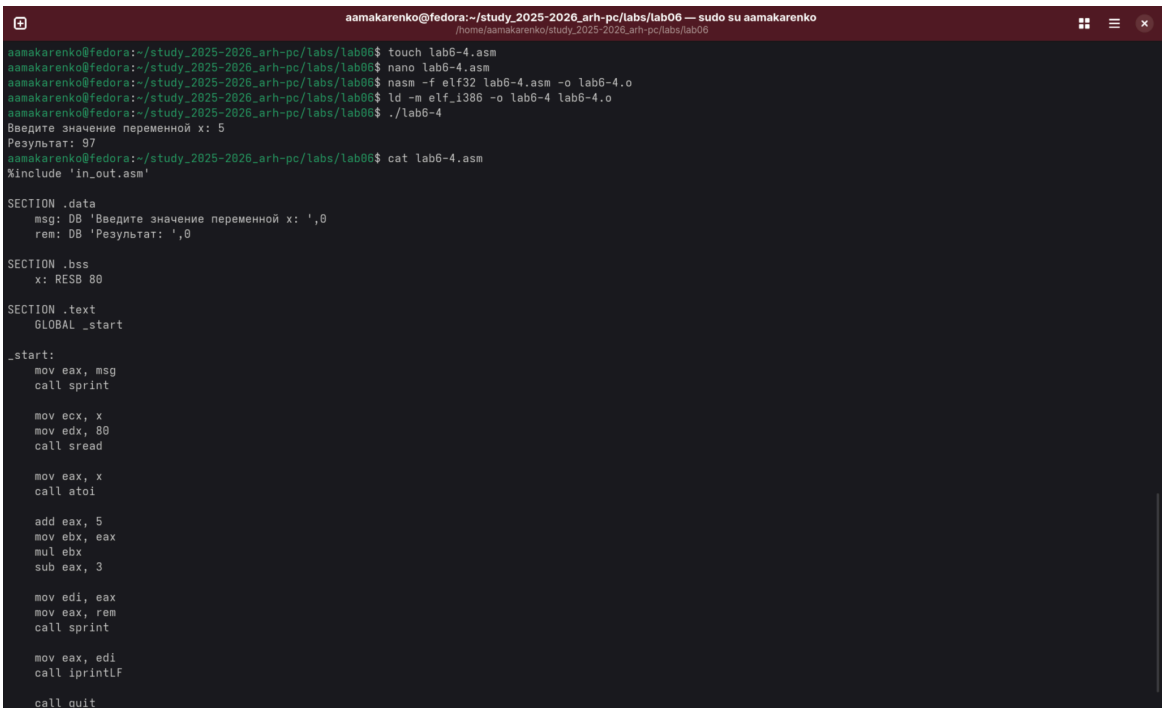
5. При выполнении инструкции div ebx остаток от деления записывается в регистр edx.

6. Инструкция inc edx увеличивает значение регистра edx на 1.

7. За вывод на экран результатов вычислений отвечают строки:
mov eax,edx call iprintLF

4.4 Задание для самостоятельной работы

В соответсвии с выбранным вариантом, я реализую программу для подсчета функции f(x) = (5 + x)^2 − 3, проверка на нескольких переменных показывает корректное выполнение программы



5 Выводы

При выполнении данной лабораторной работы я освоил арифметические инструкции языка ассемблера NASM.

6 Список литературы

1. Пример выполнения лабораторной работы
2. Курс на ТУИС
3. Лабораторная работа №6
4. Программирование на языке ассемблера NASM Столяров А. В.