РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

дисциплина: Архитектура компьютера

Студентка: Скворцова Анастасия

Группа: НБИбд-03-24

МОСКВА 2024 г

Содержание

6. **Цель работы**

Цель данной лабораторной работы - освоение арифметических 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: movax,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 6 сделает невозможным получение корректного результата при выполнении над ними арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно.

3. **Выполнение лабораторной работы**

4.1 Символьные и численные данные в NASM

1) Создаю директорию, в которой буду создавать файлы с программами для лабораторной работы №6 (рис. 1). Перехожу в созданный каталог с помощью утилиты cd и создаю файл lab6-1.asm

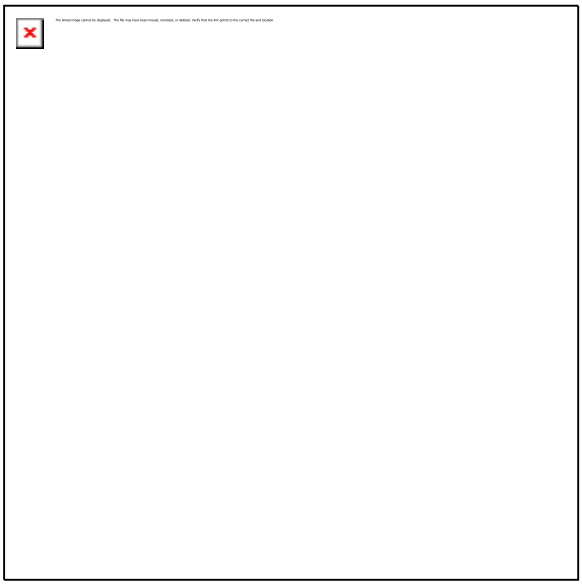

Рис. 1: создание директории и файла

2) Копирую в текущий каталог файл in_out.asm с помощью утилиты cp, т.к. он будет использоваться в других программах (рис. 2)

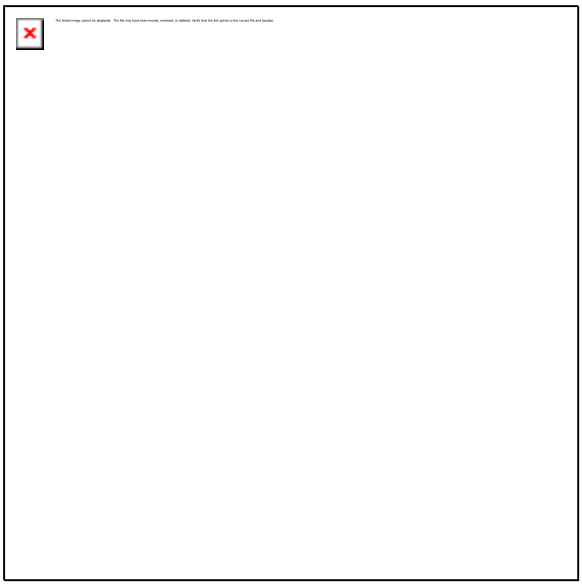

Рис. 2: создание копии файла

3) Открываю созданный файл lab6-1.asm, вставляю в него программу вывода значения регистра eax (рис. 3)

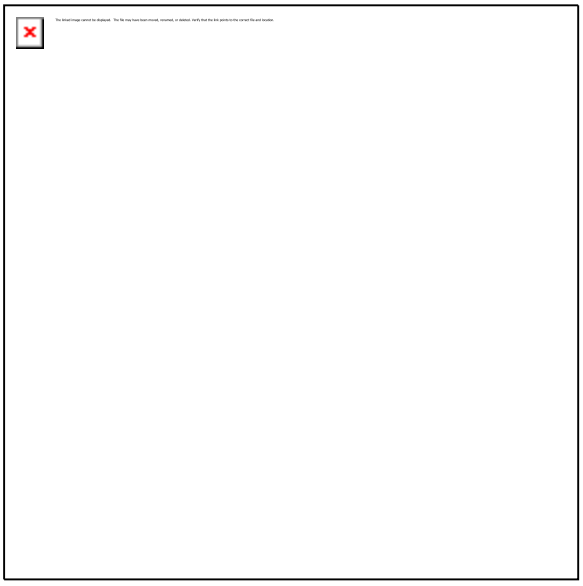

Рис. 3: редактирование файла

1) Создаю исполняемый файл программы и запускаю его (рис. 4). Вывод

   программы: символ j, потому что программа вывела символ, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6.

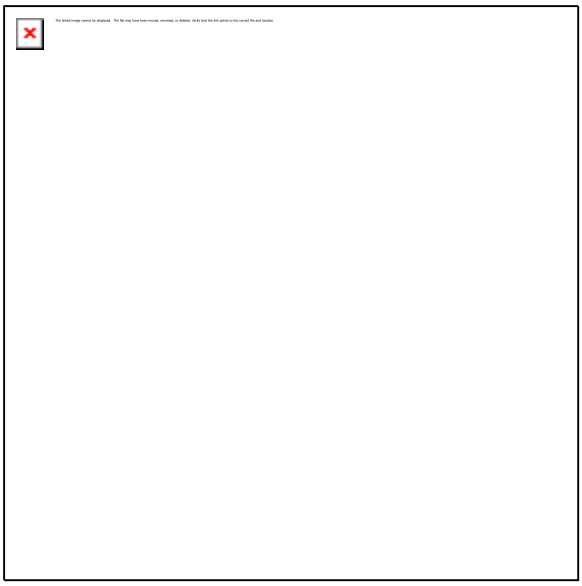

Рис. 4: запуск исполняемого файла

2) Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4 (рис. 5)

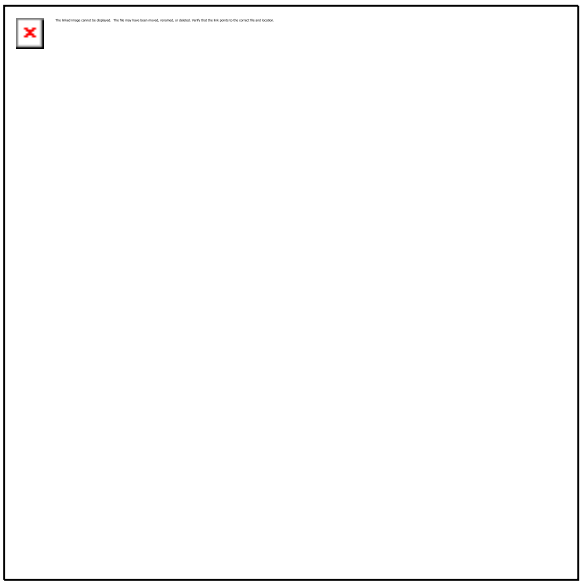

Рис. 5: редактирование файла

3) Создаю новый исполняемый файл программы и запускаю его (рис. 6). Теперь вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран (рис. 6)

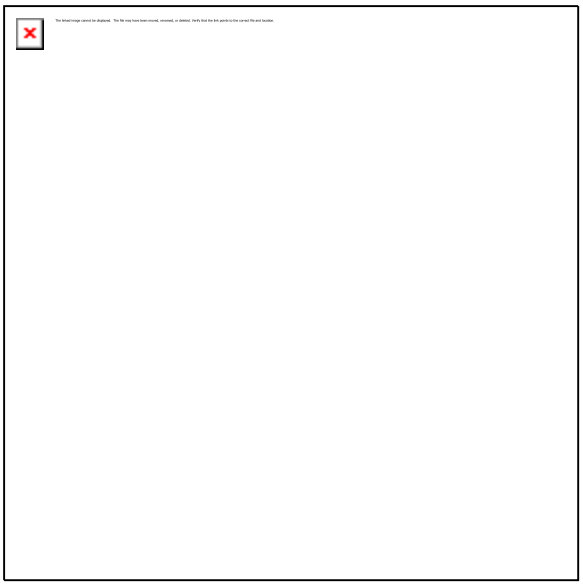

Рис. 6: запуск исполняемого файла

1) Создаю новый файл lab6-2.asm с помощью утилиты touch (рис. 7)

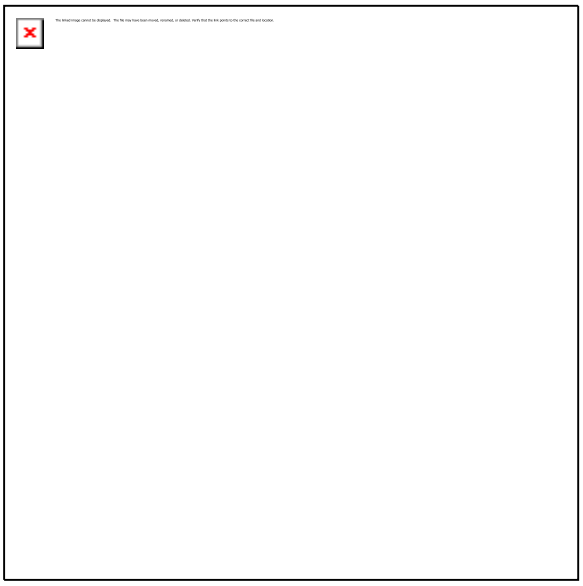

Рис. 7: создание файла

1) Ввожу в файл текст другой программы для вывода значения регистра eax (рис. 8)

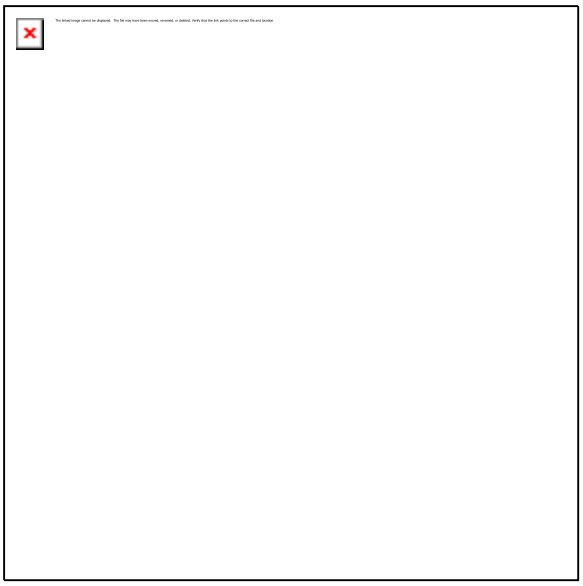

Рис. 8: редактирование файла

1) Создаю и запускаю исполняемый файл lab6-2 (рис. 9). Теперь вывод число 106, потому что программа позволяет вывести именно число, а не символ, хотя все еще происходит именно сложение кодов символов “6” и “4”.

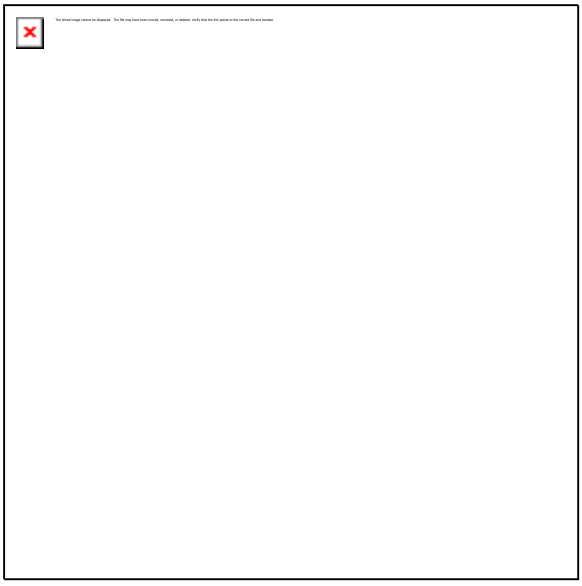

Рис. 9: запуск исполняемого файла

2) Заменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4 (рис. 10)

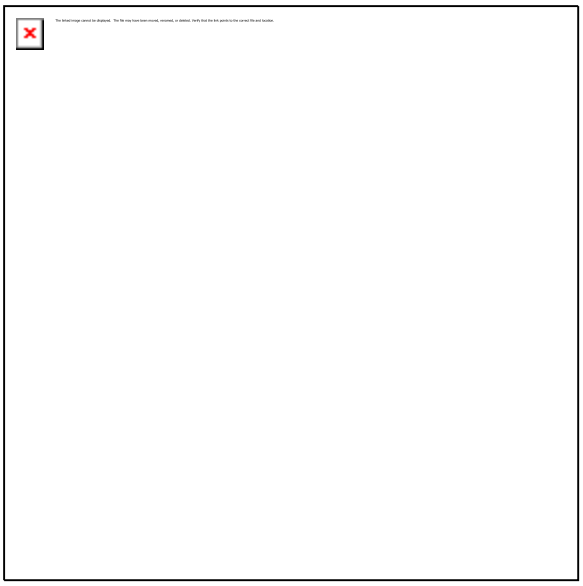

Рис. 10: редактирование файла

1) Создаю и запускаю новый исполняемый файл (рис. 11). Теперь программа складывает не соответствующие символам коды в системе ASCII, а сами числа, поэтому вывод 10.

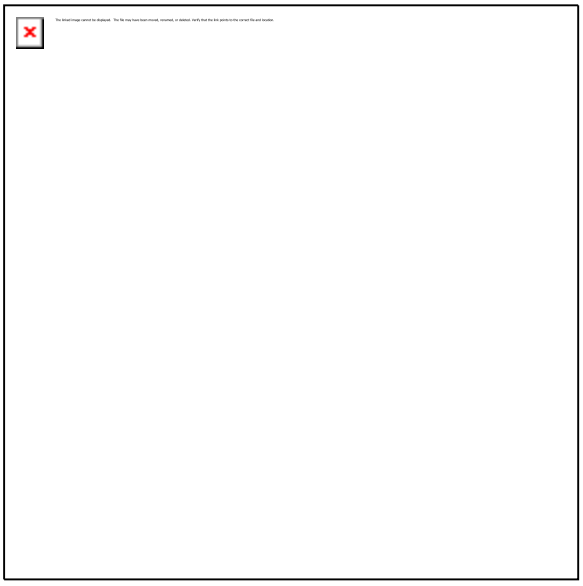

Рис. 11: запуск исполняемого файла

1) Заменяю в тексте программы функцию iprintLF на iprint (рис. 12)

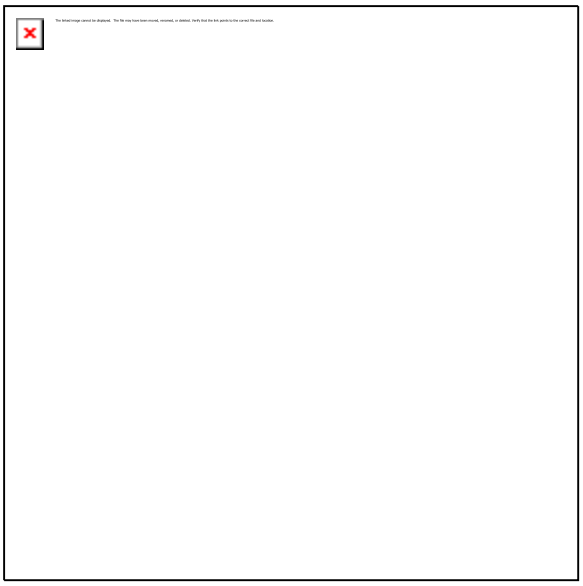

Рис. 12: редактирование файла

1) Создаю и запускаю новый исполняемый файл (рис.13). Вывод не изменился, потому что символ переноса строки не отображался, когда программа исполнялась с функцией iprintLF, а iprint не добавляет к выводу символ переноса строки, в отличие от iprintLF.

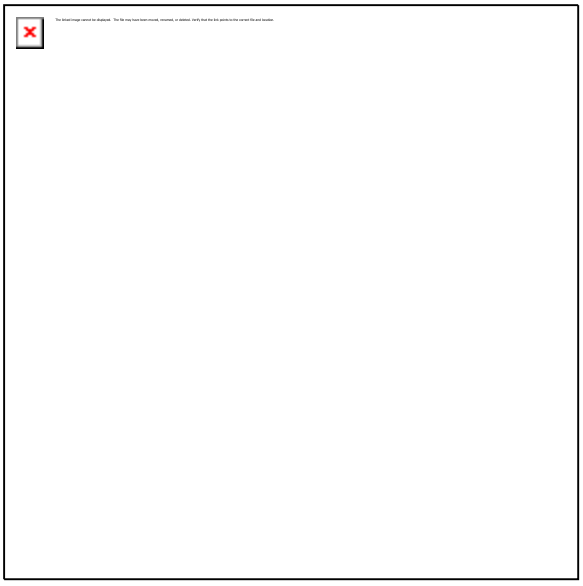

Рис. 13: запуск исполняемого файла

2. **Выполнение арифметических операций в NASM**

1) Создаю файлlab6-3.asm с помощью утилиты touch (рис. 14)

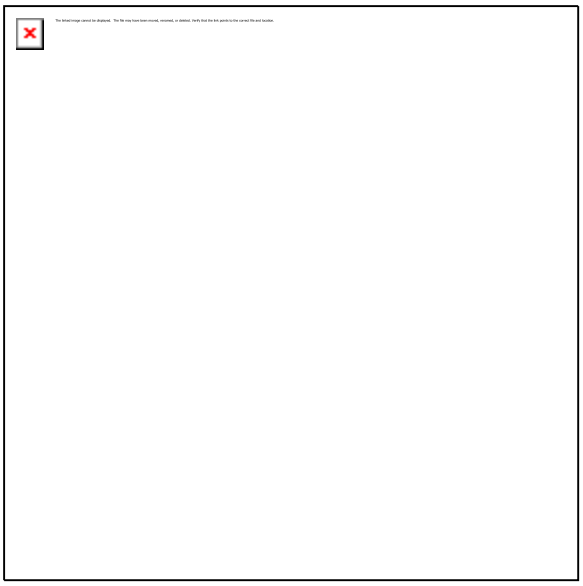

Рис. 14: создание файла

1) Ввожу в созданный файл текст программы для вычисления значения выражения f(x) = (5 * 2 + 3)/3 (рис. 15)

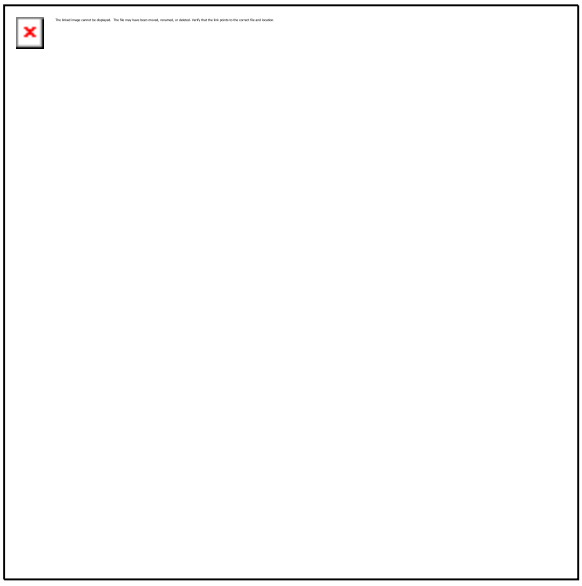

Рис. 15: редактирование файла

2) Создаю исполняемый файл и запускаю его (рис. 16)

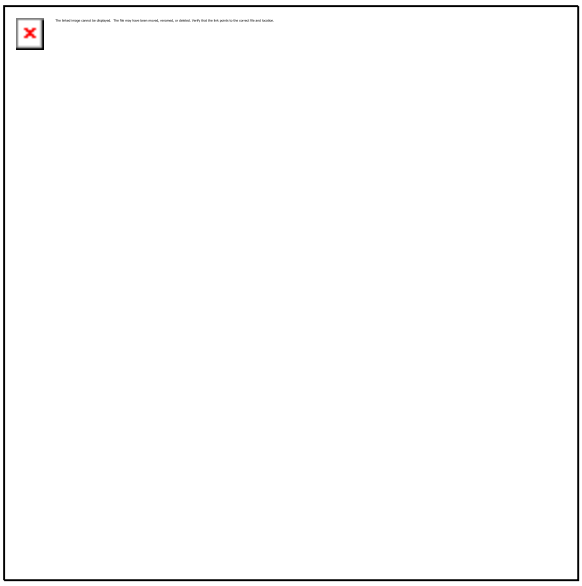

Рис. 16: запуск исполняемого файла

3) Изменяю программу так, чтобы она вычисляла значение выражения

f(x) = (4 * 6+2)/5 (рис. 17)

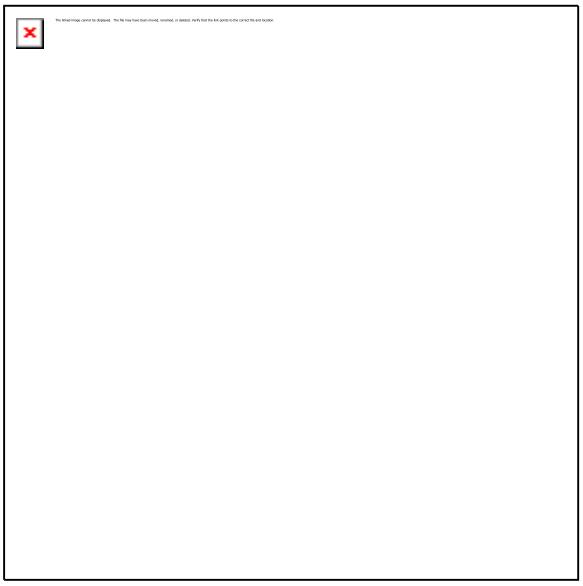

Рис. 17: изменение программы

4) Создаю и запускаю новый исполняемый файл (рис. 18). Я посчитала для проверки

правильности работы программы значение выражения самостоятельно, программа отработала верно.

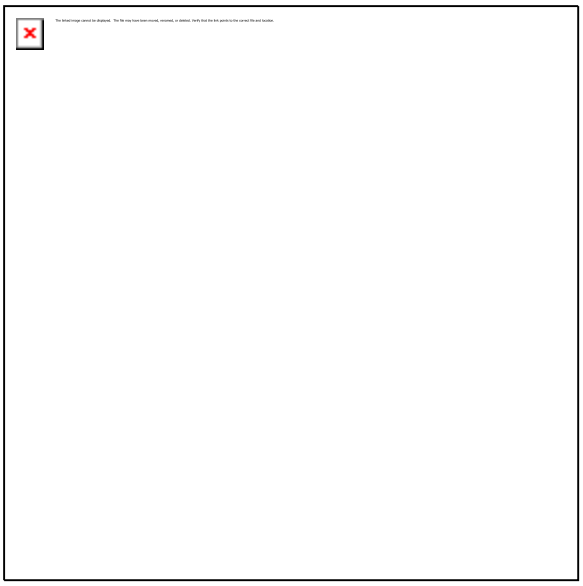

Рис. 18: запуск исполняемого файла

1) Создаю файл variant.asm с помощью утилиты touch (рис. 19).

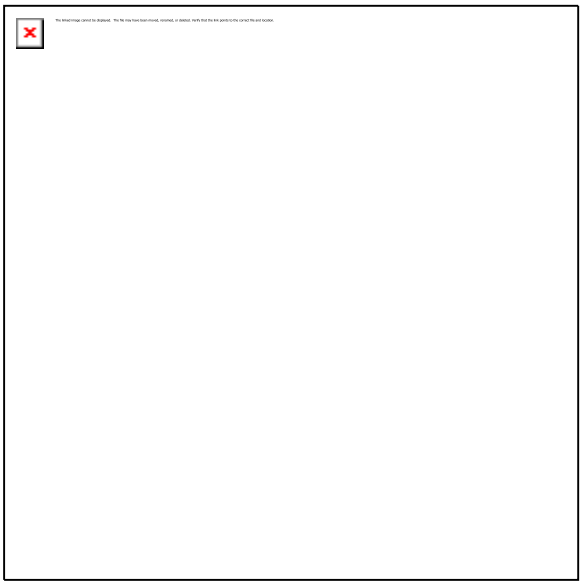

Рис. 19: создание файла

2) Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета (рис. 20).

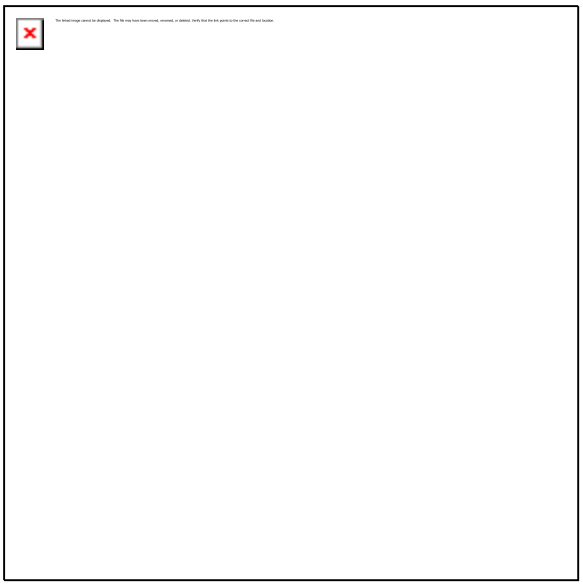

Рис. 20: редактирование файла

1) Создаю и запускаю исполняемый файл (рис. 21). Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 19.

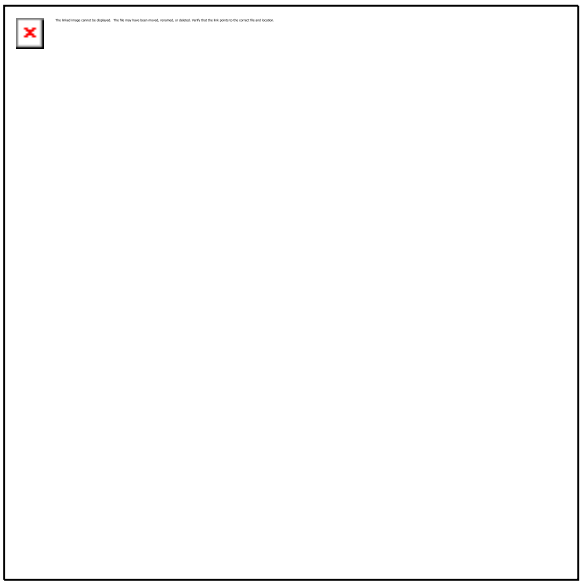

Рис. 21: запуск исполняемого файла

4.2.1 Ответы на вопросы по программе

1) За вывод сообщения “Ваш вариант” отвечают строки кода:

mov eax,rem

call sprint

1) 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.

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

2) 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

1) При выполнении инструкции div ebx остаток от деления записывается в

регистр edx

1) Инструкция inc edx увеличивает значение регистра edx на 1

2) За вывод наэкранрезультатов вычислений отвечают строки:

mov eax,edx

call iprintLF

2. **Выполнение заданий для самостоятельной работы**

1) Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения (1/3 * x + 5) * 7 (рис. 22). Это выражение было под вариантом 19.

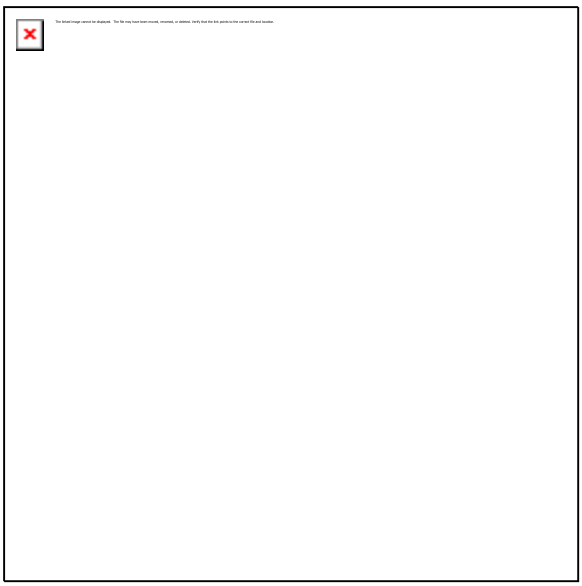

Рис. 22: написание программы

2) Создаю и запускаю исполняемый файл (рис. 23). При вводе значения 3, вывод- 35.

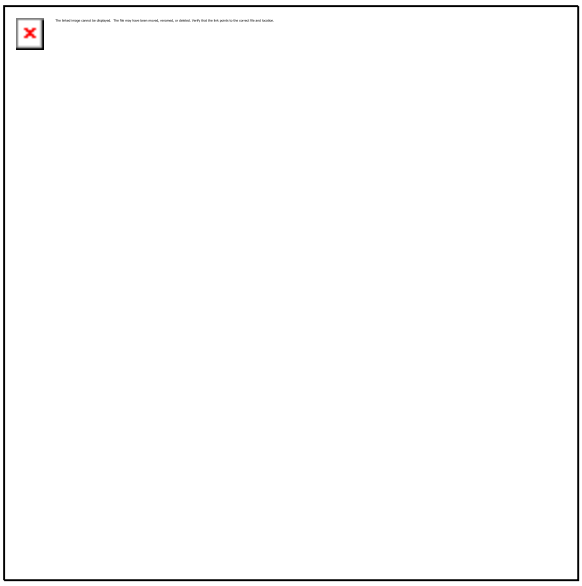

Рис. 23: запуск исполняемого файла

3) Провожу еще один запуск исполняемого файла для проверки работы программы с другим значением на входе (рис. 24). Программа отработала верно.

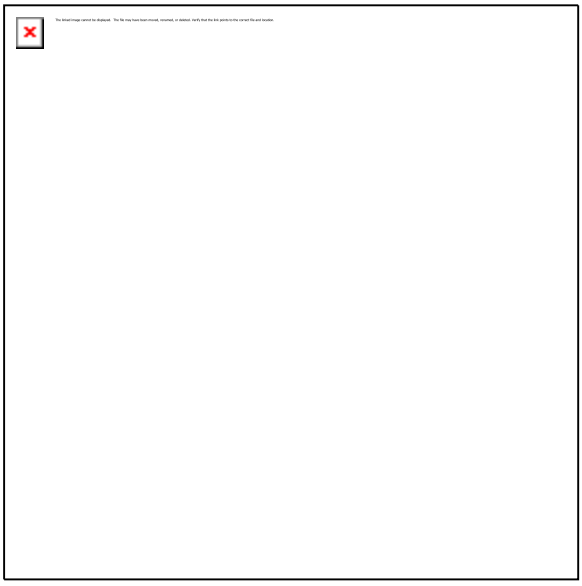

Рис. 24.1: редактирование файла

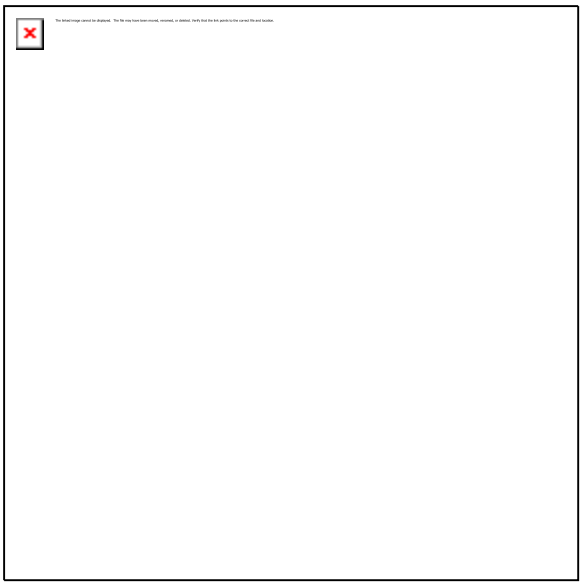

Рис. 24.2: запуск исполняемого файла

3. **Вывод**

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