Отчет по лабораторной работе № 6

Дисциплина: Архитектура компьютера

Дроздова Дарья Игоревна

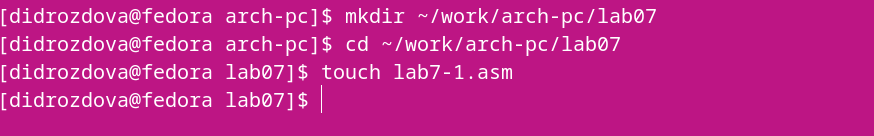
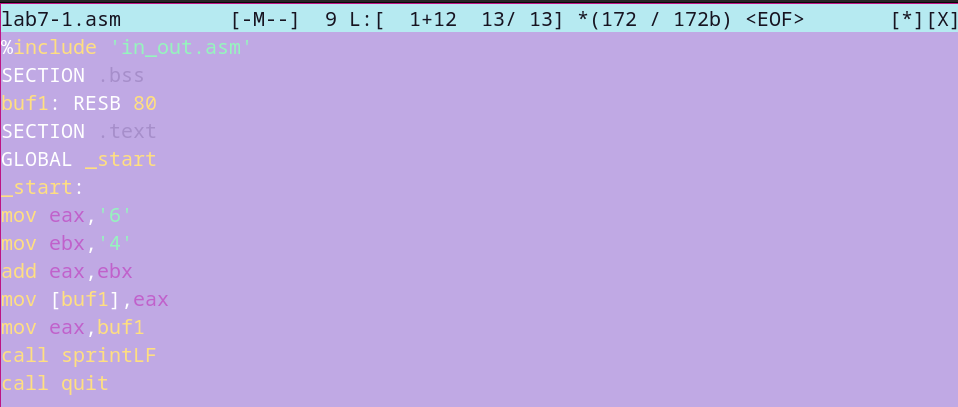
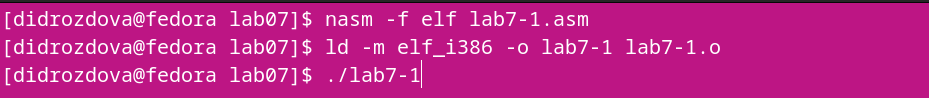
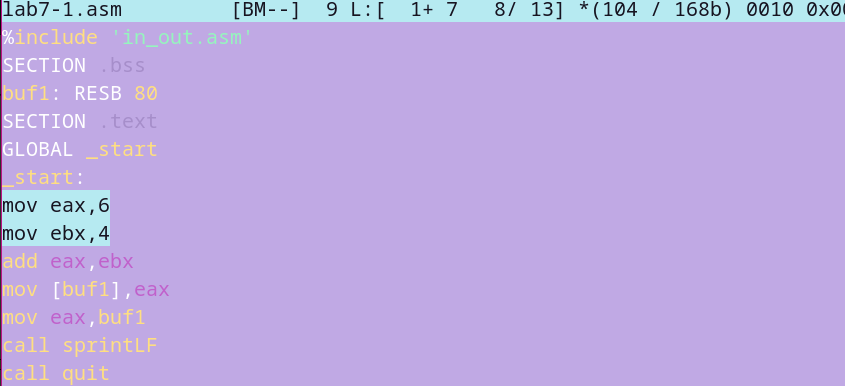
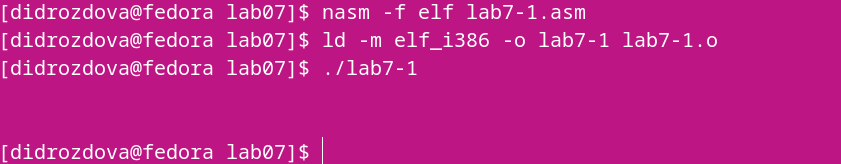
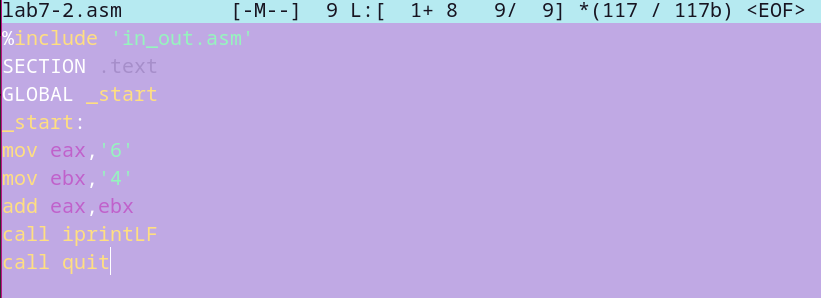
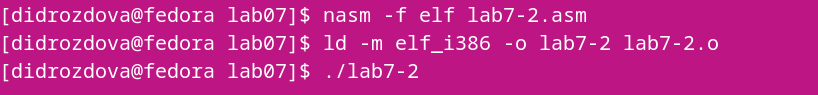
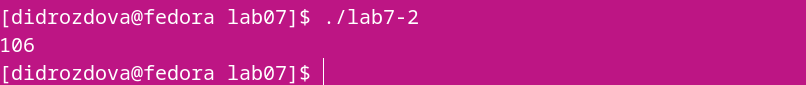
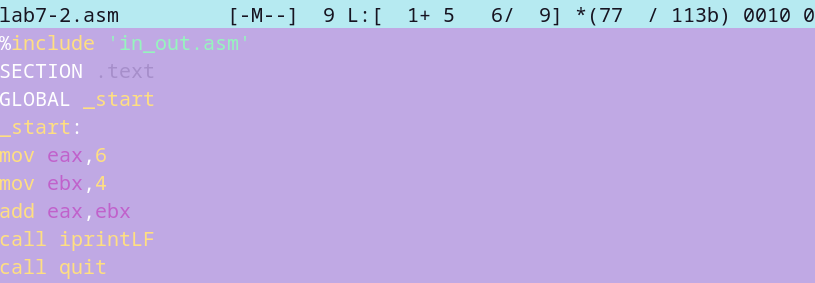
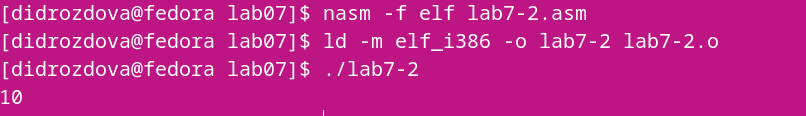
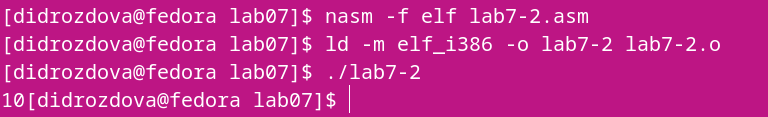
Содержание

# 1 Цель работы

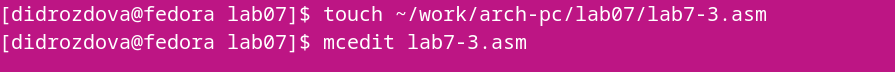
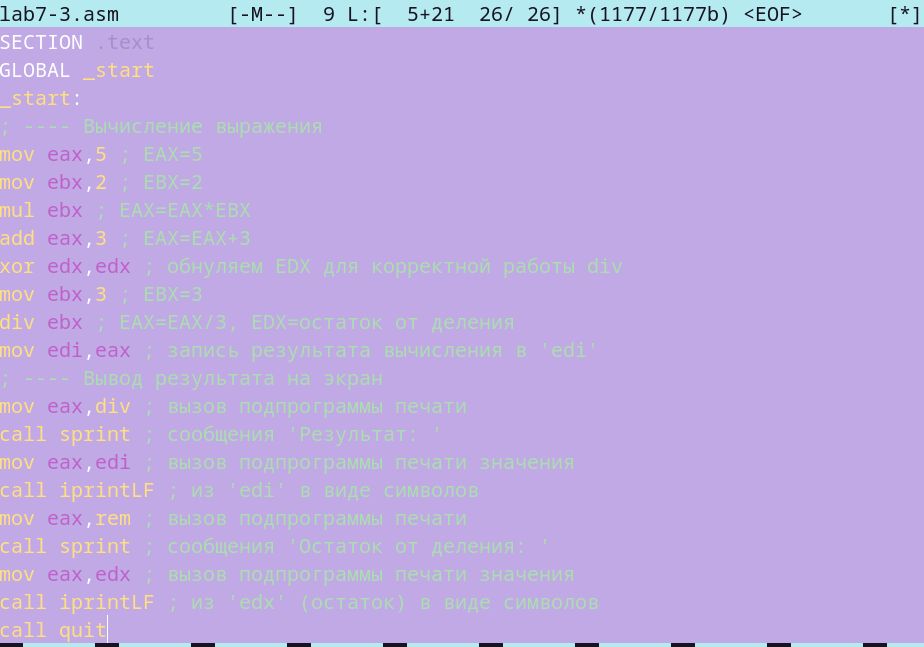
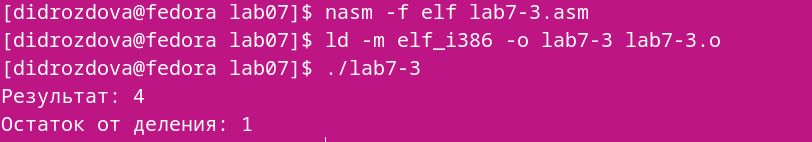
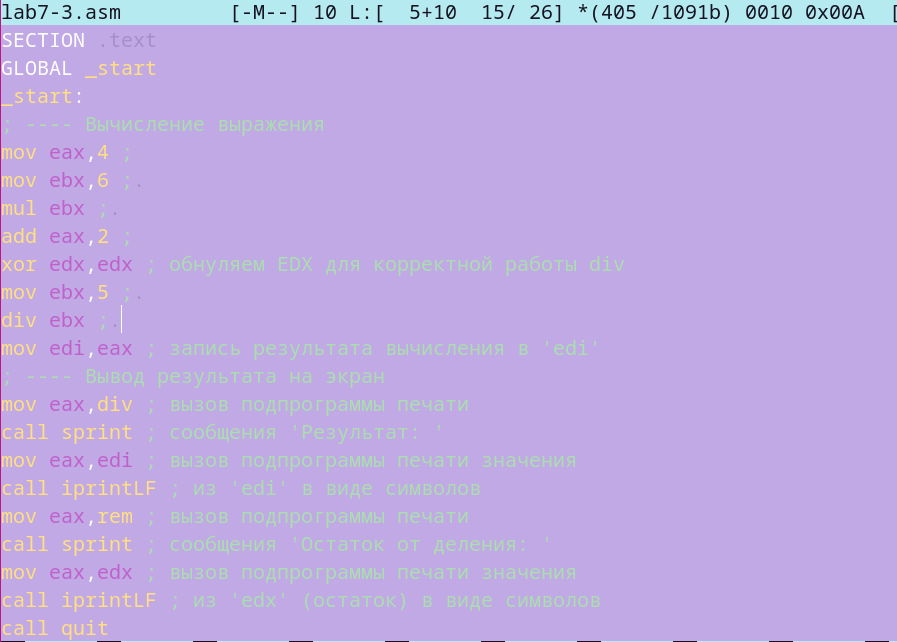
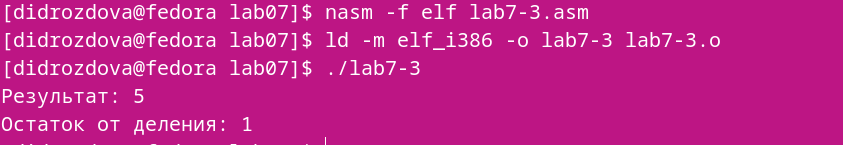
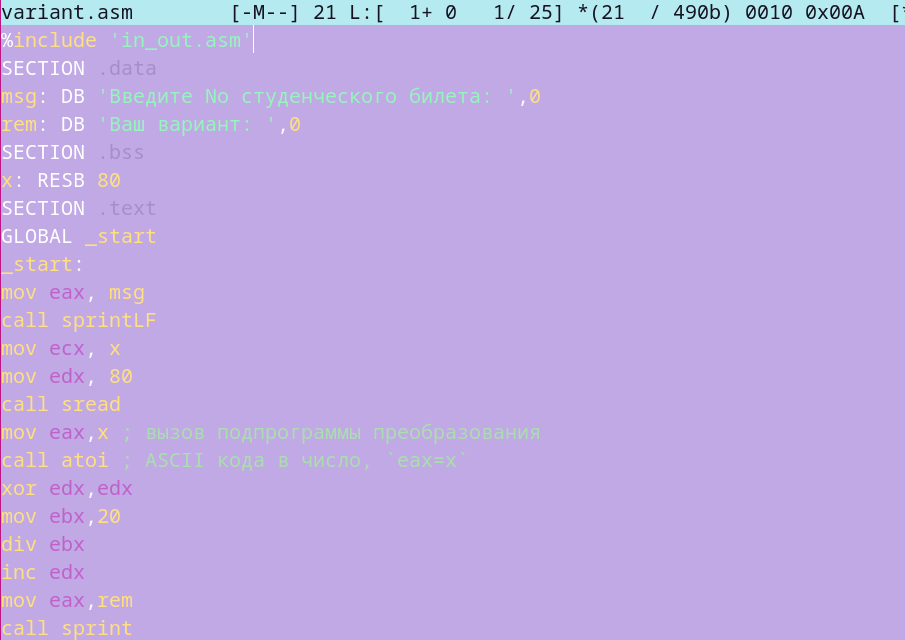
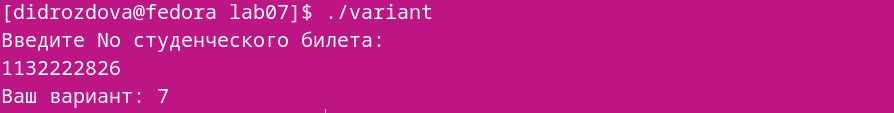
Целью данной лабораторной работы является освоение арифметических инструкций языка ассемблера NASM.

# 2 Выполнение лабораторной работы

1. **Символьные и численные данные в NASM**
   * Создаем каталог для программ лабораторной работы №6, переходим в него и создаем файл lab7-1.asm:

* 
  + Рассмотрим примеры программ вывода символьных и численных значений.
* *Пример №1:* Изучаем текст программы из листинга 7.1 и помещаем его в файл lab7-1.asm:
* 
* Создаем исполняемый файл и запускаем его:
* 
* 
* Результатом выполнения программы является символ ‘j’. Так вышло потому, что код символа 6 равен 00110110(54) в двоичном(десятичном) представлении, а код символа 4 – 00110100(52). Команда add eax,ebx запишет в регистр eax сумму кодов – 01101010(106), которая является кодом символа ‘j’.
* *Пример №2:* Изменим текст программы(из листинга 7.1) и вместо символов ‘4’ и ‘6’, запишем в регистры числа:
* 
* Создаем исполняемый файл и запускаем его:
* 
* Результатом выполнения данной программы является пустая строка. Действительно, если мы обратимся к ASCII таблице, то заметим, что сумма 6 и 4, равная 10, является кодом, отвечающим за перенос курсора на другую строку.
  + Воспользуемся подпрограммами для преобразования ASCII символов в числа и обратно(подпрограммы подключаем из файла in\_out.asm). Создаем файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 и вводим в него текст программы из листинга 7.2:
* 
* 
* Создаем исполняемый файл и запускаем его:
* 
* 
* Как и в первом случае, команда add складывает коды символов ‘6’ и ‘4’ (54+52=106). Однако, в отличии от программы из листинга 7.1, функция iprintLF позволяет вывести число, а не символ, кодом которого является это число.
  + Аналогично предыдущему примеру изменим символы на числа:
* 
* Создаем исполняемый файл и запускаем его:
* 
  + Теперь заменим функцию iprintLF на iprint. Создаем исполняемый файл и запускаем его:
* 
* 
* Заменив функцию, мы замечаем, что после вывода результата каретка не перемещается на строку ниже.

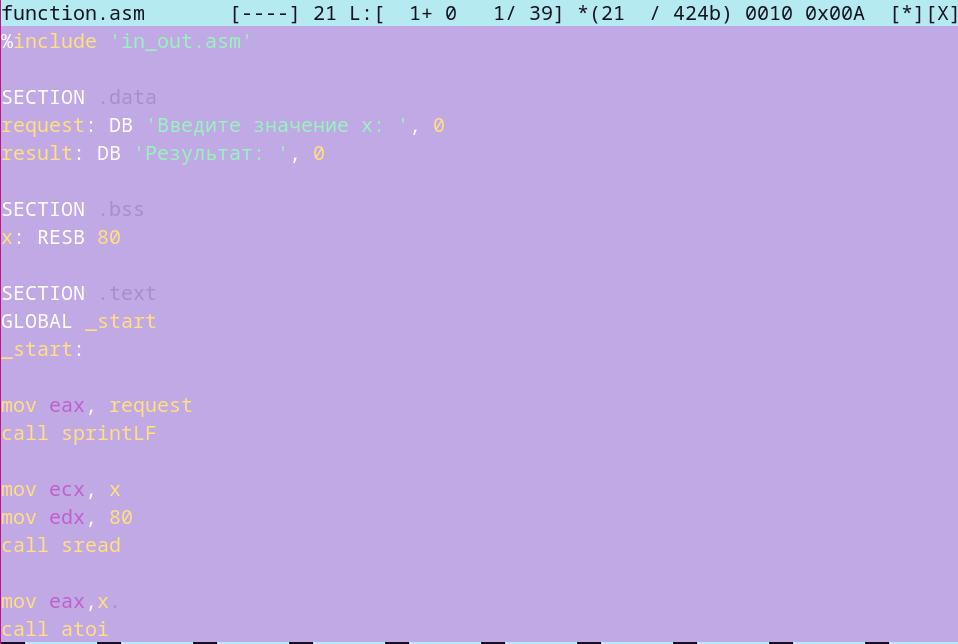
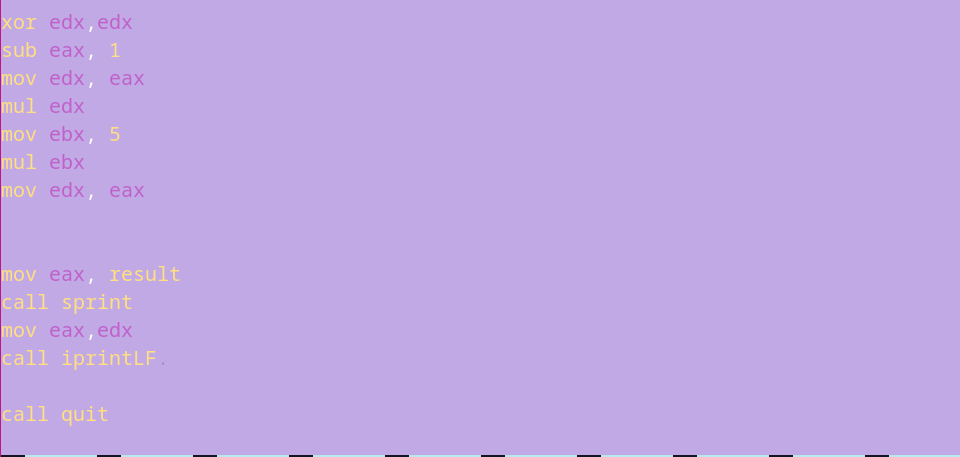
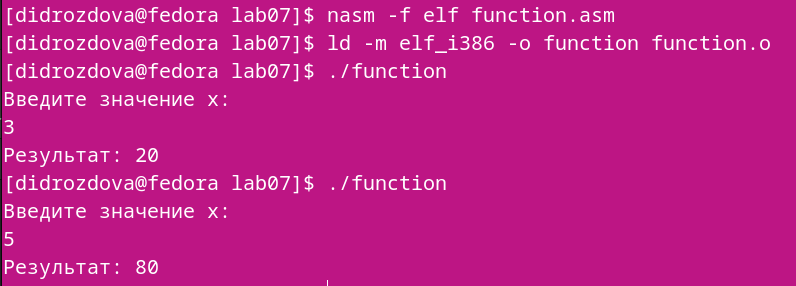
1. **Выполнение арифметических операций в NASM**
   * В качестве примера выполнения арифметических операций в NASM при- ведем программу вычисления арифметического выражения f(x) = (5 \* 2 + 3)/3. Создаем файл lab7-3.asm в каталоге ~/work/arch-pc/lab07:

* 
  + Внимательно изучаем текст программы из листинга 7.3 и введим в lab7-3.asm:
* 
* Создаем исполняемый файл и запускаем его:
* 
  + Изменяем текст программы для вычисления выражения f(x) = (4 \* 6 + 2)/5.
* 
* Создаем исполняемый файл и проверяем его работу:
* 
  + В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму: вывести запрос на введение № студенческого билета; вычислить номер варианта по формуле: (S mod 20) + 1, где S – номер студенческого билета (В данном случае a mod b– это остаток от деления a на b); вывести на экран номер варианта.
* Создаем файл variant.asm в каталоге ~/work/arch-pc/lab07, внимательно изучаем листинг 7.4 и вводим текст программы в файл variant.asm:
* 
* Создаем исполняемый файл и запускаем его:
* 
* 
* Проверяем результат работы программы, вычислив номер варианта аналитически(результат верный).

1. **Вопросы**
   1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

* mov eax,rem  
  call sprint
  1. Для чего используется следующие инструкции?
* mov ecx, x #запись адреса переменной х в “EAX”  
  mov edx, 80 #запись длины вводимого сообщения в “EBX”  
  call sread #вызов подпрограммы ввода сообщения
  1. Для чего используется инструкция “call atoi”?
* Для преобразования ASCII кода символа в число
  1. Какие строки листинга 7.4 отвечают за вычисления варианта?
* mov eax,x  
  call atoi  
  xor edx,edx  
  mov ebx,20  
  div ebx  
  inc edx
  1. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”?
* В регистр edx.
  1. Для чего используется инструкция “inc edx”?
* Увеличивает значение регистра ebx на 1
  1. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?
* mov eax,edx call iprintLF

1. **Задание для самостоятельной работы**

* В соответствии с вариантом(в моем случае - вариант №7) пишем программу функции: y = 5(x − 1)2
  + Создаем файл function.asm в каталоге ~/work/arch-pc/lab07:
* 
  + В файле function.asm пишем программу, вычисляющую значение функции при x1=3 и x2=5:
* 
* 
  + Создаем исполняемый файл, запускаем его и вводим с клавиатуры значения переменной x:
* 
* Результат работы программы корректен.

# 3 Выводы

В ходе выполнения данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM: сложение, вычитание, умножение, целое и дробное деление, а так же смена знача числа.