# Отчёт по лабораторной работе №6

**дисциплина: Архитектура компьютера**

Гайдук Софья Сергеевна

# Содержание

# Список иллюстраций

# Список таблиц

# 1 Цель работы

Освоить арифметические инструкции языка ассемблера NASM.

# 2 Выполнение лабораторной работы

Создадим каталог для программ лабораторной работы № 6, перейдем в него и создадим файл lab6-1.asm (рис. 2.1).


Рисунок 2.1: image1

Введем в файл lab6-1.asm текст программы данный в условии (рис. 2.2).

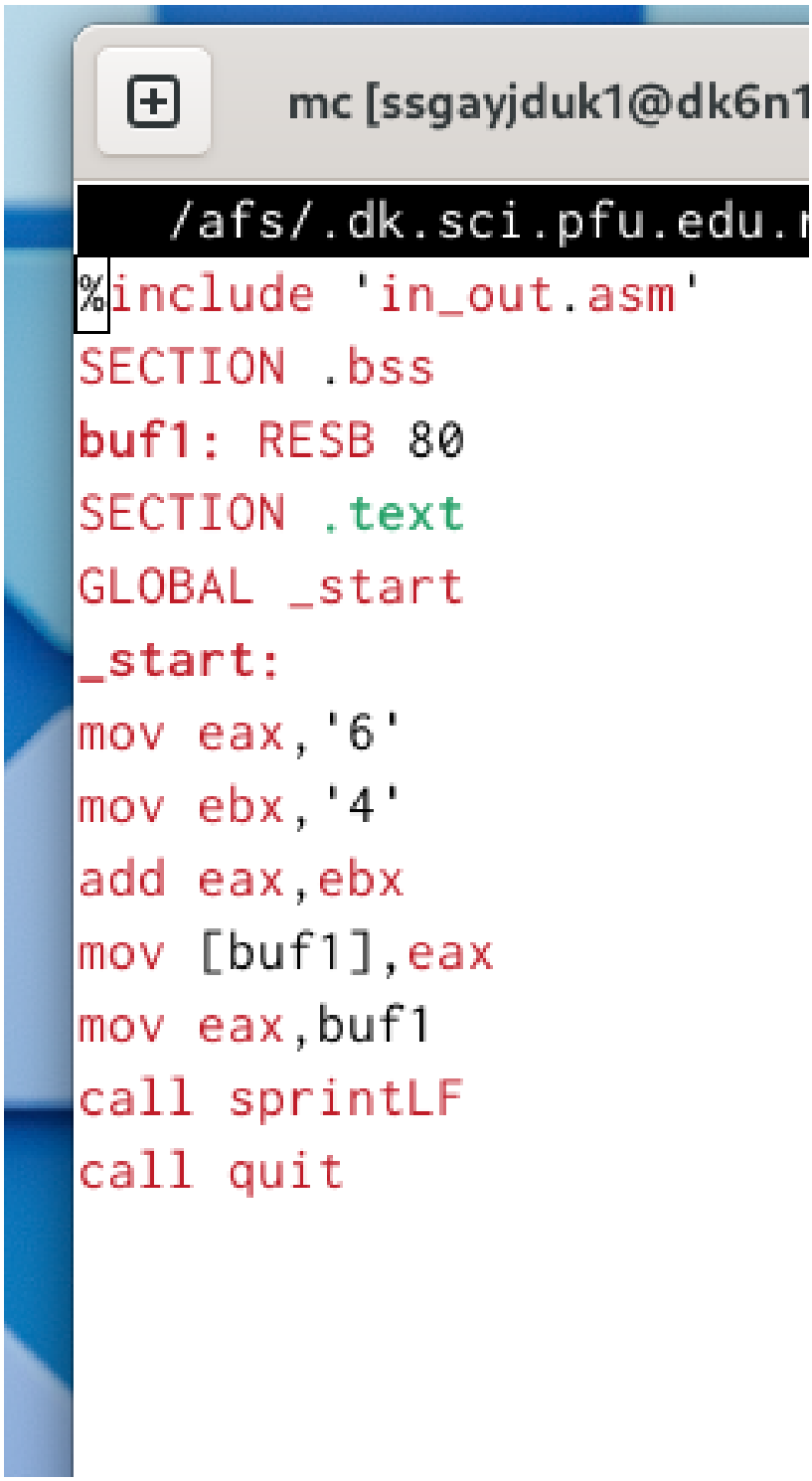


Рисунок 2.2: image2

Создадим исполняемый файл и запустим его (рис. 2.3).

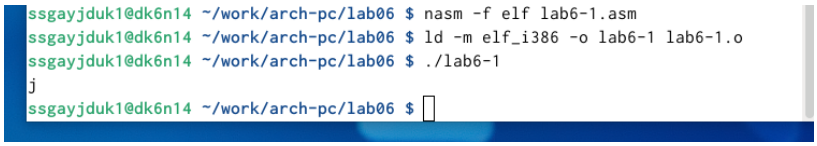


Рисунок 2.3: image3

Результатом будет символ j.

Ранее перенесли in_out.asm в каталог ~/work/arch-pc/lab06.

Изменим текст программы и вместо символов, запишем в регистры числа (рис. 2.4).

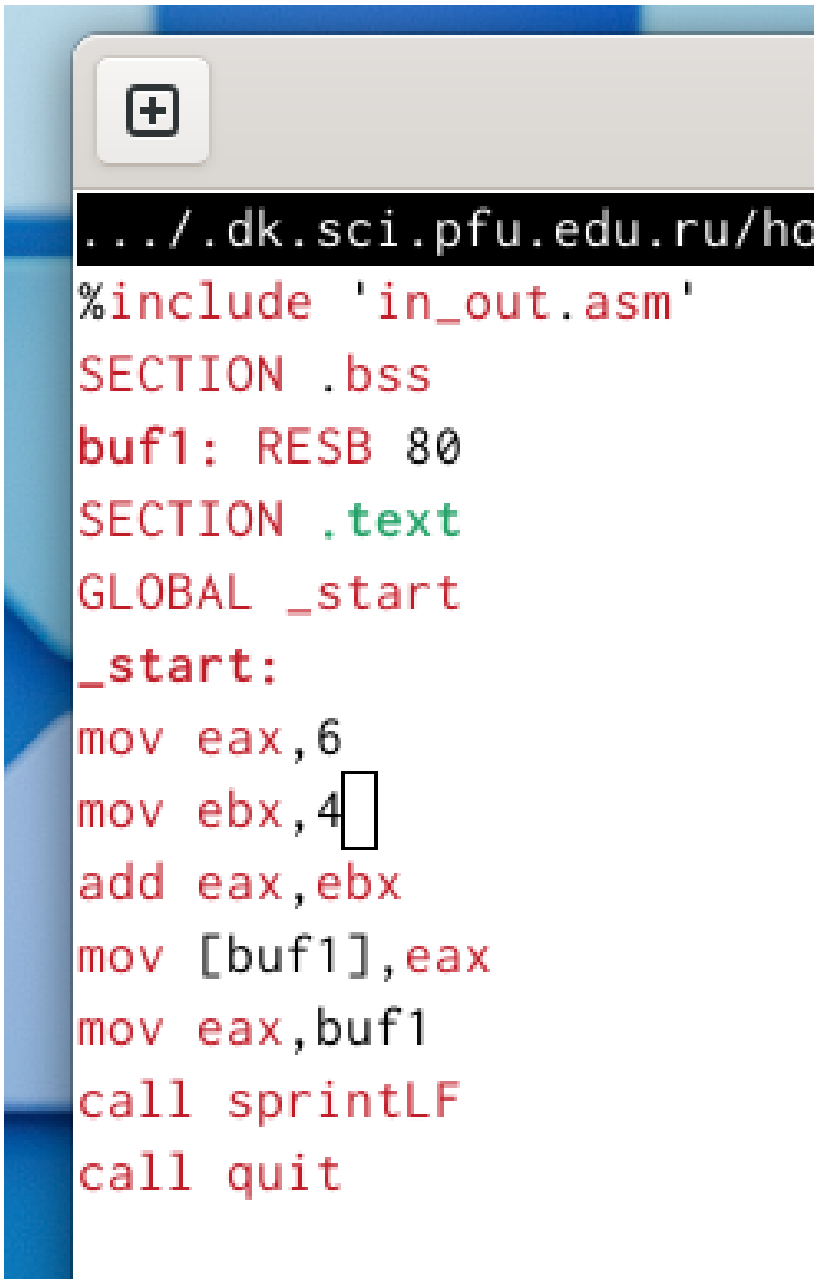


Рисунок 2.4: image4

Создадим исполняемый файл и запустим его (рис. 2.5).

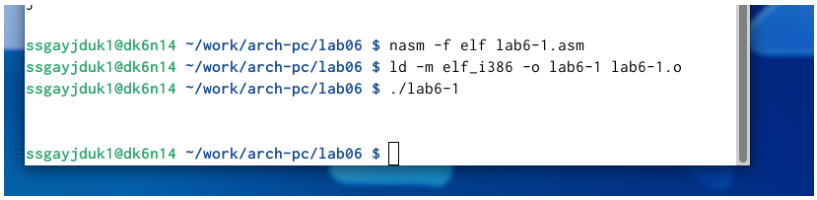


Рисунок 2.5: image5

Код 10 соответствует символу пробел (" "). Он также, как и символ j, отображается при выводе на экран.

Создадим файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и введем в него текст программы из условия (рис. 2.6)

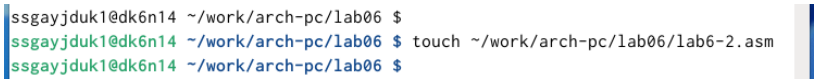


Рисунок 2.6: image6

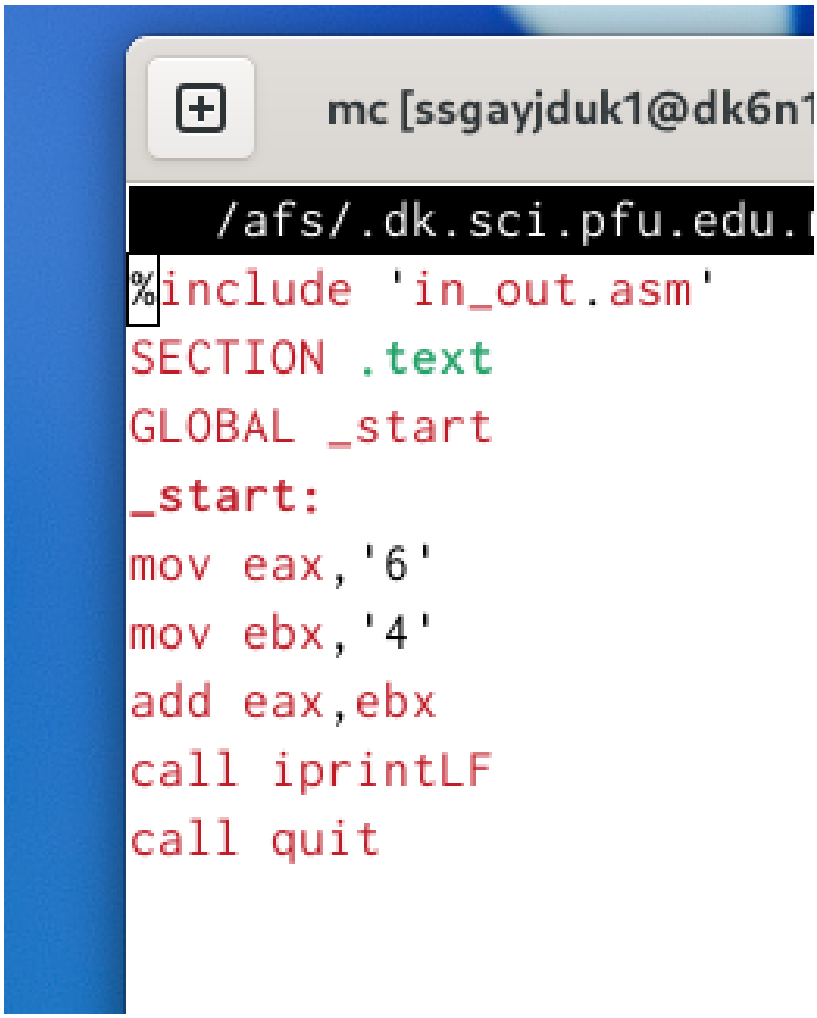


Рисунок 2.7: image7

Создадим исполняемый файл и запустим его (рис. 2.8).

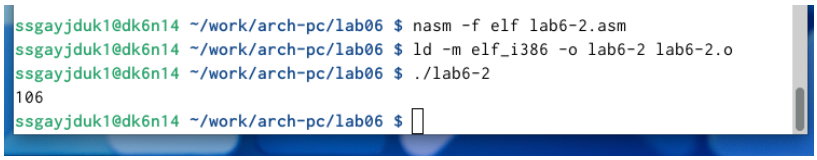


Рисунок 2.8: image8

Заменим символы на числа (рис. 2.9).

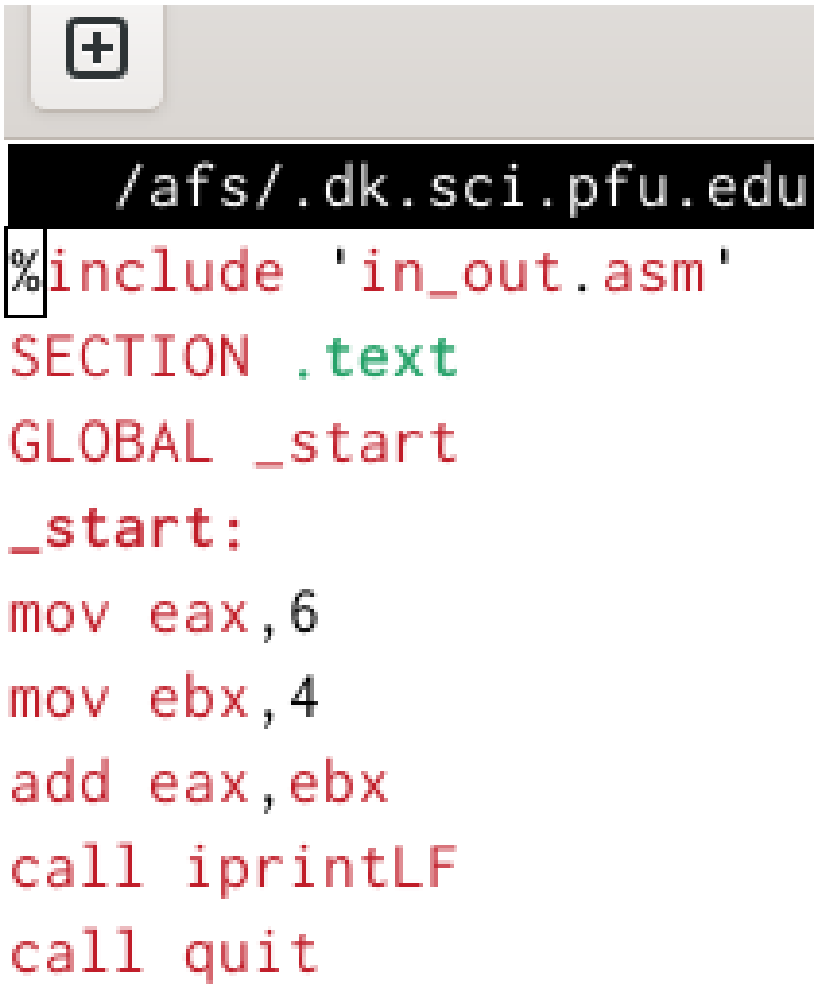


Рисунок 2.9: image9

Создадим исполняемый файл и запустим его (рис. 2.10).

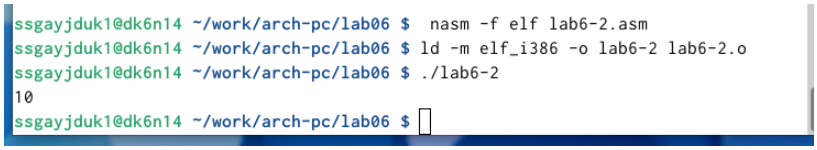


Рисунок 2.10: image10

Результатом будет выведенно 10 и командная строка будет ниже.

Заменим функцию iprintLF на iprint. Создадим исполняемый файл и запустим его (рис. 2.11)

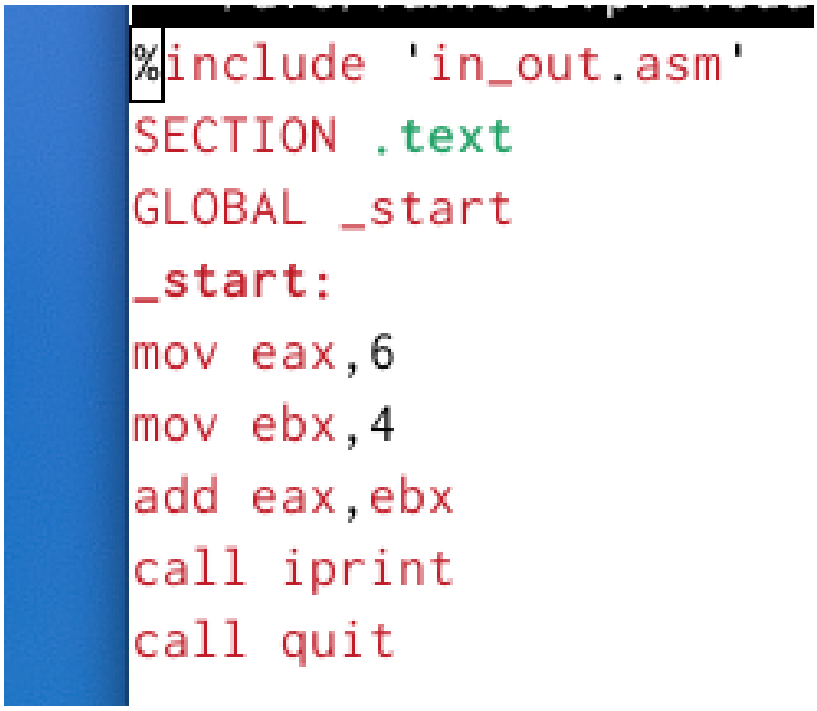


Рисунок 2.11: image11

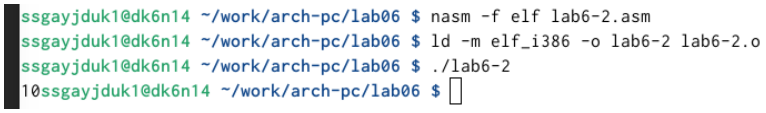


Рисунок 2.12: image12

Результатом будет выведенно 10 и командная строка идет сразу же после вывода.

Вывод функций iprintLF и iprint отличается тем, что в первом случае командная строка смещается на следующую строку, а во втором - остается и начинается сразу после вывода.

Создадим файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 (рис. 2.13).

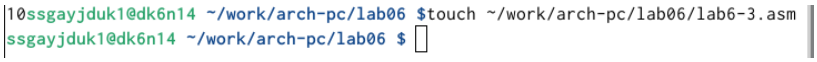


Рисунок 2.13: image13

Введем в lab6-3.asm программу из условия (рис. 2.14).

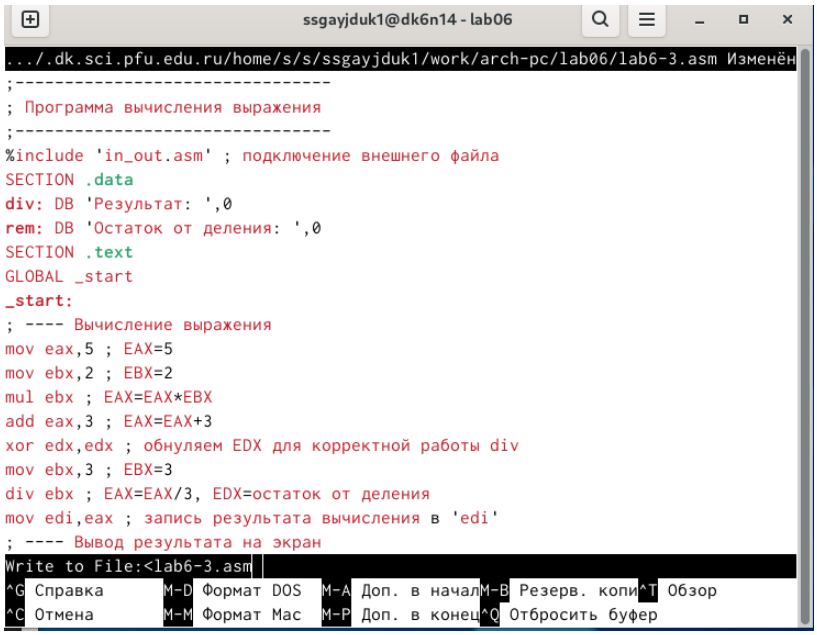


Рисунок 2.14: image14

Создадим исполняемый файл и запустим его (рис. 2.15).

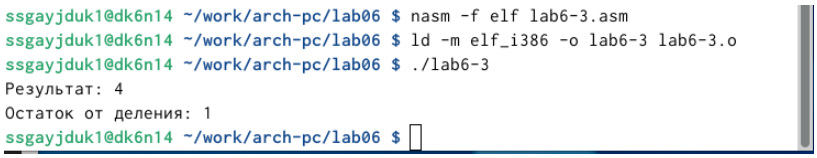


Рисунок 2.15: image15

Изменим текст программы для вычисления выражения (f(x)=(4*6+2)/5), создадим исполняемый файл и проверим его работу (рис. 2.16)

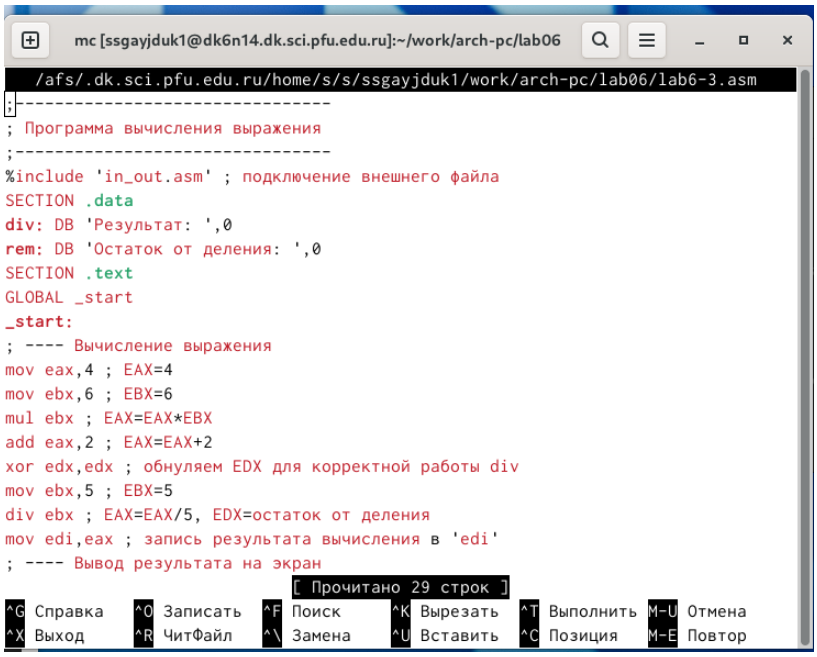


Рисунок 2.16: image16

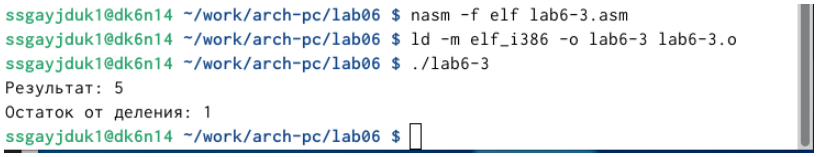


Рисунок 2.17: image17

Создадим файл variant.asm в каталоге ~/work/arch-pc/lab06 и проверим его существование (рис. 2.18)

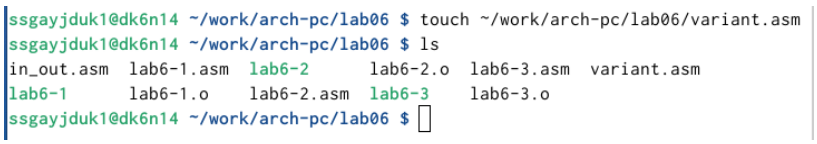


Рисунок 2.18: image18

Введем в файл текст программы из листинга (рис. 2.19)

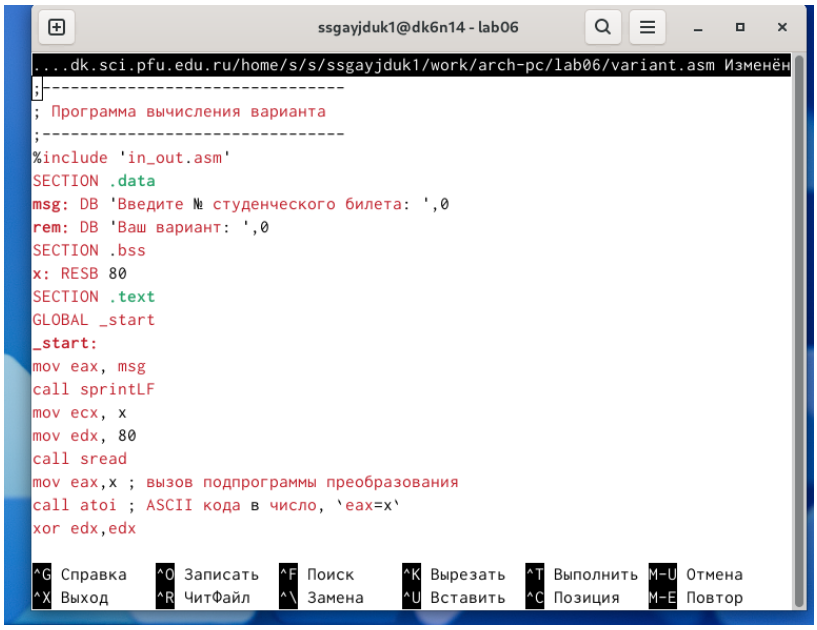


Рисунок 2.19: image19

Создадим исполняемый файл и запустим его, а также проверим результат работы программы вычислив номер варианта аналитически (с помощью языка питона) (рис. 2.20)

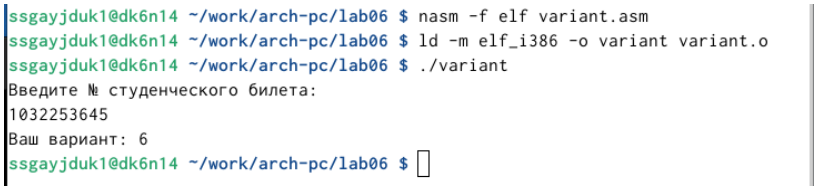


Рисунок 2.20: image20

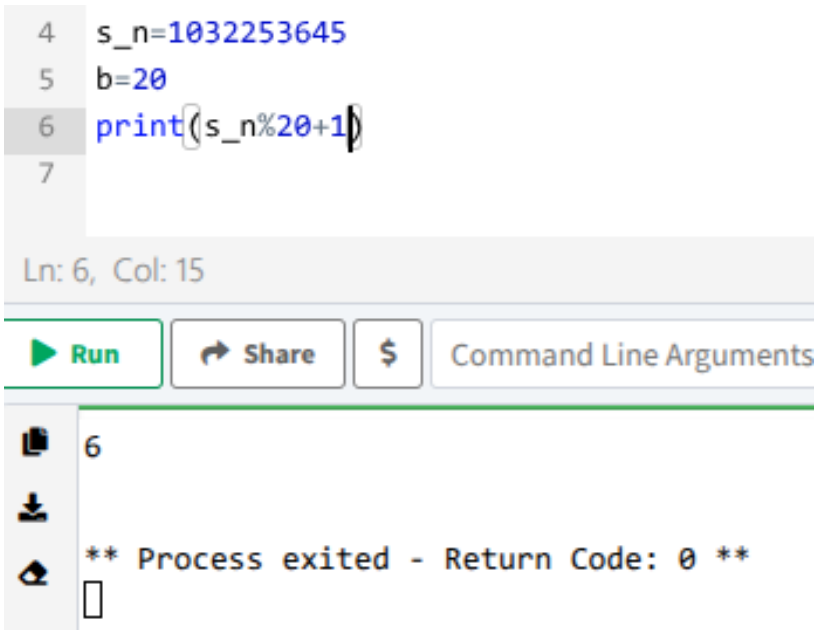


Рисунок 2.21: image21

Ответ сходится.

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения „Ваш вариант:“?

За вывод на экран сообщения „Ваш вариант:“ отвечают строки mov eax,rem и call sprint(image23), rem - название переменной, в которой находится сообщение „Ваш вариант:“. (рис. 2.22)


Рисунок 2.22: image22


Рисунок 2.23: image23

2. Для чего используется следующие инструкции? (рис. 2.24)

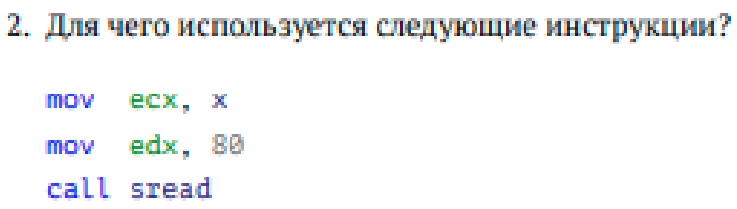


Рисунок 2.24: image24

mov ecx, x - используется для указания адреса переменной x в регистре ecx. mov edx, 80 - используется для указания размера или длины в байтах, в данной ситуации 80 call sread - вызов функции sread для чтения выше заданного кода.

3. Для чего используется инструкция «call atoi»?

call atoi - вызов функции atoi для преобразования ASCII строки в целое число.

4. Какие строки листинга 6.4 отвечают за вычисления варианта?

За вычисления варианта отвечают строки (рис. 2.25)

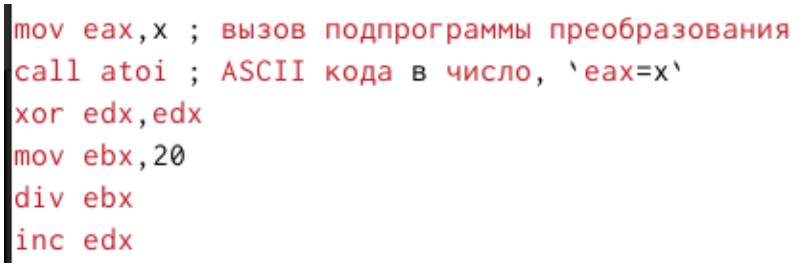


Рисунок 2.25: image25

5. В какой регистр записывается остаток от деления при выполнении инструкции «div ebx»?

Остаток от деления при выполнении инструкции «div ebx» записывается в edx.

6. Для чего используется инструкция «inc edx»?

Для увеличения значения регистра edx на единицу.

7. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений?

Строки mov eax,edx и call iprintLF (рис. 2.26)


Рисунок 2.26: image26

# 3 Задания для самоятоятельной работы

Напишем программу вычисления выражения x^3/2+1 (рис. 3.1)

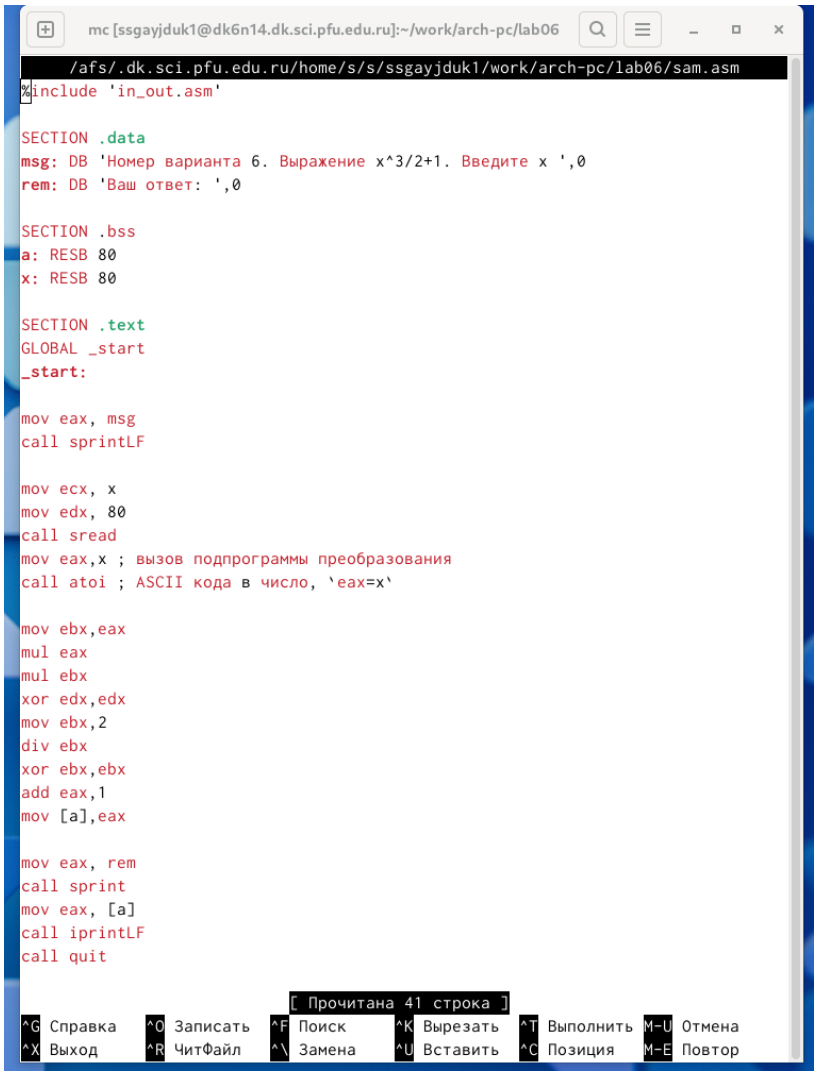


Рисунок 3.1: image27

Создадим исполняемый файл и проверим его работу для значений x=2 и x=5, заданных в условии (рис. 3.2)

```
ssgayjduk1@dk6n14 ~/work/arch-pc/lab06 $ nasm -f elf sam.asm
ssgayjduk1@dk6n14 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o sam sam.o
ssgayjduk1@dk6n14 ~/work/arch-pc/lab06 $ ./sam
Номер варианта 6. Выражение x^3/2+1. Введите x
2
Ваш ответ: 5
ssgayjduk1@dk6n14 ~/work/arch-pc/lab06 $ ./sam
Номер варианта 6. Выражение x^3/2+1. Введите x
5
Ваш ответ: 63
ssgayjduk1@dk6n14 ~/work/arch-pc/lab06 $
```

Рисунок 3.2: image28

Отправим на Github (рис. 3.3)

```
ssgayjduk1@dk6n14 ~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab
06/report $ git add .
ssgayjduk1@dk6n14 ~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab
06/report $ git commit -am 'feat(main): add lab-6'
[master 2b93473] feat(main): add lab-6
 18 files changed, 290 insertions(+)
 create mode 100644 labs/lab06/report/arch-pc--lab06--report.docx
 create mode 100644 labs/lab06/report/arch-pc--lab06--report.pdf
 create mode 100644 labs/lab06/report/in_out.asm
 create mode 100755 labs/lab06/report/lab6-1
 create mode 100644 labs/lab06/report/lab6-1.asm
 create mode 100644 labs/lab06/report/lab6-1.o
 create mode 100755 labs/lab06/report/lab6-2
 create mode 100644 labs/lab06/report/lab6-2.asm
 create mode 100644 labs/lab06/report/lab6-2.o
 create mode 100755 labs/lab06/report/lab6-3
 create mode 100644 labs/lab06/report/lab6-3.asm
 create mode 100644 labs/lab06/report/lab6-3.o
 create mode 100755 labs/lab06/report/sam
 create mode 100644 labs/lab06/report/sam.asm
 create mode 100644 labs/lab06/report/sam.o
 create mode 100755 labs/lab06/report/variant
 create mode 100644 labs/lab06/report/variant.asm
 create mode 100644 labs/lab06/report/variant.o
ssgayjduk1@dk6n14 ~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab
06/report $ git push
Перечисление объектов: 27, готово.
Подсчет объектов: 100% (27/27), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (23/23), готово.
Запись объектов: 100% (23/23), 2.22 МиБ | 3.30 МиБ/с, готово.
Total 23 (delta 8), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (8/8), completed with 2 local objects.
To github.com:SofiaGayduk/study_2025-2026_arh-pc.git
   a0bb643..2b93473  master -> master
ssgayjduk1@dk6n14 ~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab
06/report $
```

Рисунок 3.3: image29

# 4 Выводы

Мы изучили арифметические инструкции языка ассемблера NASM.