

Лабораторная работа №5

НКАбд-03-22

Шубнякова Дарья

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
5	Выводы	15
	Список литературы	16

Список иллюстраций

4.1	Создание каталога для работы	9
4.2	Создание файла hello.asm	10
4.3	Создание объектного файла	10
4.4	Создание файла с использованием расширенного синтаксиса . .	11
4.5	Работа с компоновщиком LD	12
4.6	Запуск файла	12
4.7	Задание 1	13
4.8	Задание 2 и 3	13
4.9	Задание 4	14

Список таблиц

1 Цель работы

Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

2 Задание

- 1) В каталоге `~/work/arch-рс/lab05` с помощью команды `ср` создайте копию файла `hello.asm` с именем `lab5.asm`
- 2) С помощью любого текстового редактора внесите изменения в текст программы в файле `lab5.asm` так, чтобы вместо `Hello world!` на экран выводилась строка с вашими фамилией и именем.
- 3) Оттранслируйте полученный текст программы `lab5.asm` в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.
- 4) Скопируйте файлы `hello.asm` и `lab5.asm` в Ваш локальный репозиторий в каталог `~/work/study/2022-2023/“Архитектура компьютера”/arch-рс/labs/lab05/`. Загрузите файлы на Github.

3 Теоретическое введение

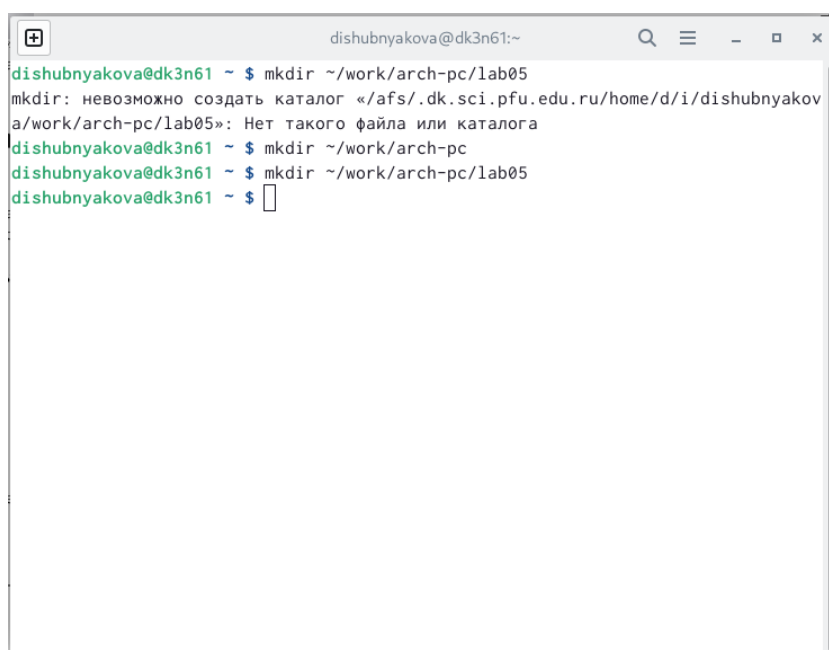
Основными функциональными элементами любой электронно-вычислительной машины (ЭВМ) являются центральный процессор, память и периферийные устройства. Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены. Физически шина представляет собой большое количество проводников, соединяющих устройства друг с другом. В современных компьютерах проводники выполнены в виде электропроводящих дорожек на материнской (системной) плате.

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный язык низкого уровня. Можно считать, что он больше любых других языков приближен к архитектуре ЭВМ и её аппаратным возможностям, что позволяет получить к ним более полный доступ, нежели в языках высокого уровня, таких как C/C++, Perl, Python и пр. Заметим, что получить полный доступ к ресурсам компьютера в современных архитектурах нельзя, самым низким уровнем работы прикладной программы является обращение напрямую к ядру операционной системы. Именно на этом уровне и работают программы, написанные на ассемблере. Но в отличие от языков высокого уровня ассемблерная программа содержит только тот код, который ввёл программист. Таким образом язык ассемблера — это язык, с помощью которого понятным для человека образом пишутся команды для процессора. Следует отметить, что процессор понимает не команды ассемблера, а последовательности из нулей и единиц — машинные коды. До появления языков ассемблера программистам приходилось писать программы, используя только лишь машинные коды, которые были крайне сложны

для запоминания, так как представляли собой числа, записанные в двоичной или шестнадцатеричной системе счисления. Преобразование или трансляция команд с языка ассемблера в исполняемый машинный код осуществляется специальной программой транслятором — Ассемблер.

4 Выполнение лабораторной работы

Создаем каталог для работы с программами на языке ассемблера NASM. Поскольку в папке work не было каталога arch-pc, я создаю также и его.(рис. 4.1)



```
dishubnyakova@dk3n61:~  
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05  
mkdir: невозможно создать каталог «/afs/.dk.sci.pfu.edu.ru/home/d/i/dishubnyakov  
a/work/arch-pc/lab05»: Нет такого файла или каталога  
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc  
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05  
dishubnyakova@dk3n61 ~ $
```

Рис. 4.1: Создание каталога для работы

Создаем текстовый файл hello.asm и открываем его с помощью текстового редактора. Вставляем туда текст из заданной лабораторной работы и сохраняем файл.(рис. 4.2)

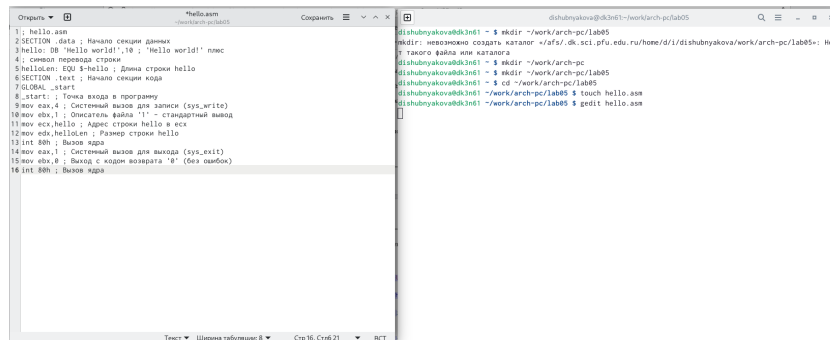


Рис. 4.2: Создание файла hello.asm

Превращаем наш текст в объектный код с помощью транслятора NASM. С помощью команды `ls` проверяем его наличие и видим, что название созданного объектного файла hello.o.(рис. 4.3)

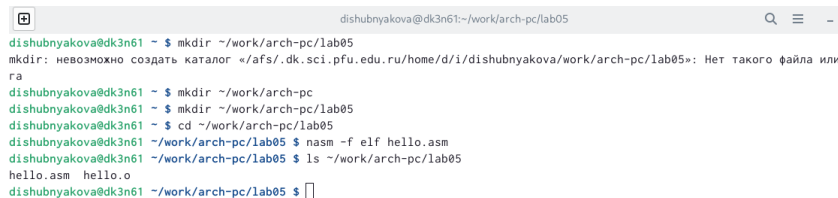
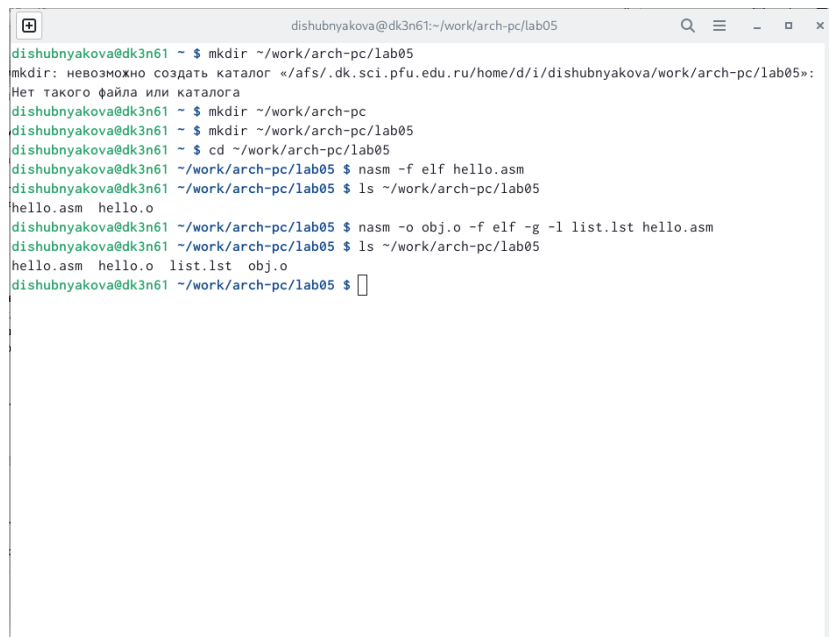


Рис. 4.3: Создание объектного файла

Выполняем заданную в работе команду, которая компилирует исходный файл с определенным именем `obj.o` в формате `elf`, где будут включены символы для отладки и создан файл листинга. С помощью команды `ls` снова проверяем правильность выполненных действий.(рис. 4.4)



```
dishubnyakova@dk3n61:~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05
mkdir: невозможно создать каталог «/afs/.dk.sci.pfu.edu.ru/home/d/i/dishubnyakova/work/arch-pc/lab05»:
Нет такого файла или каталога
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~ $ cd ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -f elf hello.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello.asm  hello.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello.asm  hello.o  list.lst  obj.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $
```

Рис. 4.4: Создание файла с использованием расширенного синтаксиса

Передаем наш объектный файл на обработку компоновщику LD, чтобы получить исполняемую программу. Проверяем, что файл успешно создан. Затем выполняем команду из заданной работы. Имя исполняемого файла будет main, а объектный файл, из которого он собран, называется obj.o(рис. 4.5)

```
dishubnyakova@dk3n61:~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05
mkdir: невозможно создать каталог «/afs/.dk.sci.pfu.edu.ru/home/d/i/dishubnyakova/work/arch-pc/lab05»:
Нет такого файла или каталога
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~ $ cd ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -f elf hello.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello.asm hello.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello.asm hello.o list.lst obj.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ man nasm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ld -m elf_i386 hello.o -o hello
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello hello.asm hello.o list.lst obj.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ld -m elf_i386 obj.o -o main
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $
```

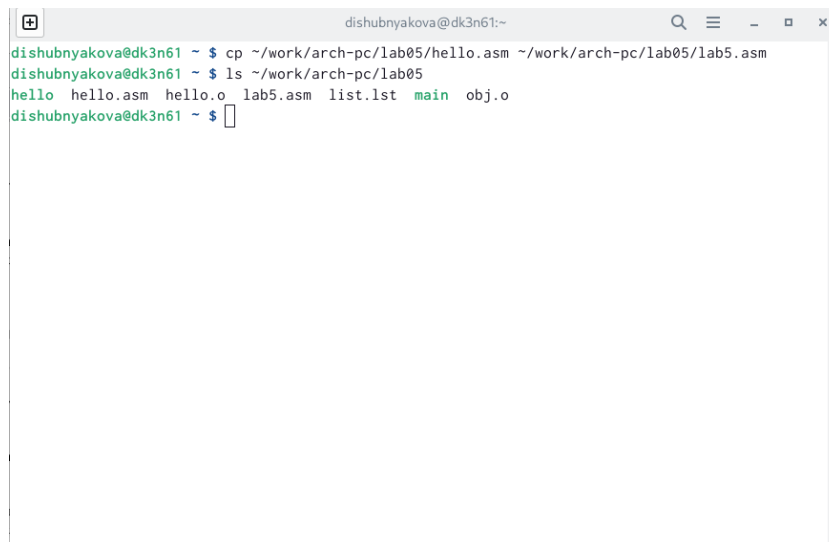
Рис. 4.5: Работа с компоновщиком LD

Запускаем наш файл и убеждаемся в его работе.(рис. 4.6)

```
dishubnyakova@dk3n61:~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05
mkdir: невозможно создать каталог «/afs/.dk.sci.pfu.edu.ru/home/d/i/dishubnyakova/work/arch-pc/lab05»:
Нет такого файла или каталога
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc
dishubnyakova@dk3n61 ~ $ mkdir ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~ $ cd ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -f elf hello.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello.asm hello.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello.asm hello.o list.lst obj.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ man nasm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ld -m elf_i386 hello.o -o hello
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ls ~/work/arch-pc/lab05
hello hello.asm hello.o list.lst obj.o
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ld -m elf_i386 obj.o -o main
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ./hello
Hello world!
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $
```

Рис. 4.6: Запуск файла

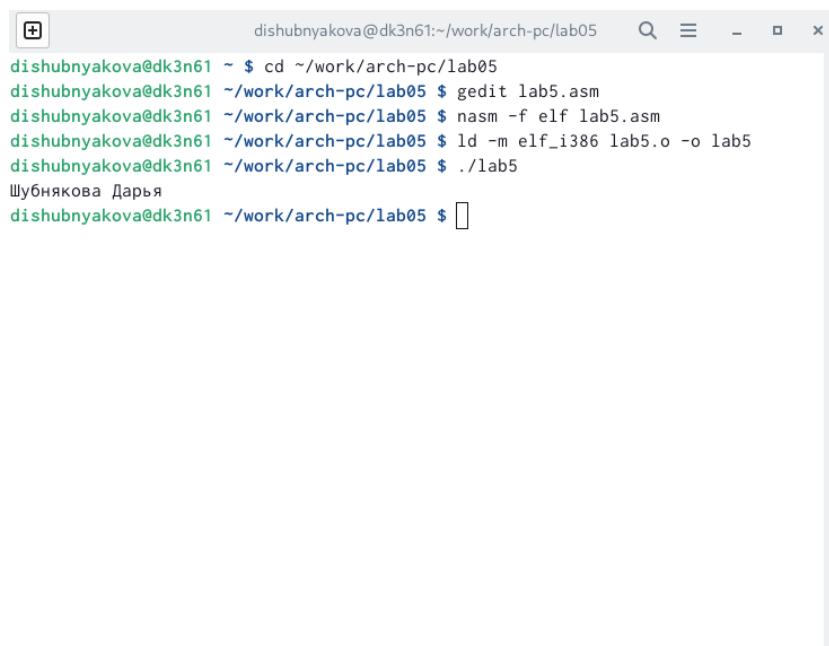
С помощью команды `cp` копируем файл `hello.asm` с новым именем `lab5.asm` и убеждаемся в том, что он был создан.(рис. 4.7)

A terminal window with the title bar 'dishubnyakova@dk3n61:~'. The terminal shows the following commands and output:

```
dishubnyakova@dk3n61 ~ $ cp ~/work/arch-pc/lab05/hello.asm ~/work/arch-pc/lab05/lab5.asm
dishubnyakova@dk3n61 ~ $ ls ~/work/arch-pc/lab05
hello  hello.asm  hello.o  lab5.asm  list.lst  main  obj.o
dishubnyakova@dk3n61 ~ $
```

Рис. 4.7: Задание 1

С помощью текстового редактора меняем “Hello world!” на имя и фамилию. После того как файл был оттранслирован в объектный файл и затем откомпонован, запускаем и видим строку с именем и фамилией.(рис. 4.8)

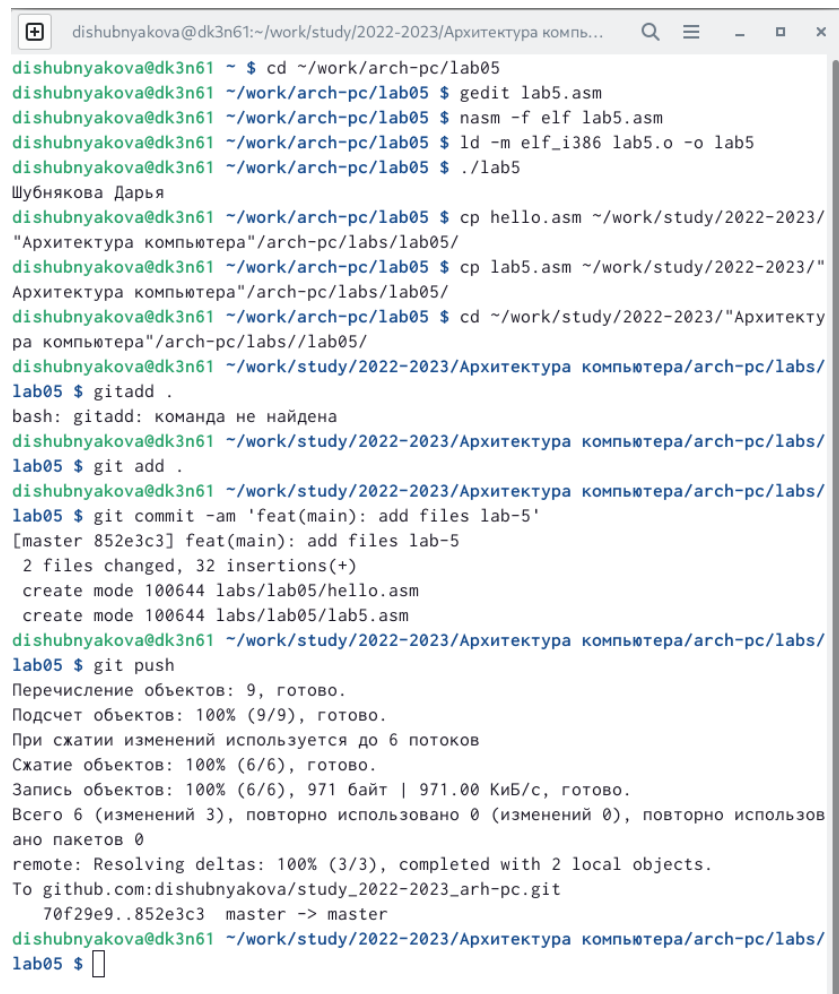
A terminal window with the title bar 'dishubnyakova@dk3n61:~/work/arch-pc/lab05'. The terminal shows the following commands and output:

```
dishubnyakova@dk3n61 ~ $ cd ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ gedit lab5.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -f elf lab5.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ld -m elf_i386 lab5.o -o lab5
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ./lab5
Шубнякова Дарья
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $
```

Рис. 4.8: Задание 2 и 3

Копируем файлы в наш локальный репозиторий и загружаем их на github.(рис.

4.9)



```
dishubnyakova@dk3n61:~/work/study/2022-2023/Архитектура компь...
dishubnyakova@dk3n61 ~ $ cd ~/work/arch-pc/lab05
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ gedit lab5.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ nasm -f elf lab5.asm
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ld -m elf_i386 lab5.o -o lab5
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ ./lab5
Шубнякова Дарья
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ cp hello.asm ~/work/study/2022-2023/
"Архитектура компьютера"/arch-pc/labs/lab05/
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ cp lab5.asm ~/work/study/2022-2023/"
Архитектура компьютера"/arch-pc/labs/lab05/
dishubnyakova@dk3n61 ~/work/arch-pc/lab05 $ cd ~/work/study/2022-2023/"Архитекту
ра компьютера"/arch-pc/labs//lab05/
dishubnyakova@dk3n61 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/
lab05 $ gitadd .
bash: gitadd: команда не найдена
dishubnyakova@dk3n61 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/
lab05 $ git add .
dishubnyakova@dk3n61 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/
lab05 $ git commit -am 'feat(main): add files lab-5'
[master 852e3c3] feat(main): add files lab-5
2 files changed, 32 insertions(+)
create mode 100644 labs/lab05/hello.asm
create mode 100644 labs/lab05/lab5.asm
dishubnyakova@dk3n61 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/
lab05 $ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 971 байт | 971.00 КиБ/с, готово.
Всего 6 (изменений 3), повторно использовано 0 (изменений 0), повторно использов
ано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:dishubnyakova/study_2022-2023_arh-pc.git
70f29e9..852e3c3 master -> master
dishubnyakova@dk3n61 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/
lab05 $
```

Рис. 4.9: Задание 4

5 Выводы

Процесс создания ассемблерной программы включает в себя набор текста, трансляцию, компоновку и запуск. В ходе работы мы прошли через все этапы и познакомились с тем, как это реализуется через терминал.

Список литературы