|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| *Федеральное государственное бюджетное образовательное учреждение высшего образования* ***«МИРЭА – Российский технологический университет»***  **РТУ МИРЭА** |

Институт Информационных технологий (ИТ)

Кафедра Математического обеспечения и стандартизации информационных технологий (МОСИТ)

|  |
| --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3** |
| **по дисциплине** |
| **«Технология разработки программных приложений»**  **Тема: «Docker»** |

|  |  |
| --- | --- |
| Выполнил студент группы \_\_\_\_\_ИКБО-03-21\_ | Хречко С.В, |
| Принял ассистент кафедры МОСИТ | Петрова А. А. |

|  |  |  |
| --- | --- | --- |
| Практическая работа выполнена | «\_\_»\_\_\_\_\_\_\_2023\_г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2023\_ г. | *(подпись руководителя)* |

Москва 2023г.

Оглавление

[Задачи 2](#_Toc133270654)

[Выполнение 3](#_Toc133270655)

[Раздел 1 3](#_Toc133270656)

[Раздел 2 3](#_Toc133270657)

[Раздел 3 4](#_Toc133270658)

[Раздел 4 6](#_Toc133270659)

[Раздел 5 7](#_Toc133270660)

[Раздел 5.1 8](#_Toc133270661)

[Раздел 5.2 9](#_Toc133270662)

[Раздел 6 10](#_Toc133270663)

[Раздел 7 10](#_Toc133270664)

[Раздел 8 11](#_Toc133270665)

[Вывод 13](#_Toc133270666)

**Цель работы**: Знакомство с системой Docker.

# Задачи

В практической работе необходимо выполнить все шаги из разделов 1–7. В отчёт должны быть включены ответы на вопросы, выделенные курсивом, результаты выполнения команд из разделов 1–7, а также выполненное индивидуальное задание (раздел 8): листинг Dockerfile, а также команды сборки и запуска контейнера.

Вариант 12

Необходимо использовать базовый образ ubuntu:20.10

Примонтировать файл data/student.txt как /mnt/files/student.txt в контейнере

Запустить веб-сервер, отображающий содержимое /mnt/files, в хостовой системе должен открываться на порту (8800 + номер варианта).Установить пакет, согласно варианту: 12. wget

# Выполнение

## Раздел 1

Посмотрим на имеющиеся образы, рисунок 1.



Рисунок 1 – Имеющиеся образы

Загрузим образ, рисунок 2.

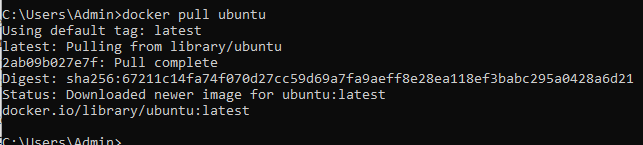


Рисунок 2 – Загрузка образа

Снова посмотрим образы, рисунок 3.



Рисунок 3 – Второй просмотр образов

Также выведем список контейнеров, убедимся, что там пока ничего нет, рисунок 4.



Рисунок 4 – Список контейнеров

## Раздел 2

Посмотрим информацию о хостовой системе. Выполним команду два раза, убедимся, что оба раза результат один и тот же. Представлено на рисунке 5.

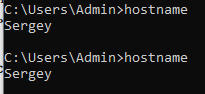


Рисунок 5 – Хостовая система

Выполним то же самое, но в контейнерах. Имеем разный результат. Представлено на рисунке 6.

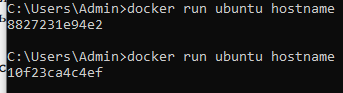


Рисунок 6 – Хостовая система в контейнерах

Снова посмотрим список запущенных контейнеров. На рисунке 7 выведен список. Видно, что было запущено два разных контейнера.

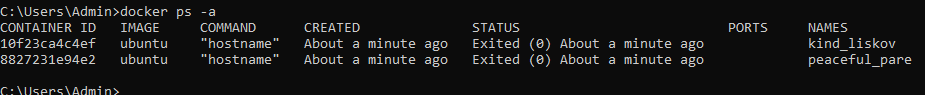


Рисунок 7 – Список контейнеров

Запустим bash. На рисунке 8 представлен результат. Результат оказался ожидаемым.



Рисунок 8 – Список контейнеров

Запустим bash правильно. Результат на рисунке 9.

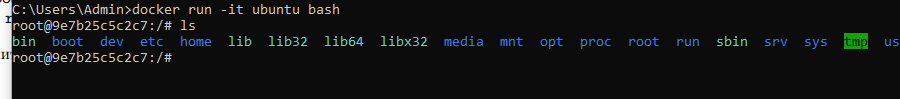


Рисунок 9 – Запущенный bash

## Раздел 3

Загрузим образ python. Результат на рисунке 10.

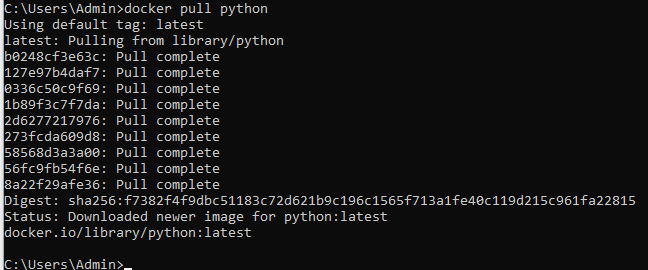


Рисунок 10 – Загрузка образа python

Запустим тестовый сервер, команда представлена на рисунке 11.



Рисунок 11 – Тестовый сервер

Однако, для работы требуется пробросить порты. Требуется запустить сервер с указанием портов. Команда представлена на рисунке 12.



Рисунок 12 – Тестовый сервер с портами

Окно тестового сервера представлено на рисунке 13.

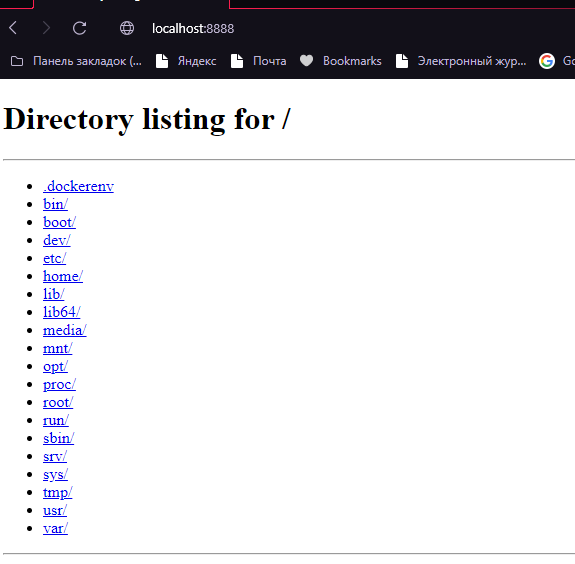


Рисунок 13 – Тестовый сервер

## Раздел 4

Запустим сервер в фоне. Рисунок 14.



Рисунок 14 – Запуск сервера в фоне

Убедимся, что контейнер работает. Рисунок 15. Как можем убедиться в столбце NAMES указано имя нашего контейнера pyserver.



Рисунок 15 – Активные контейнеры

Остановим, а затем удалим pyserver. Команды, а также проверка остановки представлены на рисунке 16.

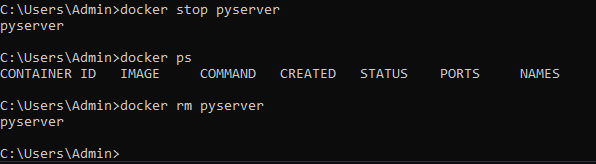


Рисунок 16 – Остановка и удаление pyserver

## Раздел 5

Запустим контейнер с сервером, который будет отдавать содержимое папки /mnt. Рисунок 17. –d /mnt означает, что модуль http.server будет имкть /mnt в качестве корневой папки отображения.



Рисунок 17 – Запуск сервера

В данной команде значение флагов запуска следующее:

-p8000:8000 – установка порта,

--name – установка имени контейнера,

--rm – пометка, означающая, что контейнер нужно удалить, после окончания его работы,

-d – запуск контейнера в фоне.

Попадем в запущенный контейнер и создадим файл hi.txt. Рисунок 18.

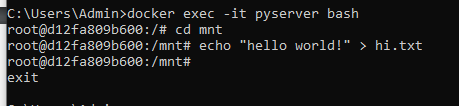


Рисунок 18 – Внутри контейнера

На рисунке 19 представлено содержание сервера после успешного создания файла hi.txt.

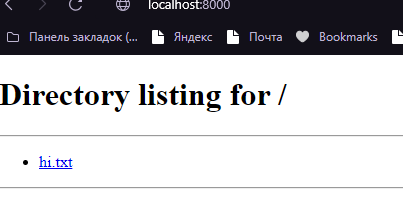


Рисунок 19 – На сервере

Остановим контейнер и создадим заново. Рисунок 20.

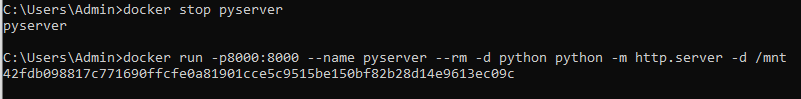


Рисунок 20 – Перезапуск контейнера

Посмотрим на сервер, hi.txt пропал. Рисунок 21.

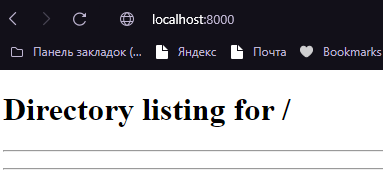


Рисунок 21 – Сервер после перезапуска

### Раздел 5.1

Создадим контейнер с примонтированным томом. Данные из тома myvolume в папке /mnt. Рисунок 22.



Рисунок 22 – Контейнер с томом

Снова создадим файл hi.txt. Рисунок 23.

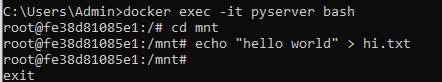


Рисунок 23 – Создание hi.txt

Узнаем, где хранятся данные. Рисунок 24. Путь показан в разделе “Source”.



Рисунок 24 – Ищем данные

### Раздел 5.2

На рисунке 25 представлена остановка предыдущего контейнера.



Рисунок 25 – Остановка контейнера

Рабочей папкой будет C:\\_\_My\_DATA\Sergey Khrechko\VUZrazrabPriloz\ThirdPrac. Создадим в ней папку mifiles, в ней создадим файл host.txt.

Далее запустим контейнер, путь до рабочей папки получим с помощью %cd%. Команда представлена на рисунке 26.



Рисунок 26 – Запуск контейнера

По отработанной схеме создадим файл hi.txt. Рисунок 27. Папка с файлами называется misfiles.

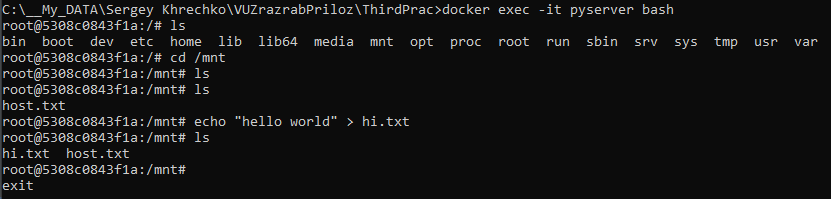


Рисунок 27 – Создание hi.txt

Можем убедиться в наличии файлов на сервере. Рисунок 28.

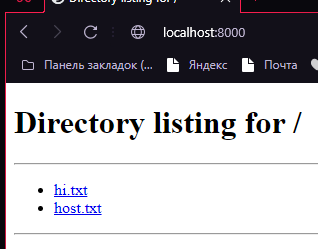


Рисунок 28 – На сервере

## Раздел 6

Создадим переменную среды. Рисунок 29.

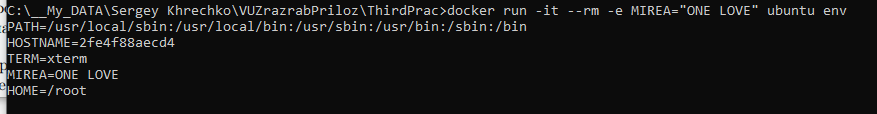


Рисунок 29 – Переменные среды

## Раздел 7

Создадим докер файл. Рисунок 30.

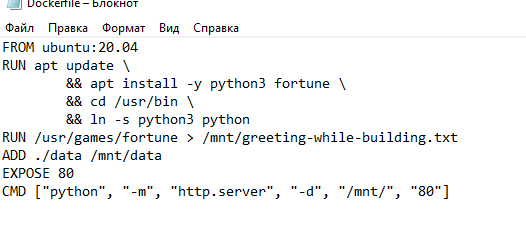


Рисунок 30 – Докер файл

Соберем образ командой “docker build -t mycoolimage .”. Рисунок 31.

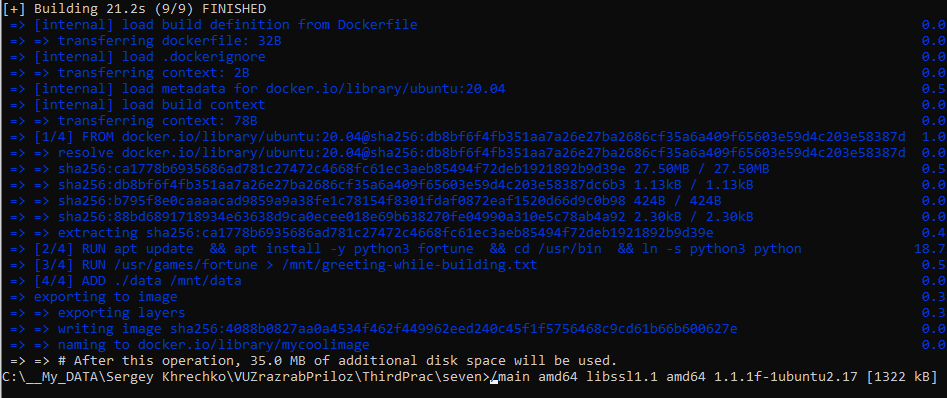


Рисунок 31 – Сборка

Запустим образ. Рисунок 32.

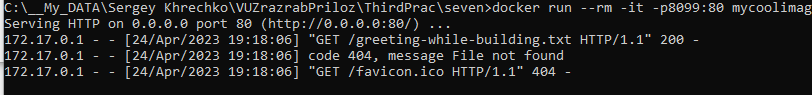


Рисунок 32 – Запуск

Проверим сгенерированный файл. Рисунок 32.

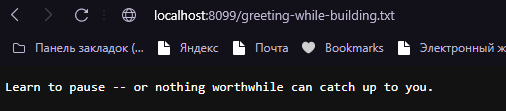


Рисунок 32 – Сгенерированный файл

## Раздел 8

Вариант 12.

Работа будет проходить в папке indiv. Создадим директорию data и в ней файл student.txt, заполним его. Рисунок 33.

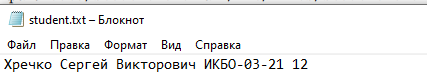


Рисунок 33 – Файл student.txt

Создадим Dockerfile, нам требуется образ ubuntu: 20.10, и установить пакет wget. Содержимое файла показано на рисунке 34.

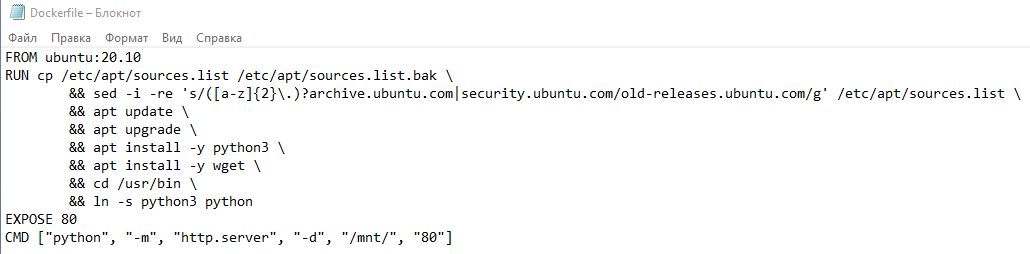


Рисунок 34 – Dockerfile

Сборка образа показана на рисунке 35. Использована команда «docker build –t indiv .»

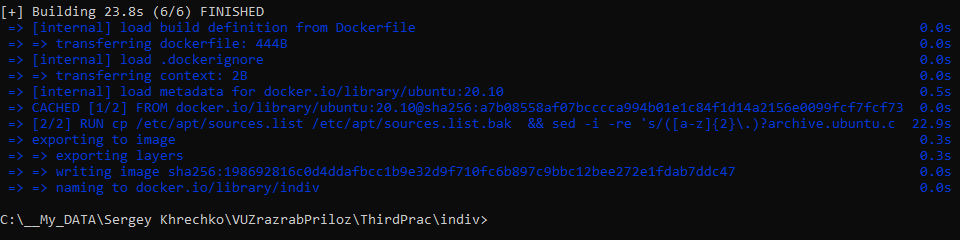


Рисунок 35 – Сборка

Теперь запишем команду для запуска, с монтированием и запуском на порте 8812. Команда представлена на рисунке 36.



Рисунок 36 – Команда запуска

Содержимое файла на сервере представлено на рисунке 37. Можно сделать вывод, что они крайне похожи.

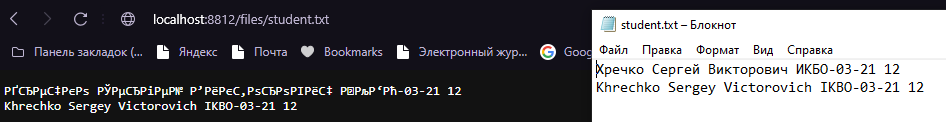


Рисунок 37 – На сервере

# Вывод

В процессе выполнения работы были получены навыки по установке docker. Были освоены базовые навыки работы с docker. Был создан и запущен свой образ.