Работа с Docker для Windows (Windows Server 2019, ver.1809)

1. Завести учетную запись на сайте docker.com (не обязательно для выполнения работы)
2. Подключиться к ВМ Microsoft Windows Server 2019 (назовем ее «host-dock»)
3. Зайти на сайт docker.com
4. Установить на хосте Docker Desktop
5. Создать рабочую папку на «host-dock», сделать ее текущей
6. Выполнить команды Docker на «host-dock» (в PowerShell)
    a. docker info
    b. docker version
    c. docker image ls (docker images)
    d. docker container ls (docker ps)
    e. docker network ls
7. Выполнить команду
    docker run hello-world
8. Создать веб-приложение ASP.NET Core (имя, напр., *Web17*) на ВМ с Visual Studio (назовем ее «host-VS»)
    a. ASP.NET Core Web Application (MVC)
    b. Персонифицировать домашнюю страницу
    c. Test on localhost
    d. Publish to Folder (например, Publish17)
9. Запустить на «host-VS» веб-приложение вне среды Visual Studio, выполнив команду
    dotnet run --project <путь_к_проекту>
10. Скопировать папку Publish17 в рабочую папку на «host-dock» под именем, напр., *Publish17*
11. Создать (в рабочей папке на хосте) Dockerfile

Содержимое Dockerfile (имена папки и приложения заменить на свои):

```
FROM mcr.microsoft.com/dotnet/core/aspnet:3.1-nanoserver-1809 AS base
FROM base AS final
WORKDIR /app
COPY publish17 .
ENTRYPOINT ["dotnet", "Web17.dll"]
```

(обратите внимание на «.» в команде COPY, она указывает на каталог в контейнере)

12. Создать свой образ (выполнение Dockerfile), выполнив в рабочей папке
    a. docker build -t webcore17-image .
    здесь имя образа, напр., *webcore17-image*
    (обратите внимание на «.»

b. docker image ls

13. Запустить контейнер из созданного образа, указав внешний порт, напр., *8000*

    a. docker run -it --name *webcore17-con* -p *8000*:80 *webcore17-image*

    здесь -p *8000*:80 означает отображение порта 80 контейнера на порт *8000* хоста,
    web webcore17-con – имя контейнера,
    webcore17-image – имя образа,
    -it означает
    -t Allocate a pseudo-tty
    -i Keep STDIN open even if not attached

    b. docker port <имя или ID контейнера> – просмотр портов

    c. docker exec <имя или ID контейнера> <command>
    – выполнение команды в контейнере, например:

    docker exec <имя или ID контейнера> *ipconfig* – просмотр IP-конфигурации (IP адрес).

14. Если ВМ с Docker – в облаке, необходимо открыть входящие запросы на порт *8000*

    a. В консоли Azure, ВМ, «Сетевые подключения» (Networking) – «Добавить правило входящего порта» (Add inbound port rule) – «Диапазоны портов назначения» (Destination port ranges), (можно дать правилу имя. Не забыть нажать кнопку «Добавить» (Add)

    b. В консоли AWS, ВМ, Security Group – Edit inbound rules – Add rule – Port Range, (можно дать правилу описание). Не забыть нажать кнопку «Save»

15. Проверить доступность веб-приложения в контейнере

    a. на хосте, в браузере задать IP адрес контейнера,

    b. на хосте, в браузере задать адрес *localhost:port* –внешний порт хоста с контейнером,

    c. в Интернете, в браузере задать адрес ВМ с Docker, порт – назначенный внешний порт хоста с контейнером.

16. Создать **новое** веб-приложение ASP.NET Core на ВМ с Visual Studio («host-VS»)

17. Опубликовать новое веб-приложение в **новую** папку, скопировать ее в рабочую папку на «host-dock», создать **новый** Dockerfile

18. Создать **новый** образ, запустить **новый** контейнер (отобразить порт 80 контейнера на **другой порт** хоста, например, 8001).

19. Проверить одновременное существование двух контейнеров, присутствие обоих веб-приложений на двух портах хоста, например:

    a. http://54.216.59.241:8080

    b. http://20.117.73.10:8081

20. Выполнить команды пинг из одного контейнера в другой, например:

    a. docker exec webappcore1-cont ping 172.120.205.41

    b. docker exec webappcore2-cont ping 172.120.197.77

21. Выполнить команды докер в разных командных окнах с входом в оболочки контейнеров, присвоив контейнерам имена, например, *core1* и *core2*

    a. docker run -it --rm --name *core1* mcr.microsoft.com/dotnet/core/aspnet cmd

    b. docker run -it --rm --name *core2* mcr.microsoft.com/dotnet/core/aspnet cmd

    (--rm *Automatically remove the container when it exits*)

22. Выполнить в двух контейнерах команды

- dir
- md <name 1>
- dir
- exit

- dir
- md <name 1>
- dir
- exit

Проверить наличие контейнеров до выполнения команд *exit* и их отсутствие после *exit*.