Лабораторная работа №2

Тема: Работа с Docker.

Цель: Познакомиться с возможностями и получить практические навыки работы с Docker.

Выполнил: Барановский Г.В. гр.253502

Задания:

1. Подготовьте рабочее окружение в соответствии с типом вашей операционной системы
    - Установите Docker
    - Выполните базовую настройку

```
PS C:\WINDOWS\system32> docker --version
Docker version 25.0.3, build 4debf41
```

```
PS D:\d0cker\multi-container-app> docker info
Client:
 Version:    25.0.3
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.12.1-desktop.4
    Path:     C:\Program Files\Docker\cli-plugins\docker-buildx.exe
  compose: Docker Compose (Docker Inc.)
    Version:  v2.24.6-desktop.1
    Path:     C:\Program Files\Docker\cli-plugins\docker-compose.exe
  debug: Get a shell into any image or container. (Docker Inc.)
    Version:  0.0.24
    Path:     C:\Program Files\Docker\cli-plugins\docker-debug.exe
  dev: Docker Dev Environments (Docker Inc.)
    Version:  v0.1.0
    Path:     C:\Program Files\Docker\cli-plugins\docker-dev.exe
  extension: Manages Docker extensions (Docker Inc.)
    Version:  v0.2.22
    Path:     C:\Program Files\Docker\cli-plugins\docker-extension.exe
  feedback: Provide feedback, right in your terminal! (Docker Inc.)
    Version:  v1.0.4
    Path:     C:\Program Files\Docker\cli-plugins\docker-feedback.exe
  init: Creates Docker-related starter files for your project (Docker Inc.)
    Version:  v1.0.1
    Path:     C:\Program Files\Docker\cli-plugins\docker-init.exe
  sbom: View the packaged-based Software Bill Of Materials (SBOM) for an image (Anchore Inc.)
    Version:  0.6.0
    Path:     C:\Program Files\Docker\cli-plugins\docker-sbom.exe
  scout: Docker Scout (Docker Inc.)
    Version:  v1.5.0
    Path:     C:\Program Files\Docker\cli-plugins\docker-scout.exe
```

2. Изучите простейшие консольные команды и возможности Docker Desktop (см. лекцию), создать собственный контейнер docker/getting-started, открыть в браузере и изучить tutorial

```
PS C:\WINDOWS\system32> docker run -d -p 80:80 docker/getting-started
Unable to find image 'docker/getting-started:latest' locally
latest: Pulling from docker/getting-started
c158987b0551: Pull complete
1e35f6679fab: Pull complete
cb9626c74200: Pull complete
b6334b6ace34: Pull complete
f1d1c9928c82: Pull complete
9b6f639ec6ea: Pull complete
ee68d3549ec8: Pull complete
33e0cbbb4673: Pull complete
4f7e34c2de10: Pull complete
Digest: sha256:d79336f4812b6547a53e735480dde67f8f8f7071b414fbd9297609ffb989abc1
Status: Downloaded newer image for docker/getting-started:latest
85df002d84c70a3c2f68413d92cc89a665dda3a3653eb502c9840283d3219dbd
```

```
PS C:\WINDOWS\system32> docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED              STATUS              PORTS
      NAMES
85df002d84c7   docker/getting-started   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:80->80
/tcp   awesome_albattani
```

3. Создайте docker image, который запускает скрипт с использованием функций из https://github.com/smartiqaorg/geometric_lib.
   Запускаемый скрипт main.py:

```python
import os
import circle
import square

env = float(os.getenv('ARG1'))

print("Circle area=", circle.area(env))
print("Circle perimeter=", circle.perimeter(env))
print("Square area=", square.area(env))
print("Square perimeter=", square.perimeter(env))

```

   Dockerfile:

```
FROM python:3
WORKDIR /usr/src/app

COPY circle.py .
COPY square.py .
COPY main.py .

ENV ARG1=5

CMD [ "python", "./main.py"]
```

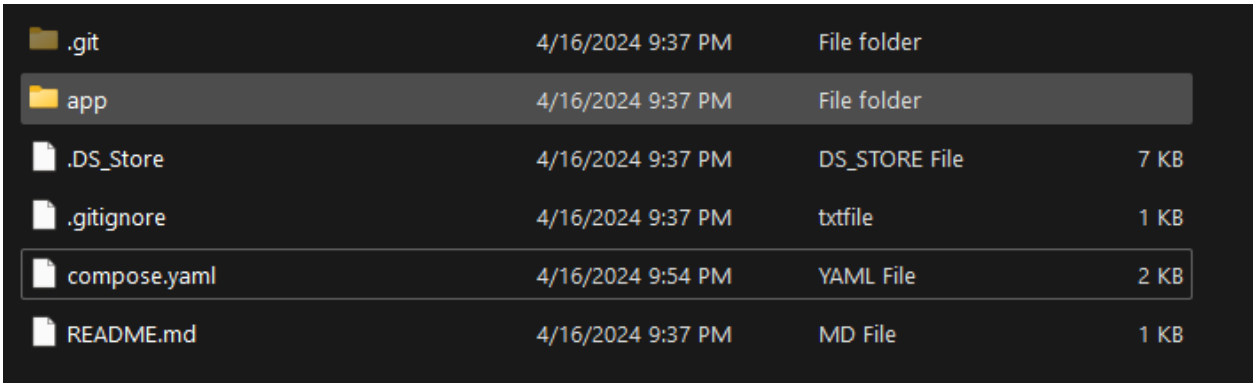
```
PS D:\d0cker\geometric_lib> docker build -t py1:v3 .
[+] Building 1.3s (10/10) FINISHED                                                                                  docker:default
 => [internal] load build definition from dockerfile                                                                          0.0s
 => => transferring dockerfile: 174B                                                                                          0.0s
 => [internal] load metadata for docker.io/library/python:3                                                                   1.0s
 => [internal] load .dockerignore                                                                                             0.0s
 => => transferring context: 2B                                                                                               0.0s
 => [1/5] FROM docker.io/library/python:3@sha256:e0e2713ebf0f7b114b8bf9fbcaba9a69ef80e996b9bb3fa5837e42c779dcdc0f             0.0s
 => [internal] load build context                                                                                             0.0s
 => => transferring context: 357B                                                                                             0.0s
 => CACHED [2/5] WORKDIR /usr/src/app                                                                                         0.0s
 => CACHED [3/5] COPY circle.py .                                                                                             0.0s
 => CACHED [4/5] COPY square.py .                                                                                             0.0s
 => [5/5] COPY main.py .                                                                                                      0.0s
 => exporting to image                                                                                                        0.1s
 => => exporting layers                                                                                                       0.0s
 => => writing image sha256:5c0cb7dc6e6b302bd76212c77c8d083ac44079389de54badb96cb980e722d55e                                  0.0s
 => => naming to docker.io/library/py1:v3                                                                                     0.0s

View build details: docker-desktop://dashboard/build/default/default/yw0wgzevvzfo6rm2ub375tsg5
```

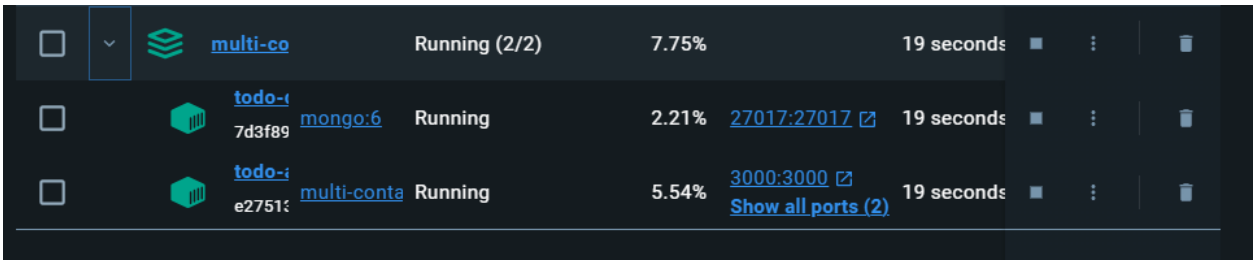
```
PS D:\d0cker\geometric_lib> docker run py1:v3
Circle area= 78.53981633974483
Circle perimeter= 31.41592653589793
Square area= 25.0
Square perimeter= 20.0
PS D:\d0cker\geometric_lib> docker run -e ARG1=10 py1:v3
Circle area= 314.1592653589793
Circle perimeter= 62.83185307179586
Square area= 100.0
Square perimeter= 40.0
```

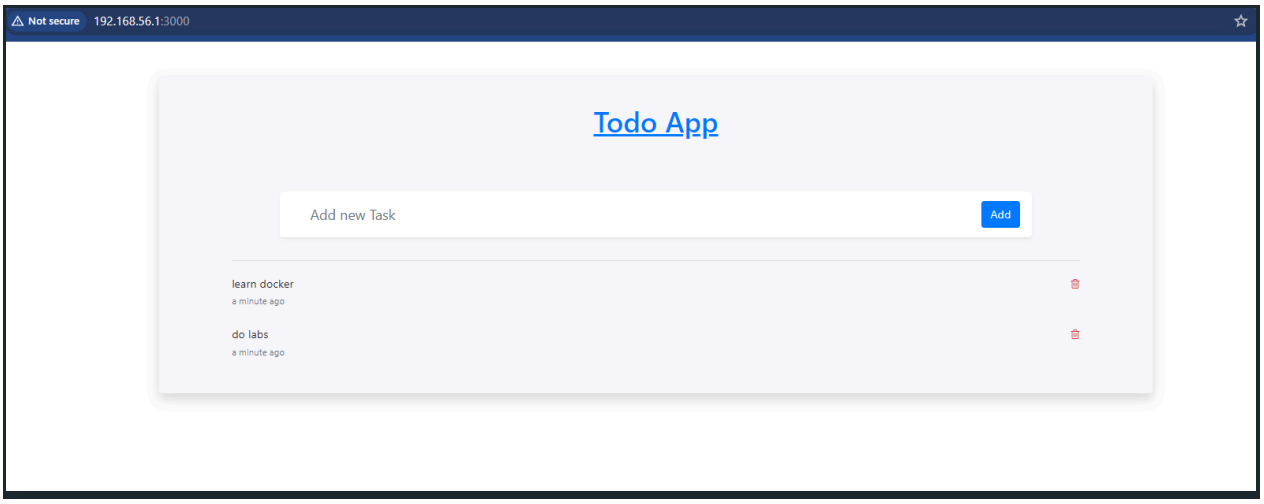
4. Скачать любой доступный проект с GitHub с произвольным стеком технологий (пример – см. индивидуальное задание) или использовать свой, ранее разработанный. Создать для него необходимый контейнер, используя Docker Compose для управления многоконтейнерными приложениями. Запустить проект в контейнере.(Примеры Images: https://hub.docker.com/_/phpmyadmin, https://hub.docker.com/_/mysql, https://hub.docker.com/_/postgres)

   Скачанный проект:



https://github.com/docker/multi-container-app





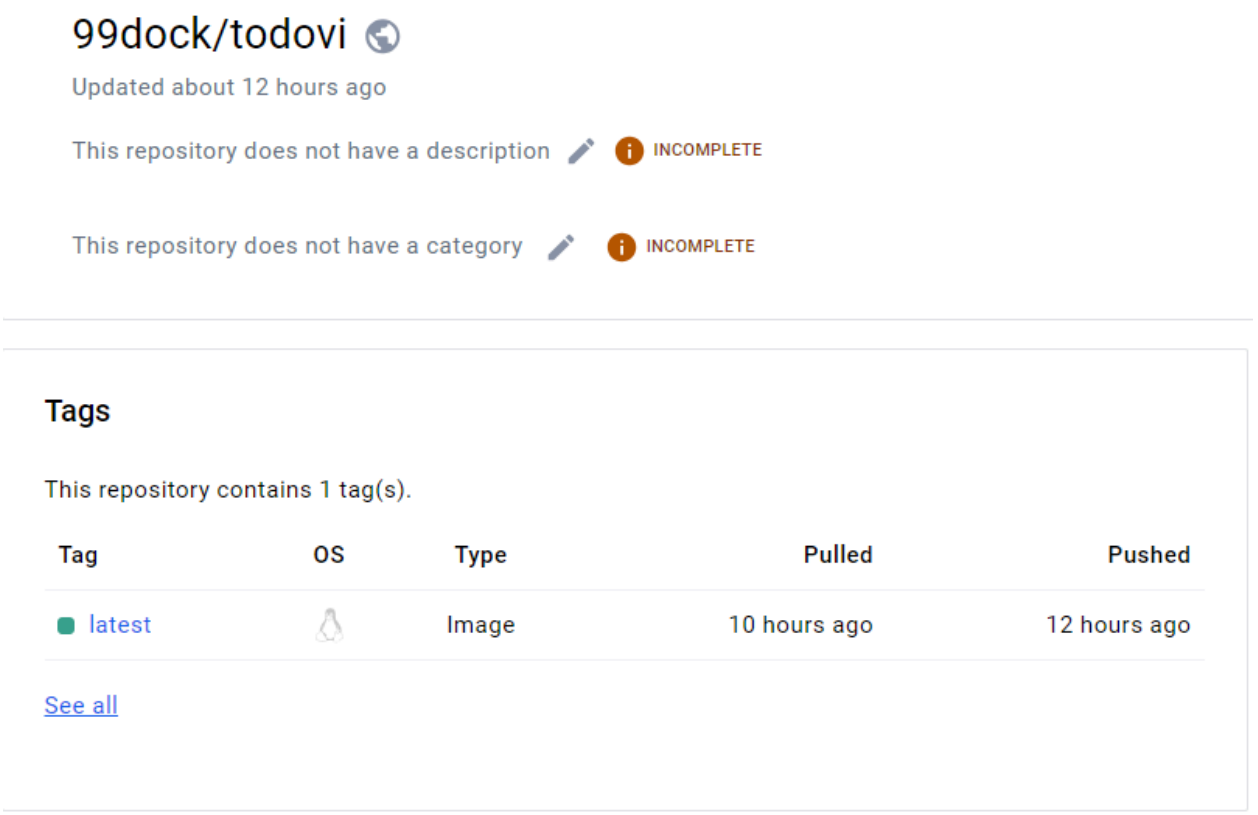
5. Настроить сети и тома для обеспечения связи между контейнерами и сохранения данных (исходные данные, логин, пароль и т.д.)

```
todo-database:
  image: mongo:6
  volumes:
    - database:/data/db
```

```
volumes:
  database:
```

6. Разместите результат в созданный репозиторий в DockerHub

```
PS D:\d0cker\multi-container-app> docker push 99dock/todovi
Using default tag: latest
The push refers to repository [docker.io/99dock/todovi]
adc446f530a6: Layer already exists
6bd95e4d6298: Layer already exists
643225c8f183: Layer already exists
927190f71e07: Layer already exists
195da39503d2: Layer already exists
07248e5314ee: Layer already exists
683a2ed728dc: Layer already exists
423f68dd19dc: Layer already exists
8e012198eea1: Layer already exists
latest: digest: sha256:d2b16f86ee3b56d1d064ac1f69ae98fb01a56320ed87bf4c6324eb48483e2559 size: 2207
```



7. Выполните следующие действия с целью изучить особенности сетевого взаимодействия:
    - Получить информацию о всех сетях, работающих на текущем хосте и подробности о каждом типе сети

```
PS D:\d0cker\multi-container-app> docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
96e41e7bc81e   bridge    bridge    local
f6306eca0a43   host      host      local
2a18ffa62f7a   myNet01   bridge    local
2cd8ae225732   none      null      local
```

```
PS D:\d0cker\multi-container-app> docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "96e41e7bc81e90cdaa9d751e8951d49eb8c02aced804c7266242ac9925f1d
        "Created": "2024-04-16T18:14:28.75596317Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        },
        "Labels": {}
    }
```

```
PS D:\d0cker\multi-container-app> docker network inspect host
[
    {
        "Name": "host",
        "Id": "f6306eca0a434bcf8f4a3b14fc51325db816aeabae14fdfdec632d2574a8a7e4",
        "Created": "2024-04-02T17:38:37.989126073Z",
        "Scope": "local",
        "Driver": "host",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": null
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

```
PS D:\d0cker\multi-container-app> docker network inspect none
[
    {
        "Name": "none",
        "Id": "2cd8ae225732f12498cfc479a32957ef25823fee125b2b3b6503a63c357c1c6b",
        "Created": "2024-04-02T17:38:37.946616262Z",
        "Scope": "local",
        "Driver": "null",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": null
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

```
PS D:\d0cker\multi-container-app> docker network inspect myNet01
[
    {
        "Name": "myNet01",
        "Id": "2a18ffa62f7a85f925fb59fc6009ab8be1a97c401284d18cdb974f9dc0c2053b",
        "Created": "2024-04-04T22:47:10.324326587Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

- Создать свою собственную сеть bridge, проверить, создана ли она, запустить Docker-контейнер в созданной сети, вывести о ней всю информацию(включая IP-адрес контейнера), отключить сеть от контейнера

```
PS D:\d0cker> docker network create testNet
dee00d3cdba723422210e90e6b0f1fe5c89b6874abfb9e7d3765e2845d2e9bb4
```

```
PS D:\d0cker> docker run -it -d --network=testNet -p 80:80 nginx
b2a353e4d8b6abeb0e0c2487d633eea2c25026c2fef44e9f327b5f9a146fffce
```

```
    "Containers": {
        "b2a353e4d8b6abeb0e0c2487d633eea2c25026c2fef44e9f327b5f9a146fffce": {
            "Name": "zealous_cray",
            "EndpointID": "db36cd68f9375e569b8f08299b21730baec3aae9fa0472f106ce400a0681c33e",
            "MacAddress": "02:42:ac:13:00:02",
            "IPv4Address": "172.19.0.2/16",
            "IPv6Address": ""
        }
    },
```

```
PS D:\d0cker> docker network disconnect testNet zealous_cray
PS D:\d0cker> docker network inspect testNet
[
    {
        "Name": "testNet",
        "Id": "dee00d3cdba723422210e90e6b0f1fe5c89b6874abfb9e7d3765e2845d2e9bb4",
        "Created": "2024-04-17T08:14:35.289108198Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.19.0.0/16",
                    "Gateway": "172.19.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

- Создать еще одну сеть bridge, вывести о ней всю информацию, запустить в ней три контейнера, подключиться к любому из контейнеров и пропинговать два других из оболочки контейнера, убедиться, что между контейнерами происходит общение по IP-адресу

```
PS C:\WINDOWS\system32> docker run --rm -it --name A1 --network otherNet -d nicolaka/netshoot /bin/bash
ddeed346862884ff895d056b981c55628240a3a3a7302e75f3d6b6f336059c61
PS C:\WINDOWS\system32> docker ps
CONTAINER ID   IMAGE              COMMAND       CREATED          STATUS          PORTS     NAMES
ddeed3468628   nicolaka/netshoot   "/bin/bash"   38 seconds ago   Up 36 seconds             A1
PS C:\WINDOWS\system32> docker run --rm -it --name A2 --network otherNet -d nicolaka/netshoot /bin/bash
70b770aaa94a6dd6194e3c00c50a658e9cfe9abc62f417617a815f74255ebc32
PS C:\WINDOWS\system32> docker run --rm -it --name A3 --network otherNet -d nicolaka/netshoot /bin/bash
0e31b2ffe5a64c4ed0291b26b94e778a16ab901308bfb85219c141c063f3d7f9
```

```
PS C:\WINDOWS\system32> docker exec -it A1 /bin/bash
ddeed3468628:~# ping A2
PING A2 (172.20.0.3) 56(84) bytes of data.
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=1 ttl=64 time=1.81 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=2 ttl=64 time=0.086 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=3 ttl=64 time=0.091 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=4 ttl=64 time=0.151 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=5 ttl=64 time=0.104 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=6 ttl=64 time=0.112 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=7 ttl=64 time=0.106 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=8 ttl=64 time=0.114 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=9 ttl=64 time=0.081 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=10 ttl=64 time=0.110 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=11 ttl=64 time=0.087 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=12 ttl=64 time=0.115 ms
64 bytes from A2.otherNet (172.20.0.3): icmp_seq=13 ttl=64 time=0.093 ms
^C
--- A2 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 12402ms
rtt min/avg/max/mdev = 0.081/0.235/1.811/0.455 ms
ddeed3468628:~# ping A3
PING A3 (172.20.0.4) 56(84) bytes of data.
64 bytes from A3.otherNet (172.20.0.4): icmp_seq=1 ttl=64 time=3.05 ms
64 bytes from A3.otherNet (172.20.0.4): icmp_seq=2 ttl=64 time=0.112 ms
64 bytes from A3.otherNet (172.20.0.4): icmp_seq=3 ttl=64 time=0.110 ms
64 bytes from A3.otherNet (172.20.0.4): icmp_seq=4 ttl=64 time=0.117 ms
^C
--- A3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3091ms
```

- Создать свою собственную сеть overlay, проверить, создана ли она, вывести о ней всю информацию

```
PS C:\WINDOWS\system32> docker network create --driver overlay overlayNet
gbzjv5bae6bxf1hdnhnlz9ty0
PS C:\WINDOWS\system32> docker network inspect overlayNet
[
    {
        "Name": "overlayNet",
        "Id": "gbzjv5bae6bxf1hdnhnlz9ty0",
        "Created": "2024-04-17T09:13:12.868934318Z",
        "Scope": "swarm",
        "Driver": "overlay",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "10.0.1.0/24",
                    "Gateway": "10.0.1.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": null,
        "Options": {
            "com.docker.network.driver.overlay.vxlanid_list": "4097"
        },
        "Labels": null
    }
```

- Создать еще одну сеть overlay, проверить, создана ли она, вывести о ней всю информацию, удалить сеть

```
PS C:\WINDOWS\system32> docker network create --driver overlay overlayNet2
q0s5kez54glslew7p8s859ey2
```

```
PS C:\WINDOWS\system32> docker network inspect overlayNet2
[
    {
        "Name": "overlayNet2",
        "Id": "q0s5kez54glslew7p8s859ey2",
        "Created": "2024-04-17T09:15:20.351385323Z",
        "Scope": "swarm",
        "Driver": "overlay",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "10.0.2.0/24",
                    "Gateway": "10.0.2.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": null,
        "Options": {
            "com.docker.network.driver.overlay.vxlanid_list": "4098"
        },
        "Labels": null
    }
]
```

- Попробовать создать сеть host, сохранить результат в отчет.

```
PS C:\WINDOWS\system32> docker network create --driver host hostNet
Error response from daemon: only one instance of "host" network is allowed
```