Установка

1. Обновляем все

```
sudo apt update
sudo apt upgrade
```

2. Устанавливаем зависимости

```
sudo apt install apt-transport-https ca-certificates curl
software-properties-common
```

3. Добавляем репозиторий

   Сначала gpg ключ

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo
gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
```

   Потом сам репозиторий

```
echo "deb [arch=$(dpkg --print-architecture)
signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs)
stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
```

4. Снова обновляем

```
sudo apt update
```

5. И устанавливаем

```
sudo apt install docker-ce docker-ce-cli containerd.io
```

6. Проверяем

```
sudo systemctl status docker
```

Чтобы выполнять команды без sudo создаем группу и добавляем своего юзера в группу Docker

```
sudo groupadd docker
sudo usermod -aG docker $USER
```

После того как добавили пользователя надо перезагрузиться

Первый контейнер. (1 балл)

Теперь командой

`docker run -d -p 80:80 --name my_nginx nginx:alpine` - запустим очень легкий контейнер с nginx тут run означает создание контейнера из образа, если его нет то docker его скачивает
-d отсоединяем его от терминала, то есть после запуска мы сможем вводить какие-то другие команды в терминале.
-p 80:80 - проброс портов
--name my_nginx - название которое мы даем (если не дадим он сам даст)
nginx:alpine - название образа из которого создаем контейнер.

Теперь можно увидеть результат, то есть что nginx работает

`curl http://localhost` - в терминале

или так

http://localhost в браузере

Это и считается результатом выполнения.

2. Создаем свой образ и запускаем из него контейнер (3 балла)

Создаем папку с любым названием например Docker_sem

В ней создаем текстовый файл в который копируем этот текст:

```
# Используем официальный образ Alpine Linux в качестве базового
FROM alpine:latest

# Устанавливаем простой веб-сервер (в данном случае, BusyBox HTTPD)
RUN apk --no-cache add busybox-extras

# Копируем файл index.html внутрь контейнера
COPY index.html /var/www/

# Задаем рабочий каталог
WORKDIR /var/www

# Определяем, какой порт будет использоваться для HTTP
EXPOSE 80

# Команда для запуска веб-сервера при старте контейнера
CMD ["httpd", "-f", "-p", "80"]
```

Сохраняем файл назвав его Dockerfile - без расширений.

Затем там же создаем файл index.html

И пишем там что-то типа:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>first image</title>
</head>
<html>
<body>
   <h1>Это мой первый контейнер сделанный из собственноручно сделанного образа!</h1>
</body>
</html>
```

Теперь из той папки где у вас Dockerfile и index.html

```
docker build -t my-first-image .
```

Где -t указывает на название которое идет после него в этом примере название - my-first-image, а . указывает на контекст, то есть все что нужно лежит в текущем каталоге.

И запускаете командой

```
docker run -d -p 8080:80 my-first-image
```

На localhost должны увидеть то что вы написали в файле index.html

Это и считается выполненным заданием.