**Задача 1.1**

В этом задании надо последовательно ответить на несколько вопросов:

1. Чем контейнер отличается от образа?

- Образ Docker – это доступный только для чтения шаблон, который задает настройки контейнера. Образ содержит исполняемый код вместе с определениями любых библиотек и зависимостей, которые могут понадобиться для его работы. Контейнер Docker представляет собой запущенный (работающий) образ Docker.

Образы — шаблоны с инструкциями только для чтения для создания контейнеров.

Контейнеры — запущенные экземпляры образов.

2. Сколько контейнеров можно сделать от одного образа?

- Сколько угодно, сколько хватит места на жёстком диске.

3. Для чего используется ключ -p в команде docker run

- По умолчанию для доступа к процессам нужно войти внутрь контейнера. Однако можно подключиться к ним извне, открыв порты. Можно сопоставить порты локальной машины с портами контейнера, чтобы передавать между ними данные. Для этого используется параметр -p

4. Почему не бывает образов с операционной системой Windows и MacOS внутри?

- Потому что Windows и MacOs требуют лицензионного использования, в то время как Linux системы находятся в свободном доступе.

5. Где находится информация, которая хранится в томе (docker volume)?

- Информация вместе с томом хранится на хосте, которым может выступать локальная машина (та, которую я сейчас использую, например).

6. Как запустить контейнер так, чтобы он видел содержимое какой-то директории на хост-машине, но не мог ее изменить?

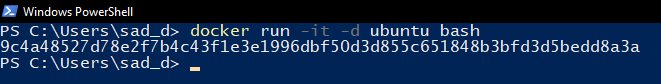
- Нужно запустить контейнер с параметром монтирования readonly

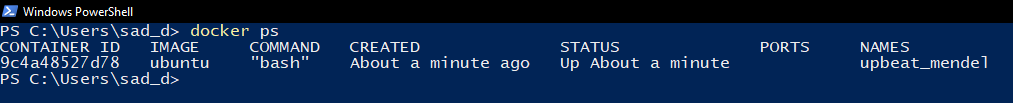
- docker run -it --mount type=bind,src=D:/LearnQA\_Docker/dir\_for\_bind,target=/bind/,readonly ubuntu bash

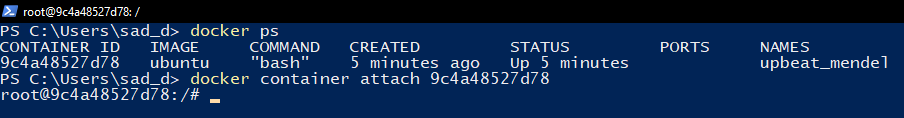
**Задача 1.2**

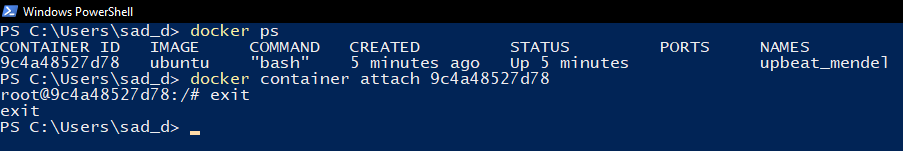
В этой задаче надо выполнить несколько последовательных шагов:

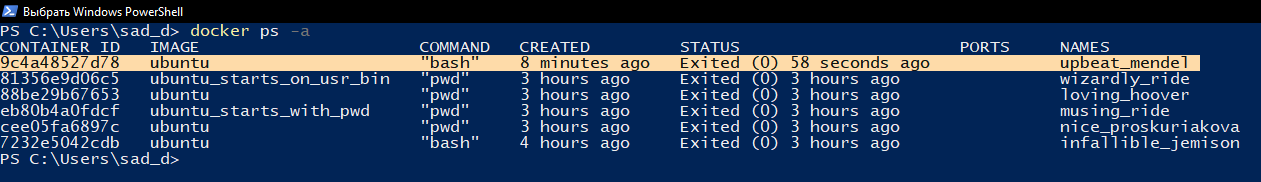
1. Запустить контейнер Ubuntu с detached флагом.

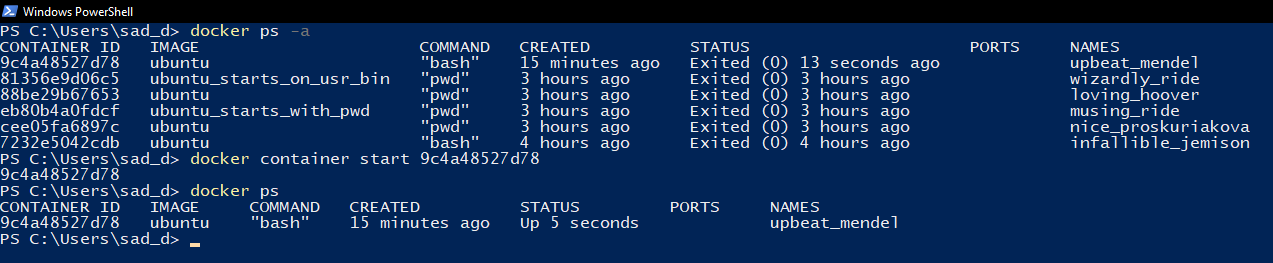
2. Найти запущенный контейнер в списке контейнеров.

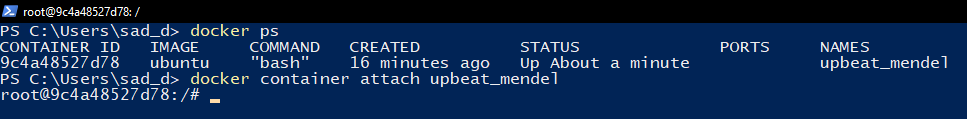
3. Подключиться к нему.

4. Выйти из него командой exit (тем самым остановив контейнер).

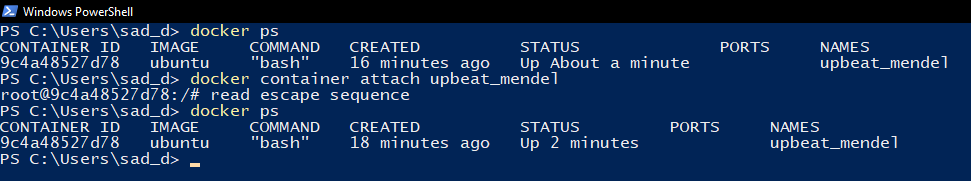
5. Снова найти контейнер в списке контейнеров.

6. Запустить его.

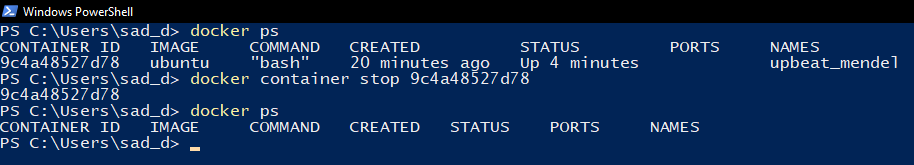
7. Подключиться к нему.

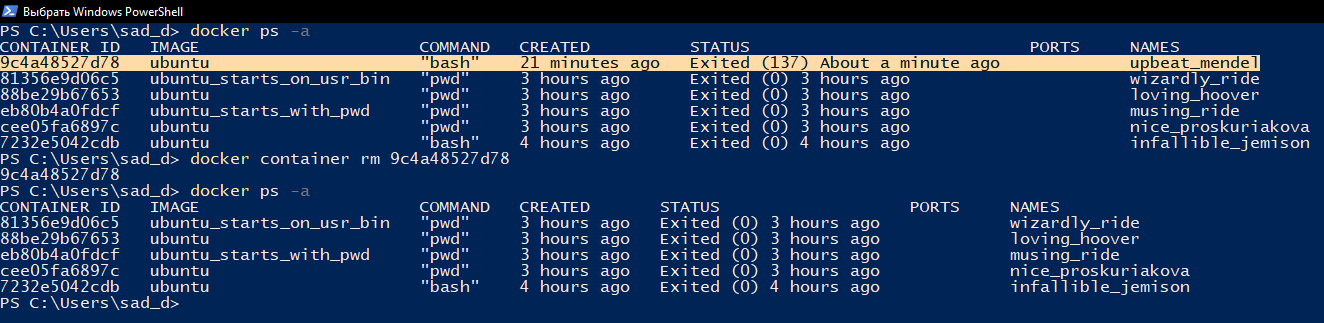
8. Выйти из контейнера, не остановив его.

- Нужно выполнить две комбинации клавиш: Ctrl+P, Ctrl+Q

9. Найти запущенный контейнер в списке. (скрин выше)

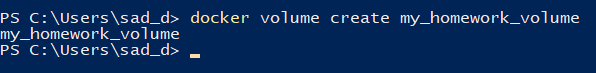
10. Остановить его.

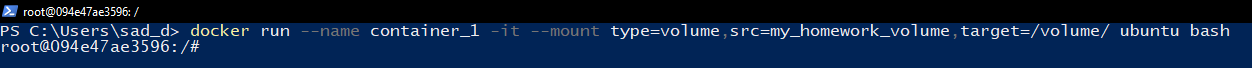
11. Удалить его из списка.

  
  
**Задача 1.3**

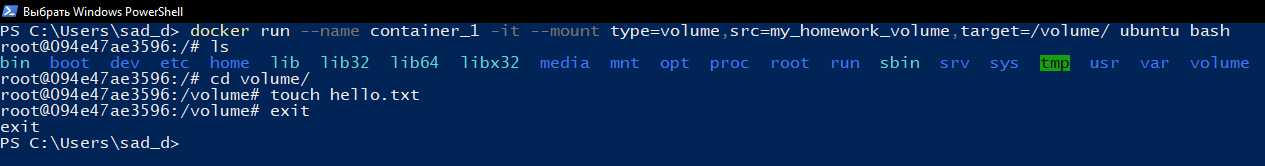
В этой задаче надо выполнить несколько последовательных шагов:

1. Создать том с названием my\_homework\_volume

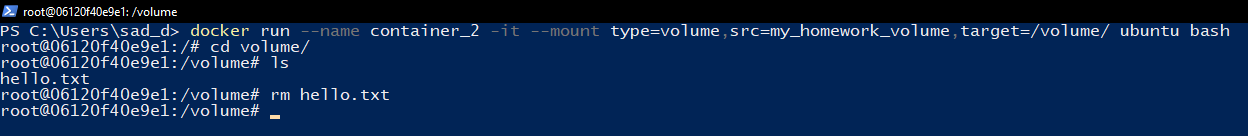
2. Запустить контейнер Ubuntu с названием container\_1 и подключенным к нему томом.

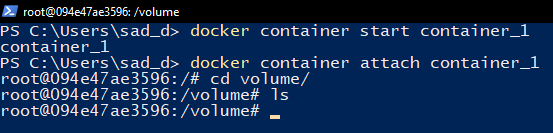
3. Оказавшись внутри контейнера container\_1, перейти в директорию тома и создать там файл hello.txt без содержимого. Для этого используйте команду touch [название файла].

4. Выйти из контейнера.

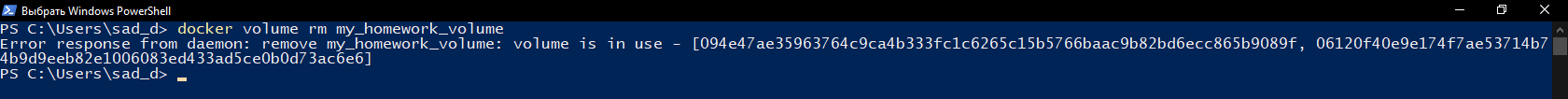
5. Создать второй контейнер Ubuntu с названием container\_2 и с тем же подключенным томом.

6. Оказавшись внутри контейнера container\_2, перейти в директорию тома и найти там файл hello.txt (команда ls). Удалить файл командой rm.

7. Выйти из контейнера container\_2 и вернуться в container\_1. Перейдя в директорию с томом при помощи команды ls проверить наличие созданного файла hello.txt.

8. Выйти из контейнера.

9. Попытаться удалить том, выполнив команду docker volume rm my\_homework\_volume

  
  
Результатом должны быть ответы на все три вопроса:

1. Удалось ли найти созданный файл hello.txt внутри container\_1 после того, как вы удалили файл внутри container\_2?

**- Нет.**

2. Какая ошибка возникла при попытке удалить том?

**- Ошибка сообщает, что том находится в использовании (примонтирован) в контейнере container\_1 и container\_2**

3. Что необходимо сделать, чтобы том был удален?

**- Необходимо отвязать том my\_homework\_volume от контейнеров.**

**Задача 1.4**

Есть код на языке PHP:

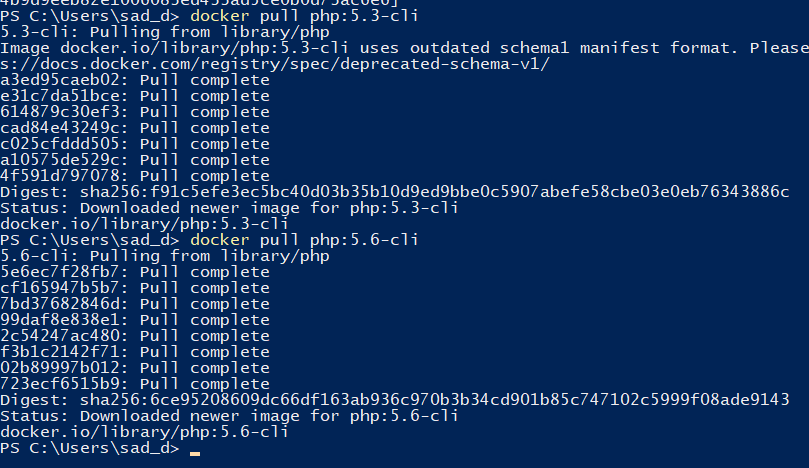
<?php foreach ([1,2,3] as $i) {echo $i . PHP\_EOL;}

Этот код необходимо скопировать и положить в файл counter.php

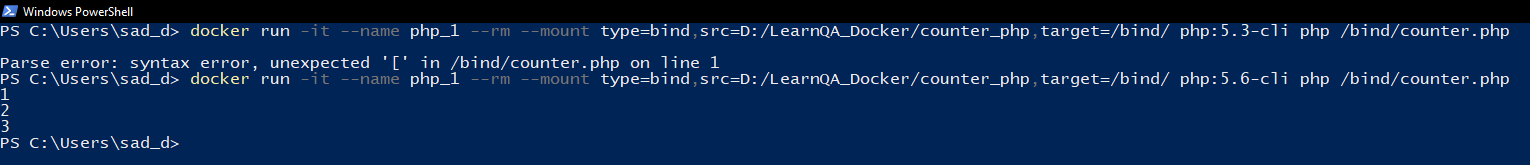
После чего необходимо скачать два образа:

docker pull php:5.3-cli

docker pull php:5.6-cli

  
После чего необходимо запустить программу на php в обоих контейнерах и проверить ее работоспособность.

Для одного из контейнеров программа сломается - это ожидаемо.

Результатом должны быть ответы на все три вопроса:

1. Какой тип mount выбрали - bind или volume? Почему?

**- bind. Потому что нужно примонтировать внешнюю папку на локальной машине.**

2. Что вывела программа, запущенная внутри контейнера версии 5.3?

Parse error: syntax error, unexpected '[' in /bind/counter.php on line 1

3. Что вывела программа, запущенная внутри контейнера версии 5.6?

1 2 3

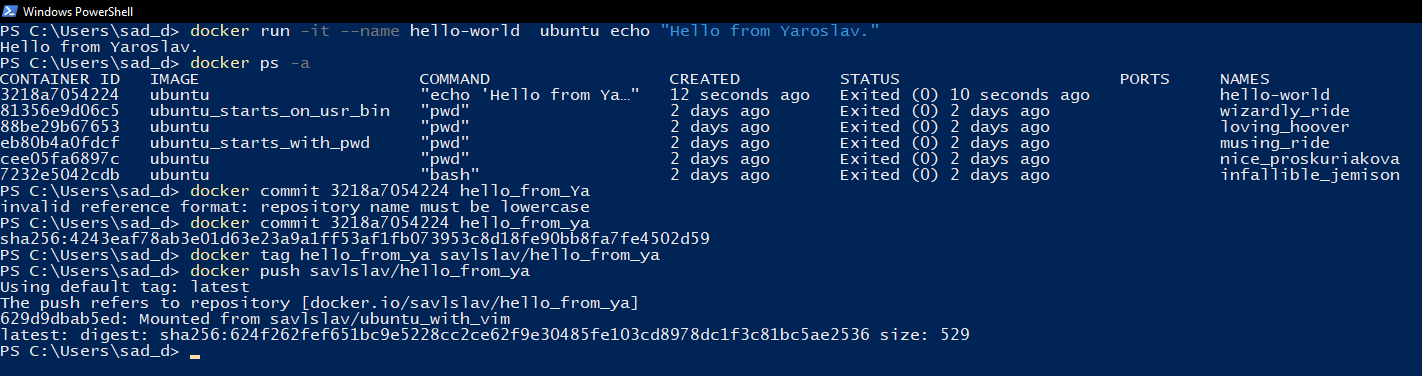
**Задача 1.5**

Необходимо сделать свой собственный hello-world образ и выложить его в свой репозиторий на Docker Hub.

При запуске контейнера должно писаться следующее:

Hello from [Ваше Имя]

После чего контейнер должен останавливаться.

  
Ссылка на Docker hub:

https://hub.docker.com/repository/docker/savlslav/hello\_from\_ya