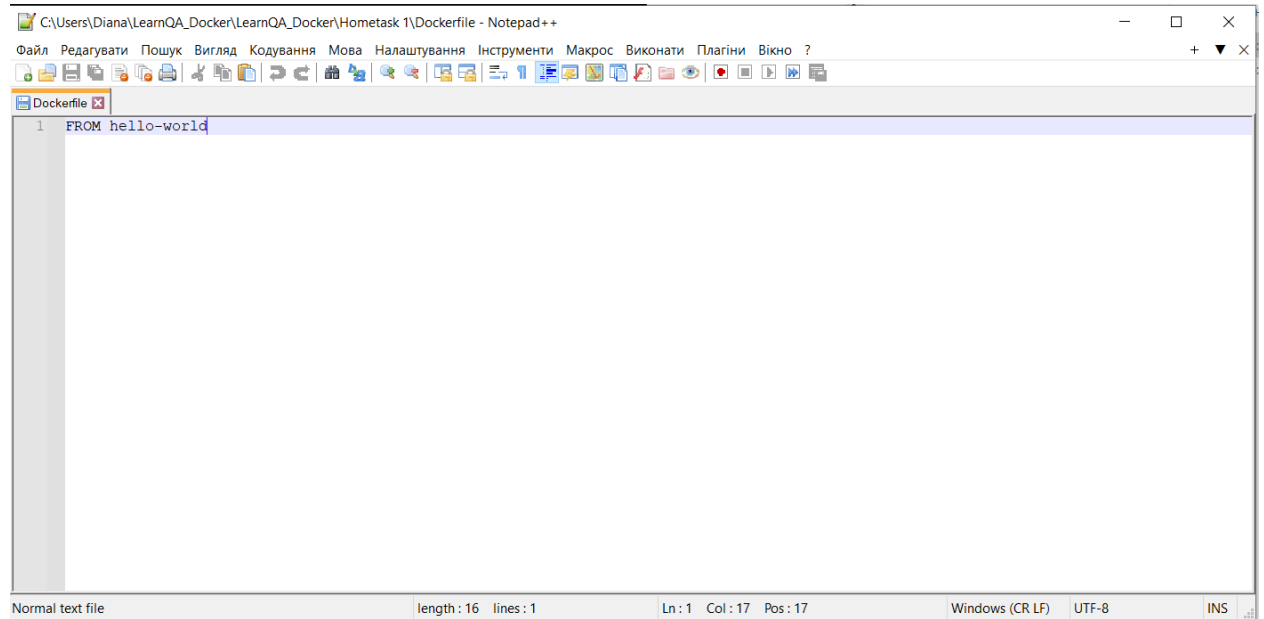


## Задача 2.1

Составить Dockerfile, который создает образ вашей hello-world программы, которую мы делали на предыдущем уроке.

Результат: содержимое Dockerfile



```
C:\Users\Diana\LearnQA_Docker\LearnQA_Docker\Hometask 1>docker build -t hello-world-2 .
[+] Building 0.1s (5/5) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 31B                                              0.0s
=> [internal] load .dockerignore                                                0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/hello-world:latest           0.0s
=> CACHED [1/1] FROM docker.io/library/hello-world                           0.0s
=> exporting to image                                                         0.0s
=> => exporting layers                                                         0.0s
=> => writing image sha256:022d83bd8967a4de69f03fdbc8a0adcd414c5fd9af793aaed25aeb632d162397 0.0s
=> => naming to docker.io/library/hello-world-2                               0.0s

C:\Users\Diana\LearnQA_Docker\LearnQA_Docker\Hometask 1>docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
<none>              <none>             98de91dab506       4 days ago         114MB
hello-world         latest             cc56deeb390a       4 days ago         77.8MB
hello-world-2       latest             022d83bd8967       4 days ago         77.8MB
name                Dockerfile          022d83bd8967       4 days ago         77.8MB
<none>              <none>             30ad9270d6ec       4 days ago         77.8MB
<none>              <none>             3ee62cd3f64b       4 days ago         77.8MB
run_test            latest             3e25ceb004ac       4 days ago         368MB
python_counter      latest             dc0d0c475834       4 days ago         922MB
ubuntu_with_curl    latest             068b24b1bc12       4 days ago         303MB
selenium/standalone-chrome  latest             812317d6b5b4       9 days ago         1.21GB
diankavoy/hello-world  latest             fa5c945e8b36       12 days ago        77.8MB
<none>              <none>             e62b75633721       12 days ago        77.8MB
ubuntu              latest             df5de72bdb3b       2 weeks ago        77.8MB
selenium/standalone-chrome  3.141.59           24e6c4e56d39       10 months ago      1.08GB
diankavoy/hello-world  <none>             feb5d9fea6a5       11 months ago      13.3kB
hello-world          <none>             feb5d9fea6a5       11 months ago      13.3kB
docker/dev-environments-default  stable-1           7c85b0303242       12 months ago      607MB
docker/desktop-git-helper  5a4fca126aadcd3f6cc3a011aa991de982ae7000 efe2d67c403b       12 months ago      44.2MB
maven                3.6.3-ibmjava-8    ad8f5e6bed87       17 months ago      368MB
php                   5.6-cli            36c3c974e6ee       3 years ago        344MB
php                   5.3-cli            7f2a82c91480       7 years ago        729MB

C:\Users\Diana\LearnQA_Docker\LearnQA_Docker\Hometask 1>docker run hello-world-2
'Hello from Diana'
```

## Задача 2.2

Умение искать необходимую информацию в интернете - бесценно.

У Dockerfile существует несколько инструкций (кроме FROM, COPY и т.д.), которые мы не осветили.

Необходимо самостоятельно найти список инструкций, выбрать одну и подробно описать для чего она необходима.

### **EXPOSE**

Ця команда допомагає в спілкуванні між контейнерами, вона зв'язує порт контейнера з хостом. Інструкція EXPOSE відкриває вказаний порт і робить його доступним лише для зв'язку між контейнерами.

Припустимо, у нас є два контейнери, програма nodejs і сервер redis. Наша програма node потребує зв'язку з сервером redis з кількох причин.

Щоб програма node могла зв'язуватися з сервером redis, контейнер redis має відкрити порт. Подивіться на Dockerfile офіційного образу Redis ([https://hub.docker.com/\\_/redis/](https://hub.docker.com/_/redis/)), і ви побачите рядок із написом EXPOSE 6379. Це те, що допомагає двом контейнерам спілкуватися один з одним.

Отже, коли ваш контейнер програми nodejs намагається підключитися до порту 6379 контейнера redis, інструкція EXPOSE робить це можливим.

Також інструкція EXPOSE повідомляє Docker, що контейнер прослуховує вказані мережеві порти під час виконання. Ви можете вказати, чи порт прослуховує TCP чи UDP, і за замовчуванням TCP, якщо протокол не вказано.

За замовчуванням EXPOSE передбачається TCP. Ми також можемо вказати UDP:

EXPOSE 80/udp

## Задача 2.3

В одном из домашних заданий прошлого урока мы тестировали программу counter.php для двух версий PHP.

В этом домашнем задании необходимо создать docker-compose файл, который последовательно запускает программу для обеих версий PHP и пишет результаты работы.

Напоминаем название контейнеров для PHP:

docker pull php:5.3-cli

`docker pull php:5.6-cli`

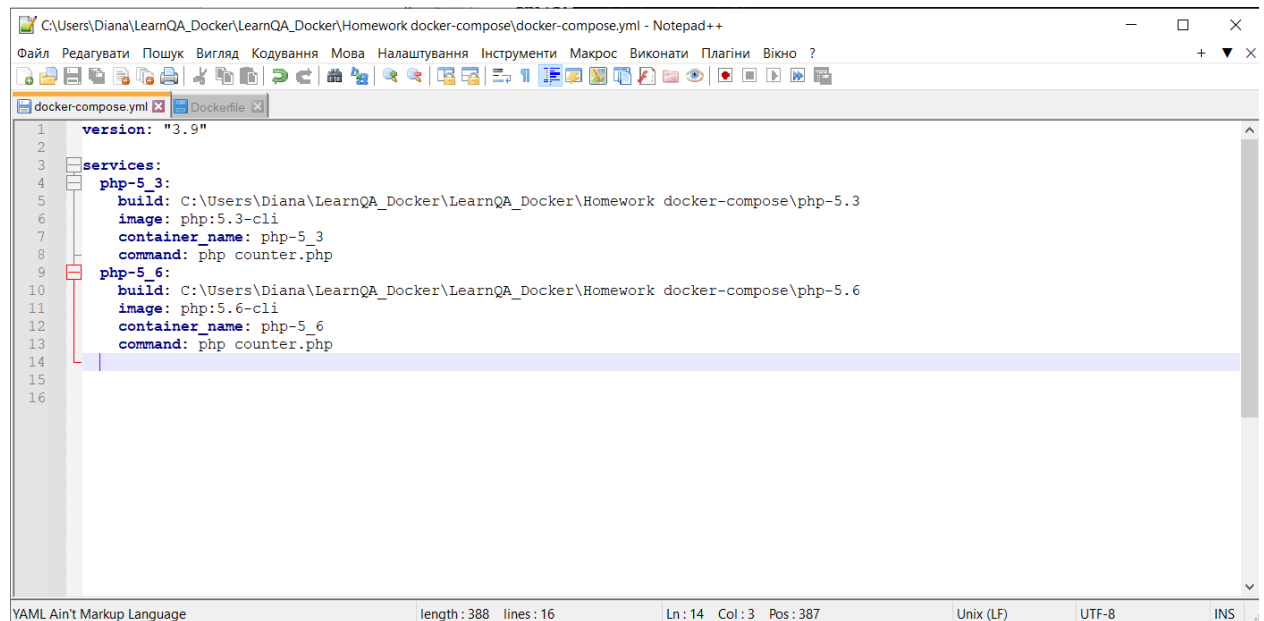
Напоминаем код программы:

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

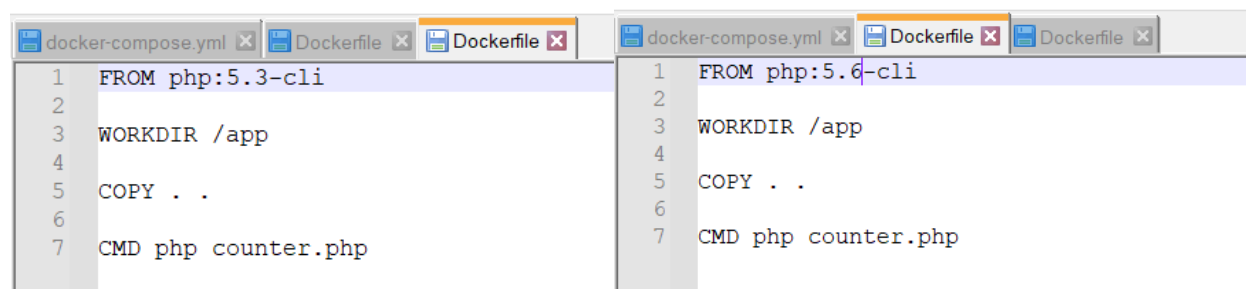
Этот код необходимо скопировать и положить в файл `counter.php`

Результат: содержимое `docker-compose.yml` файла,

запустив который можно последовательно запустить программу `counter.php` на обеих версиях языка (то есть для обоих образов).



```
1 version: "3.9"
2
3 services:
4   php-5.3:
5     build: C:\Users\Diana\LearnQA_Docker\LearnQA_Docker\Homework docker-compose\php-5.3
6     image: php:5.3-cli
7     container_name: php-5.3
8     command: php counter.php
9   php-5.6:
10    build: C:\Users\Diana\LearnQA_Docker\LearnQA_Docker\Homework docker-compose\php-5.6
11    image: php:5.6-cli
12    container_name: php-5.6
13    command: php counter.php
14
15
16
```













```
1 FROM php:5.3-cli
2
3 WORKDIR /app
4
5 COPY . .
6
7 CMD php counter.php
```

```
1 FROM php:5.6-cli
2
3 WORKDIR /app
4
5 COPY . .
6
7 CMD php counter.php
```

```

C:\Users\Diana\LearnQA_Docker\LearnQA_Docker\Homework docker-compose>docker-compose up --abort-on-container-exit --build
Building php-5_3
[+] Building 0.5s (8/8) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 31B                                              0.0s
=> [internal] load .dockerignore                                                0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/php:5.3-cli                  0.0s
=> [1/3] FROM docker.io/library/php:5.3-cli                                    0.2s
=> [internal] load build context                                                0.0s
=> => transferring context: 154B                                                0.0s
=> [2/3] WORKDIR /app                                                          0.0s
=> [3/3] COPY . .                                                              0.1s
=> exporting to image                                                          0.1s
=> => exporting layers                                                          0.1s
=> => writing image sha256:6d6e247c19f143f8c65d86b663984d3a3582ec82463bbeb279c311fc5719cf6e 0.0s
=> => naming to docker.io/library/php:5.3-cli                                  0.0s
Building php-5_6
[+] Building 0.4s (8/8) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 31B                                              0.0s
=> [internal] load .dockerignore                                                0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/php:5.6-cli                  0.0s
=> [1/3] FROM docker.io/library/php:5.6-cli                                    0.2s
=> [internal] load build context                                                0.0s
=> => transferring context: 154B                                                0.0s
=> [2/3] WORKDIR /app                                                          0.0s
=> [3/3] COPY . .                                                              0.0s
=> exporting to image                                                          0.1s
=> => exporting layers                                                          0.1s
=> => writing image sha256:6bcbefc687c28b02c2e1efeb53aaef87aff0927956c2801bab9728e20f6a9e2 0.0s
=> => naming to docker.io/library/php:5.6-cli                                  0.0s
Recreating php-5_3 ... done
Recreating php-5_6 ... done
Attaching to php-5_3, php-5_6
php-5_3 |
php-5_3 | Parse error: syntax error, unexpected '[' in /app/counter.php on line 1
php-5_6 | 1
php-5_6 | 2
php-5_6 | 3
php-5_3 exited with code 255
Aborting on container exit...
ERROR: 255

```

<input type="checkbox"/>		homeworkdocker-compos	2 containers		Exited	-	 Open 	
<input type="checkbox"/>		php-5_3	03f937d90c99 	php:5.3-cli	Exited (255)	-		
<input type="checkbox"/>		php-5_6	54d6d3763b48 	php:5.6-cli	Exited	-		

## Задача 2.4

Скачать последний образ Selenium Server:

`docker pull selenium/standalone-chrome:latest`

Запустить его так, чтобы открыв localhost в браузере, вы увидели стартовую страничку Selenium Server. На ней необходимо найти версию.

`docker run -d -p 4444:4444 selenium/standalone-chrome:latest`

Результат: указать текущую версию.

