Assignment 1

Part A

Explain the use of the following docker commands:

docker build

> *The docker build command builds Docker images from a Dockerfile and a "context". A build's context is the set of files located in the specified PATH or URL. The build process can refer to any of the files in the context. For example, your build can use a COPY instruction to reference a file in the context.*

`docker build` is used to build a docker image using a `Dockerfile`, which, analogous to a `Makefile`, contains a set of instructions for building the docker image.

A *docker image* is a template, a saved state of a docker that can have multiple containers running from it.

docker run

> *The docker run command first creates a writeable container layer over the specified image, and then starts it using the specified command. That is, docker run is equivalent to the API /containers/create then /containers/(id)/start. A stopped container can be restarted with all its previous changes intact using docker start. See docker ps -a to view a list of all containers.*

`docker run` is used to run a docker container from a docker image. A *docker container* is a running instance of a docker image.

Part B

Write a Dockerfile that will create a docker image (tagged my/python) using the most recent python image on "docker hub". Please make sure that port 8080 is exposed and that a command line is available upon starting the image in a container.

```
FROM python:latest      # Use the latest python image
EXPOSE 8080             # Expose port 8080
CMD ["/bin/bash"]       # Start a bash shell
```

To build the image, run the following command:

```
docker build -t my/python .
```

Part C

Write the docker command that will start a container named `python1` using the image tagged my/python (from part b). Ensure that the container has access to port 8080 and a folder on the host's file system (you are free to select any folder you wish, e.g. demo/python).

```
docker run -it --name python1 -p 8080:8080 -v
/home/mosaic/cmpt353/a1/docker:/home my/python
```

OPTION	DESCRIPTION
`-it`	Run the container in interactive mode
`--name python1`	Name the container `python1`
`-p 8080:8080`	Map port 8080 on the host to port 8080 on the container
`-v /home/mosaic/cmpt353/a1/docker:/home`	Mount the host folder `/home/mosaic/cmpt353/a1/docker` to the container folder `/home`
`my/python`	The image to run

Part D

Explain how a developer can use the container named python1 to develop and run a simple "hello world" python program.

The developer can use the container to develop and run a simple "hello world" python program by running the following commands:

```
cd /home
echo "print('Hello World')" > hello.py
python hello.py
```

in this case, the developer first changes the working directory to `/home`, then creates a file `hello.py` containing the python code to print "Hello World", and finally runs the python code in the file `hello.py`.

This is very simple example, if the developer wants to develop a more complex python program, they can open up text editor such as `vim` or `nano` and edit the file `hello.py` directly in the container.

Note that the editor need to be installed in the container first. We can do that by using `apt-get` to install the editor in the container, or include the command to install the editor in the `Dockerfile` and rebuild the image. `RUN ["apt-get", "update"]` and `RUN ["apt-get", "install", "-y", "vim"]` are two examples of how to install `vim` in the container. [1]

Also, the developer can use IDEs to develop the python program and run it in the container.

---

1. https://dev.to/greenteabiscuit/installing-vim-in-a-docker-container-15i6 ↩