Q1) Pull any image from the docker hub, create its container, and execute it showing the output.

Docker is a set of platforms as a service products that use of operating system level visualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries, and configuration files; they can communicate with each other through well-defined channels. All containers are run by a single operating system kernel and therefore use fewer resources than a virtual machine.

Docker version command:

```
C:\Users\HP\srinitha>docker version
Client:
 Cloud integration: v1.0.29
 Version:           20.10.22
 API version:       1.41
 Go version:        go1.18.9
 Git commit:        3a2c30b
 Built:             Thu Dec 15 22:36:18 2022
 OS/Arch:           windows/amd64
 Context:           default
 Experimental:      true

Server: Docker Desktop 4.16.3 (96739)
 Engine:
  Version:          20.10.22
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.18.9
  Git commit:       42c8b31
  Built:            Thu Dec 15 22:26:14 2022
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.14
  GitCommit:        9ba4b250366a5ddde94bb7c9d1def331423aa323
 runc:
  Version:          1.1.4
  GitCommit:        v1.1.4-0-g5fd4c4d
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0

C:\Users\HP\srinitha>
```

Step 1:

We can pull the image from the docker hub using docker pull command

Lets us download a image named redis from the docker hub. Later we get this output.

```
C:\Users\HP\srinitha>docker pull redis
Using default tag: latest
latest: Pulling from library/redis
bb263680fed1: Pull complete
ac509f65c3e9: Pull complete
51afc2cce3df: Pull complete
817f7e347ebd: Pull complete
ab1a1215d5f9: Pull complete
db7c27bf3552: Pull complete
Digest: sha256:6a59f1cbb8d28ac484176d52c473494859a512ddba3ea62a547258cf16c9b3ae
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest
```

Step 2:

We can create a new redis image from the downloaded image and expose it on port 90 using the following command.

docker run --name docker-redis –p 90:90 –d redis

```
C:\Users\HP\srinitha>docker run --name docker-redis -p 90:90 -d redis
8849e48daeed79ae68cacb3207a676b1ada5de9b57774c91f12c6bde0a38f40f
```

This is the executed output

Give docker ps command to get the recent downloaded image and its image id.

If we give docker ps –all then it show all the available docker images.

```
C:\Users\HP\srinitha>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                           NAM
ES
8849e48daeed   redis     "docker-entrypoint.s…"   17 seconds ago   Up 15 seconds   0.0.0.0:90->90/tcp, 6379/tcp   doc
ker-redis

C:\Users\HP\srinitha>docker exec -t docker-redis "docker-entrypoint.s
```

Now connect to the running container with command “docker run <image-name or image-id>”.

Or “docker exec –it docker-redis sh”

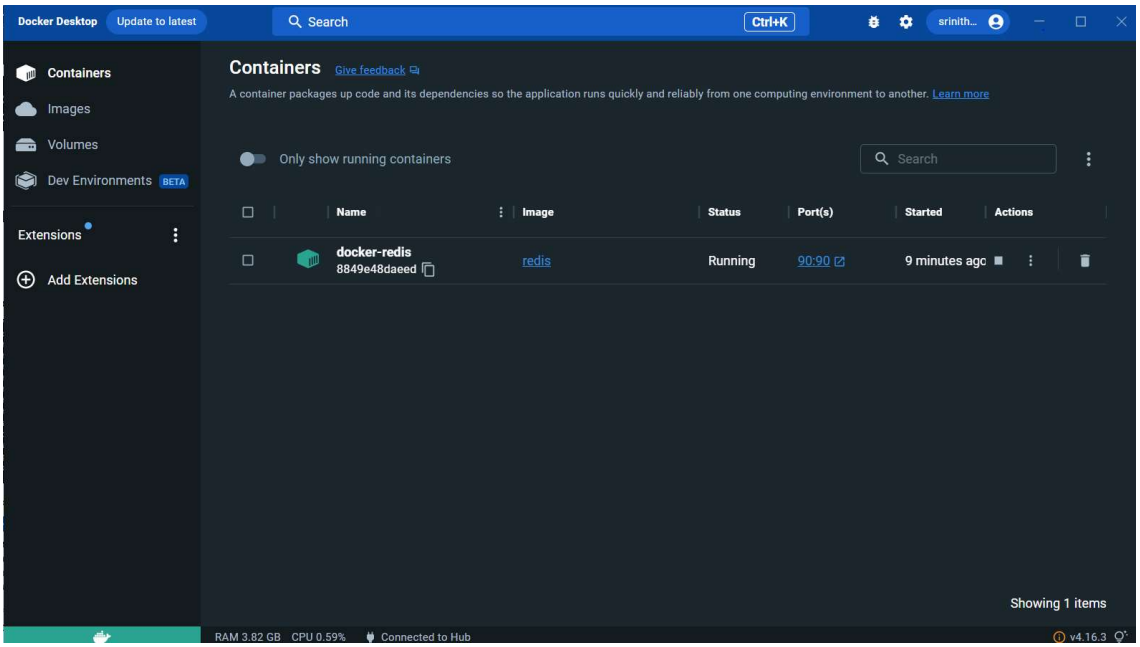
It results in the following output.

```
C:\Users\HP\srinitha>docker exec -t docker-redis sh
# ^C
C:\Users\HP\srinitha>rerredis-cli
'rerredis-cli' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\HP\srinitha>
C:\Users\HP\srinitha>
C:\Users\HP\srinitha>
C:\Users\HP\srinitha>docker exec -it docker-redis sh
# redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> set id 12
OK
127.0.0.1:6379> get id
```

Now we are connected to the running container.

Open docker desktop to view the container is created or not.



Q2) Create the basic java application, generate its image with necessary files, and execute it with docker.

Creating the basic java application.

Step1:

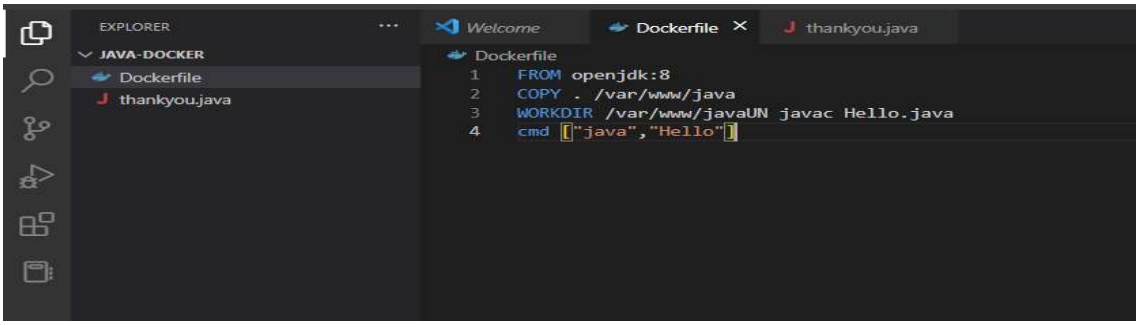
Create a directory it is used to store the files.

```
C:\Users\HP\srinitha>mkdir java-docker

C:\Users\HP\srinitha>cd java-docker
```

```
C:\Users\HP\srinitha\java-docker>code .

C:\Users\HP\srinitha\java-docker>
```



Create a docker file.

Create a java file, save it as Hello.java

```
C:\Users\HP\java-docker>docker build -t java-app .
[+] Building 5.8s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 140B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/openjdk:8
 => [auth] library/openjdk:pull token for registry-1.docker.io
 => [internal] load build context
 => => transferring context: 322B
 => CACHED [1/4] FROM docker.io/library/openjdk:8@sha256:86e863cc57215cfb181bd319736d0baf625fe8f15057
 => [2/4] COPY . /var/www/java
 => [3/4] WORKDIR /var/www/java
 => [4/4] RUN javac Thank.java
 => exporting to image
 => => exporting layers
 => => writing image sha256:bdb0febd876ed34285b98944f0c23ac8f408fcc0bfffa13ce0d257a7f75f1a5e
 => => naming to docker.io/library/java-app
```

```
C:\Users\HP\java-docker>docker run java-app
first time using java application

C:\Users\HP\java-docker>
```