Problem1:

Create bridge network with subnet 192.168.0.0/24.
Run 2 containers and attach containers to this network.
Create another bridge network with subnet 10.5.0.0/24.
Run any container and attach it to the new network.
Make sure that the containers at different network can’t ping each other

```
fady@fady-Elia:~$ docker network create --subnet=192.168.0.0/24 mynetwork1
0d3c08b75077ec07158995dd743a8e5e7882778e7d207b850235b9a1085b5696
fady@fady-Elia:~$ docker run -d --name container1 --network=mynetwork1 nginx

f9970ac99da5122cd5fce16f08c0fa19063f3efa818471f14173854a144af18b
fady@fady-Elia:~$ docker run -d --name container2 --network=mynetwork1 nginx
d501d0281593a9f980e681122c7510d4f6ea3921f68208226ebdfdfaa9fd1306
fady@fady-Elia:~$ docker network create --subnet=10.5.0.0/24 mynetwork2
d03b2f1dac58ad3315ad7db42bd289ae57c5fd2ee565a4144362241816f1e9ce
fady@fady-Elia:~$ docker run -d --name container3 --network=mynetwork2 nginx
fad4a6305051dde8f199105889ab57c8be29e13044d3038ddfc568fa79f2e1e2
fady@fady-Elia:~$ docker exec -it container1 sh
```

```
fady@fady-Elia:~$ docker exec -it container1 /bin/bash
root@f9970ac99da5:/# ping container3
ping: unknown host
root@f9970ac99da5:/# ping container2
PING container2 (192.168.0.3): 56 data bytes
64 bytes from 192.168.0.3: icmp_seq=0 ttl=64 time=0.551 ms
64 bytes from 192.168.0.3: icmp_seq=1 ttl=64 time=0.258 ms
64 bytes from 192.168.0.3: icmp_seq=2 ttl=64 time=0.233 ms
64 bytes from 192.168.0.3: icmp_seq=3 ttl=64 time=0.212 ms
64 bytes from 192.168.0.3: icmp_seq=4 ttl=64 time=0.246 ms
64 bytes from 192.168.0.3: icmp_seq=5 ttl=64 time=0.139 ms
^X64 bytes from 192.168.0.3: icmp_seq=6 ttl=64 time=0.254 ms
^Z
[1]+  Stopped                 ping container2
root@f9970ac99da5:/# exit
exit
There are stopped jobs.
root@f9970ac99da5:/# exit
exit
fady@fady-Elia:~$
```

2. Problem 2:
   Create static html file
   Write Dockerfile to build image based on httpd to host the html file and specify the following
   Copy the html file.
   Copy a new configuration file to listen on port 9999 instead of 80
   Open the port 9999 in the container
   Add environment variable CONTAINER with value docker .
   Add startup command to echo the variable

```
FROM httpd

COPY index.html /usr/local/apache2/htdocs/

COPY httpd.conf /usr/local/apache2/conf/httpd.conf

EXPOSE 9999

ENV CONTAINER docker

CMD [ "sh", "-c", "echo The container is running with CONTAINER=$CONTAINER && httpd-foreground" ]
```

```
fady@fady-Elia:~/problem2$ docker build -t static-html .
Sending build context to Docker daemon  4.096kB
Step 1/6 : FROM httpd
 ---> ad303d7f80f9
Step 2/6 : COPY index.html /usr/local/apache2/htdocs/
 ---> a0e1c083d8fe
Step 3/6 : COPY httpd.conf /usr/local/apache2/conf/httpd.conf
 ---> 3acb1501cd50
Step 4/6 : EXPOSE 9999
 ---> Running in adf93e4fc774
Removing intermediate container adf93e4fc774
 ---> e80cd16e04e0
Step 5/6 : ENV CONTAINER docker
 ---> Running in 4d5f785b0cc5
Removing intermediate container 4d5f785b0cc5
 ---> 198498cbf8cb
Step 6/6 : CMD [ "sh", "-c", "echo The container is running with CONTAINER=$CONTAINER && httpd-foreground" ]
 ---> Running in 1dc79ca941ae
Removing intermediate container 1dc79ca941ae
 ---> 14ce7beeac3f
Successfully built 14ce7beeac3f
Successfully tagged static-html:latest
fady@fady-Elia:~/problem2$ docker run -d -p 9999:9999 static-html
e4feafbc8b7c5f89383f982e1843e11fedfc409d3791acd18514440faa541fab
fady@fady-Elia:~/problem2$
```

3. Problem 3:
   Create a docker compose to up mysql container, and
   https://github.com/sabreensalama/dockerize-node-app-task which depend on mysqldb.
   Add volume for mysqldb

```
fady@fady-Elia:~/problem3/dockerize-node-app-task$ docker-compose up
dockerize-node-app-task_db_1 is up-to-date
dockerize-node-app-task_app_1 is up-to-date
Attaching to dockerize-node-app-task_db_1, dockerize-node-app-task_app_1
app_1  | internal/modules/cjs/loader.js:934
app_1  |   throw err;
app_1  |   ^
app_1  |
app_1  | Error: Cannot find module '/app/app.js'
app_1  |     at Function.Module._resolveFilename (internal/modules/cjs/loader.js:931:15)
app_1  |     at Function.Module._load (internal/modules/cjs/loader.js:774:27)
app_1  |     at Function.executeUserEntryPoint [as runMain] (internal/modules/run_main.js:75:12)
app_1  |     at internal/main/run_main_module.js:17:47 {
app_1  |   code: 'MODULE_NOT_FOUND',
app_1  |   requireStack: []
app_1  | }
app_1  | internal/modules/cjs/loader.js:934
```

```yaml
version: '3'
services:
  db:
    image: mysql:latest
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rootpassword
      MYSQL_DATABASE: mydb
      MYSQL_USER: myuser
      MYSQL_PASSWORD: mypassword
    volumes:
      - db-data:/var/lib/mysql

  app:
    build: /home/fady/problem3/dockerize-node-app-task
    restart: always
    ports:
      - 3000:3000
    depends_on:
      - db
    environment:
      DB_HOST: db
      DB_PORT: 3306
      DB_USER: myuser
      DB_PASSWORD: mypassword
      DB_NAME: mydb

volumes:
  db-data:
```

```
FROM node:14

WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

EXPOSE 3000

CMD [ "node", "app.js" ]
```

Use docker compose to deploy ghost platform (image: ghost:1-alpine)(Ghost is a free and open source blogging platform written in JavaScript)

Use mysql database instead of sqlite

```
fady@fady-Elia:~/problem4$ docker-compose up -d
problem4_db_1 is up-to-date
Starting problem4_ghost_1 ... done
fady@fady-Elia:~/problem4$ docker ps
version: '3'
services:
  db:
    image: mysql:5.7
    volumes:
      - mysql_data:/var/lib/mysql
    environment:
      - MYSQL_ROOT_PASSWORD=your_root_password
      - MYSQL_DATABASE=ghost
      - MYSQL_USER=ghost_user
      - MYSQL_PASSWORD=ghost_password

  ghost:
    image: ghost:1-alpine
    ports:
      - 2368:2368
    environment:
      - url=http://localhost:2368
      - database__client=mysql
      - database__connection__host=db
      - database__connection__user=ghost_user
      - database__connection__password=ghost_password
      - database__connection__database=ghost
    depends_on:
      - db

volumes:
  mysql_data:
```