Assignment 6 **(30 Sep 2022)**

A) Warm up on Docker

Install and configure Docker/Container platform in an EC2 instance on AWS. Depending on your system, you might have to use `sudo` for all commands given below.

1. Install Docker Community Edition on your machine : `https://docs.docker.com/install/`
   On ubuntu it should be as easy as
   ```
   apt-get install docker docker.io -y
   ```
2. type **docker** in your terminal and check if its properly working.
3. Go to docker `https://hub.docker.com`. Search for the Python official image. And pull the image having python version 3.8.
   ```
   docker pull python:3.8-slim
   #check the downloaded image

   docker images
   ```
4. Fire up a container container.
   ```
   #you can name it anything you want

   docker run -dit --name=pyContainer python:3.8-slim
   #your container should be listed here

   docker container ls
   #go inside your container, remember this

   dockerexec-it pyContainer /bin/bash
   ```

   If you have done it correctly, the something like the following should show up in your terminal
   ```
   root@02UwU3:/#
   ```

5. Inside the container go ahead check of the version python that you want is running correctly.
   ```
   root@02UwU3:/# python -c"print('hello world')"
   root@02UwU3:/# python --version
   #take note of the directory structure

   root@02UwW3:/# ls
   # exit the container like so

   root@02UwW3:/#exit
   ```

6. The container is basically running linux, so think of it as being inside the terminal of your own laptop. You can run **apt-get update** and see for yourself. However, it is running in isolation, so its interaction with your machine is limited (can be changed). This container will have python3.8 installed by default, so that you don't needlessly waste time setting up a another installation of python. This concept extends to any tool that you want to use!

7. Since the container is running isolated and running inside your terminal, the only way for you now is to create a new file inside the container and use commandline tools like vim, nano to make write any sort of script inside the container. This could be an annoyance.(if you are okay with it,then good for you.) Lets fix that. Since we cannot edit a launched container, we will stop and delete it and create a new container where we mount a folder of your local machine inside the container, so that any changes you make in that directory of your local machine shows up in your container. So, in your local terminal, run the following

```
# recall what you named your container

# if you dont, just 'docker container ls'

docker container stop pyContainer
docker container rm pyContainer
#create a new folder or use an existing one

mkdir testfolder
#find out the path of your folder

pwd

# should print out something like /Users/userName/

docker run -dit --name=pyC¥

    -v /Users/userName/testfolder:/myfolder¥
    python:3.8-slim
dockerexec-it pyC /bin/bash
root@02UwU3:/# ls
# your mounted folder should up as 'myfolder'. Any changes in this

# directory should show up in both your machine and this container
```

1. So just pull in your co-workers file, make the necessary changes and make sure it runs. And that it! You are done!

Here is the code that you need to finish implementing

```
fromtypingimportCallable, List
importmath
#DO NOT MAKE UNNECESSARY CHANGES

classDistanceFuncs:
    defcalc_distance(
        self, point_a: List[float], point_b: List[float], dist_func: Callable,/
    )->float:
        """ Calculates distance between two points using the passed dist_func """

        returndist_func(point_a, point_b)

    @staticmethod
    defeuclidean(point_a: List[float], point_b: List[float],/)->float:
        """

        Calculates Euclidean Distance between two points Example:

        >>> DistanceFuncs.euclidean([1,1],[1,1])
        0.0

        """

        returnmath.dist(point_a, point_b)
    @staticmethod
    defmanhattan_distance(point_a: List[float], point_b: List[float],/): """Compute
        the manhattan_distance between two points""" raiseNotImplementedError()

    @staticmethod
    defjaccard_distance(point_a: List[float], point_b: List[float],/): """Compute
        the jaccard_distance between two points""" raiseNotImplementedError()

defmain():
    """Demonstrate the usage of DistanceFuncs """

    pass
if_name_=="_main_":
    main()
```

B) Your task is to extend the assignment 4. This time you are going to use containers instead of virtual machine images for deploying your applications to the cloud (the containers, however, will still run in virtual machines.