```
import numpy
import scipy
import torch
import torchvision
from matplotlib import pyplot as plt

from IPython.display import display, HTML

def task1a():
    X = numpy.genfromtxt('csvdata/Wholesale customers data.csv',
                         delimiter=',',
                         names=True,
                         usecols=numpy.arange(2,8))

    features = X.dtype.names

    for feature in features:
        print(f"{feature:20s} {X[feature].mean():10.3f} {numpy.median(X[feature]):10.3f}")

def task1b():
    X = numpy.genfromtxt('csvdata/CortexNuclear.csv',
                     delimiter=',',
                     skip_header=1,
                     usecols=numpy.arange(1,78))

    X = X[~numpy.isnan(X).any(axis=1)]
    plt.imshow(X[:30])

def task2a(db):

    cursor = db.cursor()
    query = '''SELECT genres.Name,tracks.Milliseconds
               FROM tracks INNER JOIN genres
               ON genres.GenreID = tracks.GenreID;'''
    results = numpy.array(cursor.execute(query).fetchall())

    genres = results[:,0]
    duration = results[:,1].astype('float')/1000.0

    for genre in sorted(list(set(genres))):

        mean = duration[genres==genre].mean()

        print(f"{genre:20s} {mean:8.3f}")

def task2b(db):

    cursor = db.cursor()
    query = '''
        SELECT genres.Name,customers.Country FROM
        invoice_items
        INNER JOIN invoices ON invoice_items.InvoiceId = invoices.InvoiceId
        INNER JOIN customers ON customers.CustomerId = invoices.CustomerId
        INNER JOIN tracks ON invoice_items.TrackId = tracks.TrackId
        INNER JOIN genres ON genres.GenreId = tracks.GenreId;'''
    results = numpy.array(cursor.execute(query).fetchall())

    genres    = results[:,0]
    countries = results[:,1]

    listgenres = sorted(list(set(genres)))
    listcountries = sorted(list(set(countries)))

    print(f"{'':15s}" + "".join([f"{c[:3]:3s}" for c in listcountries]))

    for genre in listgenres:
        counts = []
        for country in listcountries:
            count = len(results[(genres==genre) & (countries==country)])
```

```
            counts.append(f"{count:3d}")

        print(f"{genre[:15]:15s}" + "".join(counts))

def task3a(imagesresize):
    X = numpy.array(imagesresize)
    D = scipy.spatial.distance.cdist(X,X)
    return D

def task3b(images,model,normalize):

    Z = []

    for img in images:

        x = torchvision.transforms.ToTensor()(img).unsqueeze_(0)
        z = model.forward(normalize(x))
        Z.append(z.data.numpy()[0].mean(axis=2).mean(axis=1))

    Z = numpy.array(Z)

    D = scipy.spatial.distance.cdist(Z,Z)

    return D
```