Python Senarios

# 1st Senarios

# area variables (in square meters)

hall = 11.25

kit = 18.0

liv = 20.0

bed = 10.75

bath = 9.50

# Create list areas

areas =[hall,kit,liv,bed,bath] – you can store diff data type values also

areas = ["hallway",hall, "kitchen",kit, "living room", liv, "bedroom",bed, "bathroom", bath] – With diff datatype

cities = ["Albuquerque", "Anaheim", "Anchorage", "Arlington", "Atlanta"] -- assining the value of cities

crime\_rates = [749, 371, 828, 503, 1379]

* [1, 3, 4, 2] B. [[1, 2, 3], [4, 5, 7]] C. [1 + 2, "a" \* 5, 3] – it is also list

house = [["hallway", hall],

["kitchen", kit],

["living room", liv],

["bedroom", bed],

["bathroom", bath]]

downstairs=areas[0:6] – first 6 elements or [:6]

upstairs=areas[6:10] – last 4 elements or[6:]

areas[-1]=10.50 – replace the value in list

areas[4]="chill zone" -- replace the value in list

areas\_1 =areas + ["poolhouse", 24.5] – oto extend the list

del(areas[-4,-2])---- for remove the elements

# Create areas\_copy

areas\_copy = list(areas) –explicity change the list will not affect areas

# tuple is same as list but we cannot remove or udate the tuple but list can. Tuple is declared as () while list as []. we can do the compare operation in tuple

# Print areas

print(areas), print(type(house)), print(rows[0:5]) [Start:end][Inclusive:Exclusive]

x=areas[1] – to get first element

y=areas[-1] – to get last element

z=areas[5]

-----Functions in python ------

[print(HYPERLINK "https://docs.python.org/3/library/functions.html")](https://docs.python.org/3/library/functions.html),[type()](https://docs.python.org/3/library/functions.html).str(), int(), bool() and [float()](https://docs.python.org/3/library/functions.html) , replace

full\_sorted = sorted(full, reverse = True) – for sort

# Use upper() on room: room\_up

room\_up = room.upper()

# Print out the number of o's in room

print(room.count("o"))

print(areas.index(20.0)) – to get the index of the element

append(),remove(),reverse()

areas.append(24.5)

areas.reverse()

# Import the math package

import math –full import or from math import radians – selective import

from scipy.linalg import inv as my\_inv

C = 2 \* r \* math.pi

# Travel distance of Moon over 12 degrees. Store in dist.

dist=radians(12)\*r

Numpy Array

# Import the numpy package as np

import numpy as np

# Create a numpy array from baseball: np\_baseball

np\_baseball=np.array(baseball)

numpy arrays cannot contain elements with different types. If you try to build such a list, some of the elements' types are changed to end up with a homogeneous list. This is known as *type coercion*.

2d arithmetic ans stastics

# Create np\_height from np\_baseball

np\_height=np.array(np\_baseball[:,0])

# Print out the mean of np\_height

print(np.mean(np\_height))

# Print out the median of np\_height

print(np.median(np\_height))

# Convert positions and heights to numpy arrays: np\_positions, np\_heights

np\_positions = np.array(positions)

np\_heights = np.array(heights)

# Heights of the goalkeepers: gk\_heights

gk\_heights = np\_heights[np\_positions == 'GK']

# Heights of the other players: other\_heights

other\_heights = np\_heights[np\_positions != 'GK']

# Print out the median height of goalkeepers. Replace 'None'

print("Median height of goalkeepers: " + str(np.median(gk\_heights)))

# Print out the median height of other players. Replace 'None'

print("Median height of other players: " + str(np.median(other\_heights)))

-----Matplotlib---------

# Import matplotlib.pyplot as plt

import matplotlib.pyplot as plt

# Make a line plot: year on the x-axis, pop on the y-axis

plt.plot(year, pop)---for line plot

plt.scatter(x,y) --for scatter plot bubbles

plt.hist(life\_exp) --- Histogram View

plt.hist(life\_exp,bins=5) -- with bins or width of block

plt.xscale('log') -- make xaxis to log

plt.xlabel(xlab)

plt.ylabel(ylab)

plt.title(title)

plt.yticks([0,1,2], ["one","two","three"])

tick\_val = [1000,10000,100000]

tick\_lab = ['1k','10k','100k']

plt.xticks(tick\_val,tick\_lab)

plt.scatter(gdp\_cap, life\_exp, s = pop) -- s for size if we want to show the population acccording to size of bubble in population

plt.scatter(x = gdp\_cap, y = life\_exp, s = np.array(pop) \* 2, c=col, alpha =0.8) -- for color and alpha for darkness for color

# Additional customizations

plt.text(1550, 71, 'India')

plt.text(5700, 80, 'China')

# Add grid() call

plt.grid(True)

plt.clf() -- for clean up plot

plt.show()