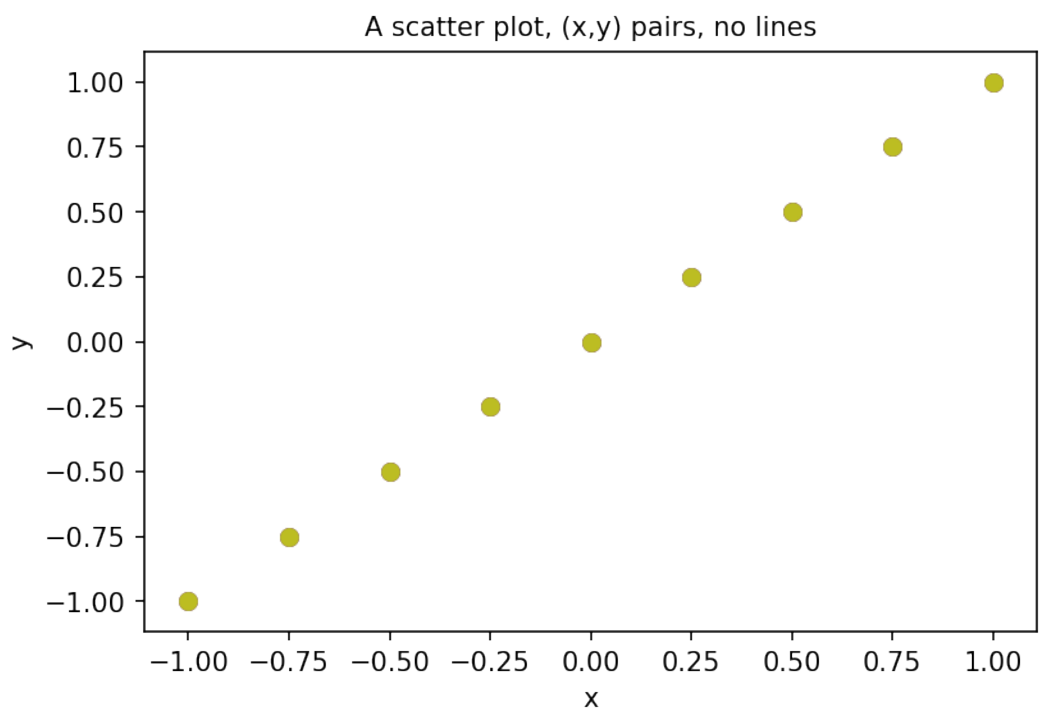
Matplotlib-2

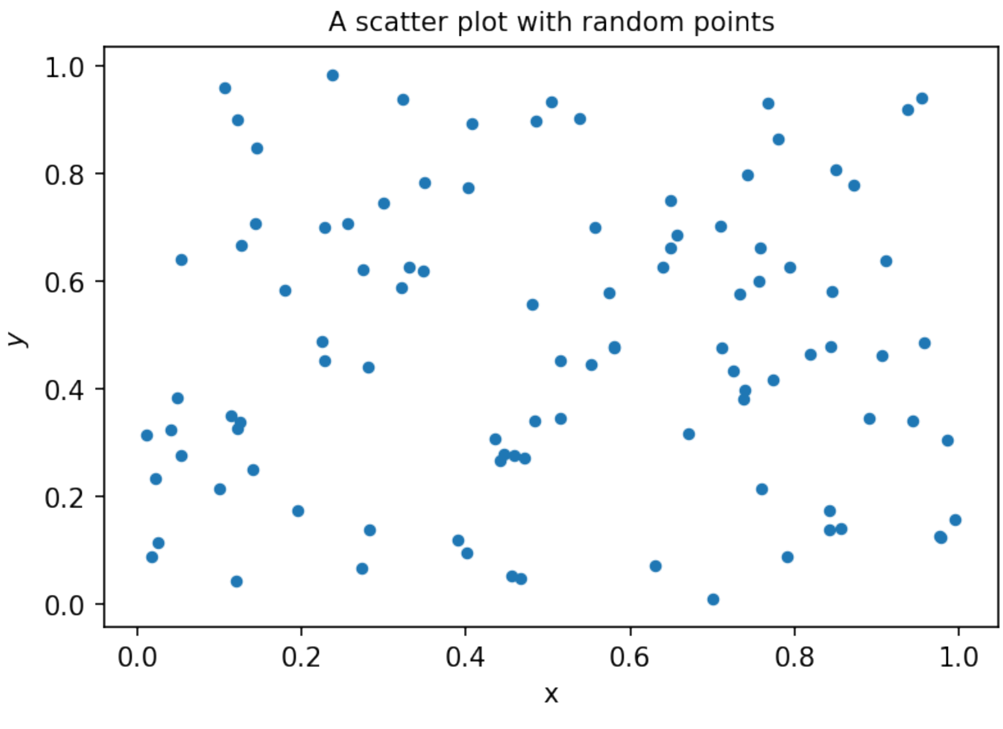
November 19, 2019

In [148]:
```
import matplotlib as mpl
import matplotlib.pyplot as plt  # import the module
mpl.rcParams['figure.dpi']= 150  # set the resolution to x dpi

import numpy as np
```

In [149]:
```
plt.title("A scatter plot, (x,y) pairs, no lines", fontsize=10)
plt.xlabel("x")
plt.ylabel("y")
x = np.arange(-1, 1.1, 0.25)  #  x coordinate points
y = x

for xval in x:
    plt.scatter(x, y)  # a scatter-plot doesn't include lines!

plt.show()
```

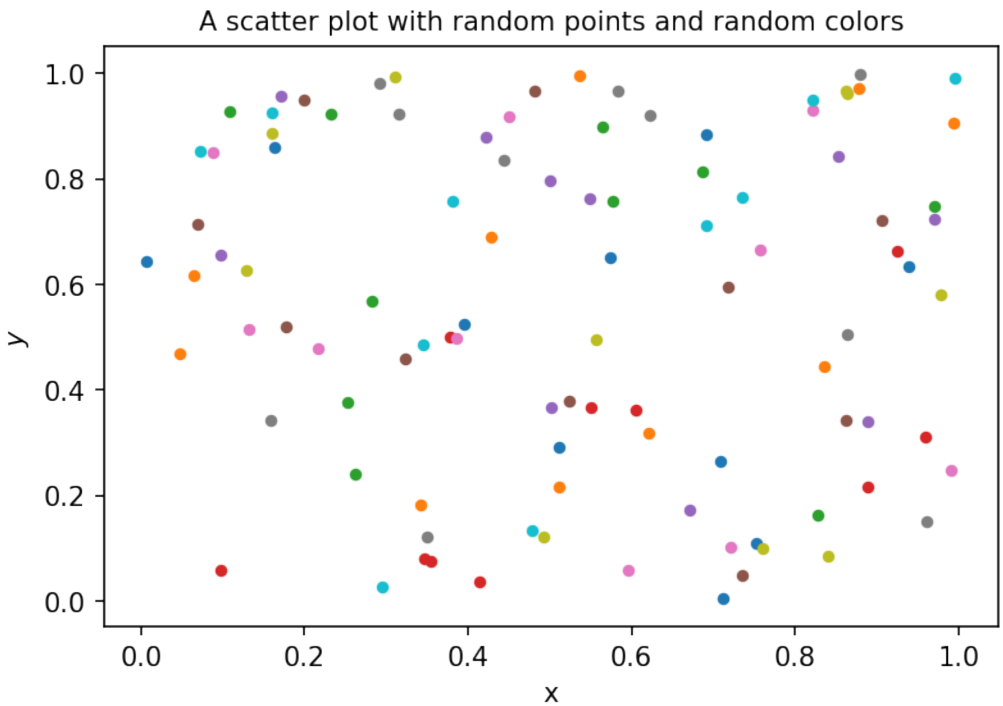


In [150]:
```
import random

plt.title("A scatter plot with random points", fontsize=10)
plt.xlabel("x")
plt.ylabel("$y$")

num_pts = 100
x = []
y = []

for i in range(num_pts):
    x.append(random.uniform(0, 1))
    y.append(random.uniform(0, 1))
    # the method uniform() returns a random number uniformly generated
    # in the real interval [a,b]

plt.scatter(x, y, marker='.', s=50)
plt.show()
```



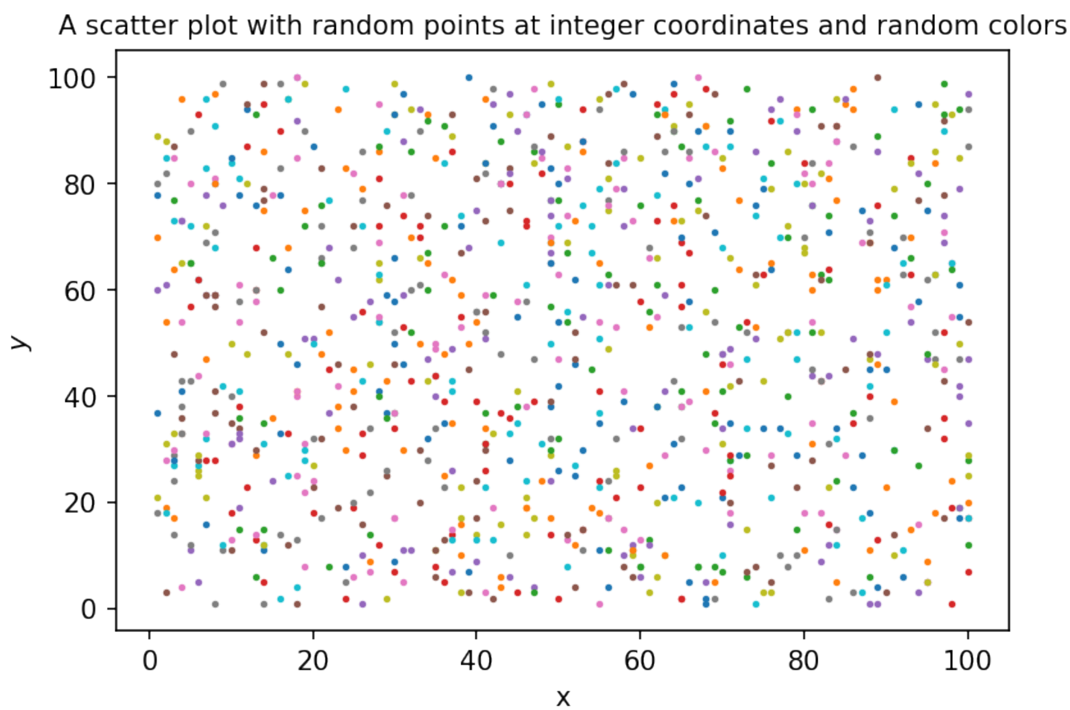
In [151]:
```
import random
```

```
plt.title("A scatter plot with random points and random colors", fontsize=10)
plt.xlabel("x")
plt.ylabel("$y$")

num_pts = 100

for i in range(num_pts):
    x = random.uniform(0, 1)
    y = random.uniform(0, 1)
    plt.scatter(x, y, marker='.', s=50)

plt.show()
```



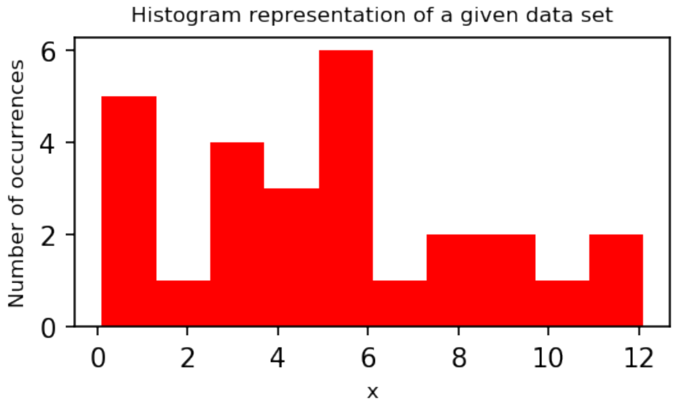
In [152]:
```
import random

plt.title("A scatter plot with random points at integer coordinates and random color
          fontsize=10)
plt.xlabel("x")
plt.ylabel("$y$")

num_pts = 1000


for i in range(num_pts):
```

```
    x = random.randint(1, 100)
    y = random.randint(1, 100)
    # the method randint(a, b) returns a random integer number uniformly generated
    # in the interval between a and b

    plt.scatter(x, y, marker='.', s=10)

plt.show()
```

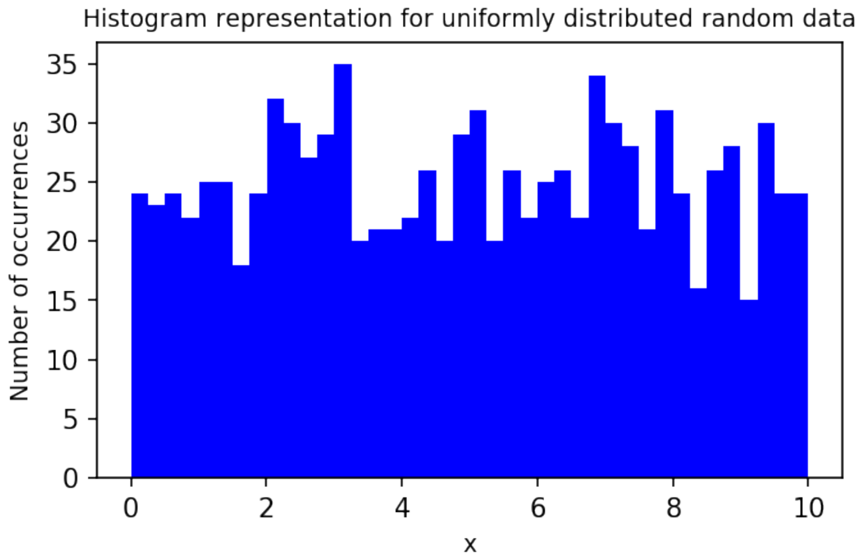


In [153]:
```
# Plot of an histogram of a set of values

# the dataset
xvals = [1, 5, 6, 6, 5.5, 4, 3, 7, 11, 2.3, 1.2, 0.5, 9,
         8.1, 7.7, 5.6, 3, 2.5, 3.1, 4, 5.2, 0.1, 12.1, 4.5, 10, 9.3, 0.9]

plt.figure(figsize = (4,2))
plt.title('Histogram representation of a given data set', fontsize=8)

ylabel = 'Number of occurrences'
plt.ylabel(ylabel, fontsize=8)

xlabel = 'x'
plt.xlabel(xlabel, fontsize=8)
```

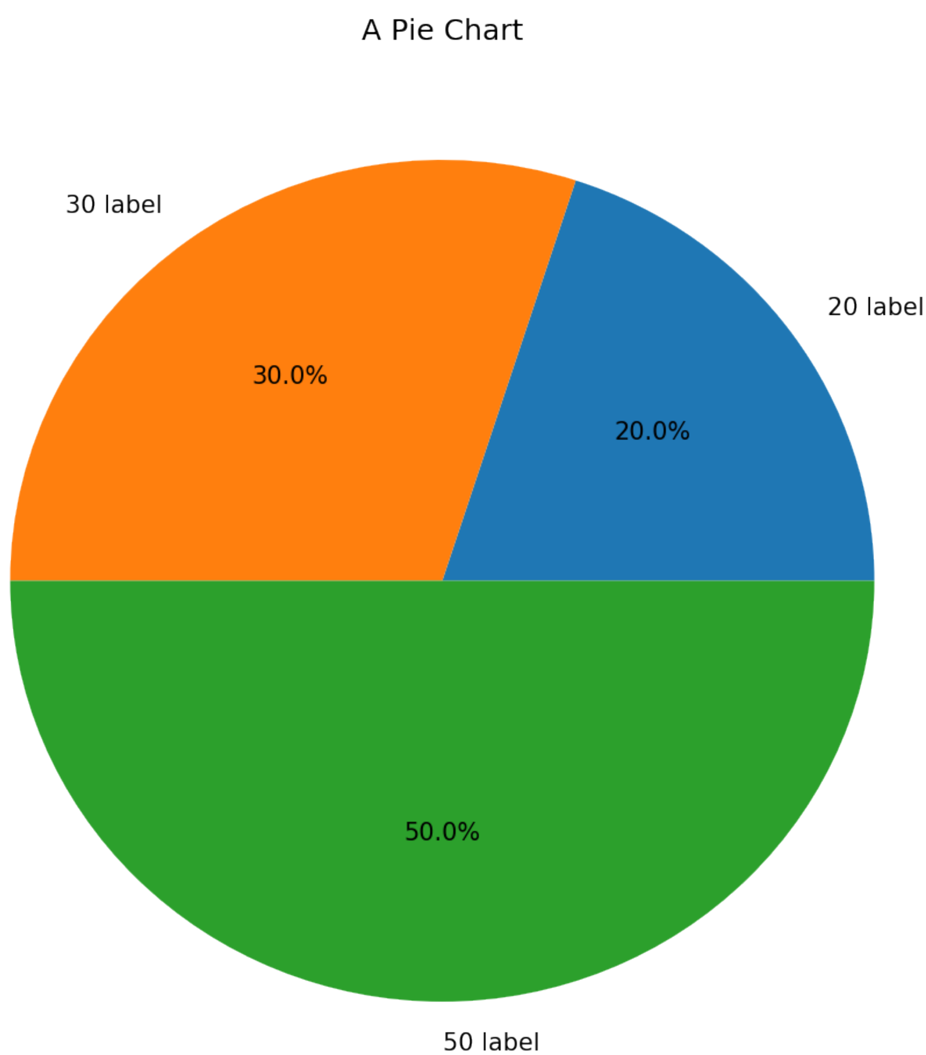
```
nbins = 10
plt.hist(xvals, nbins, density=False, color='red')

plt.show()
```



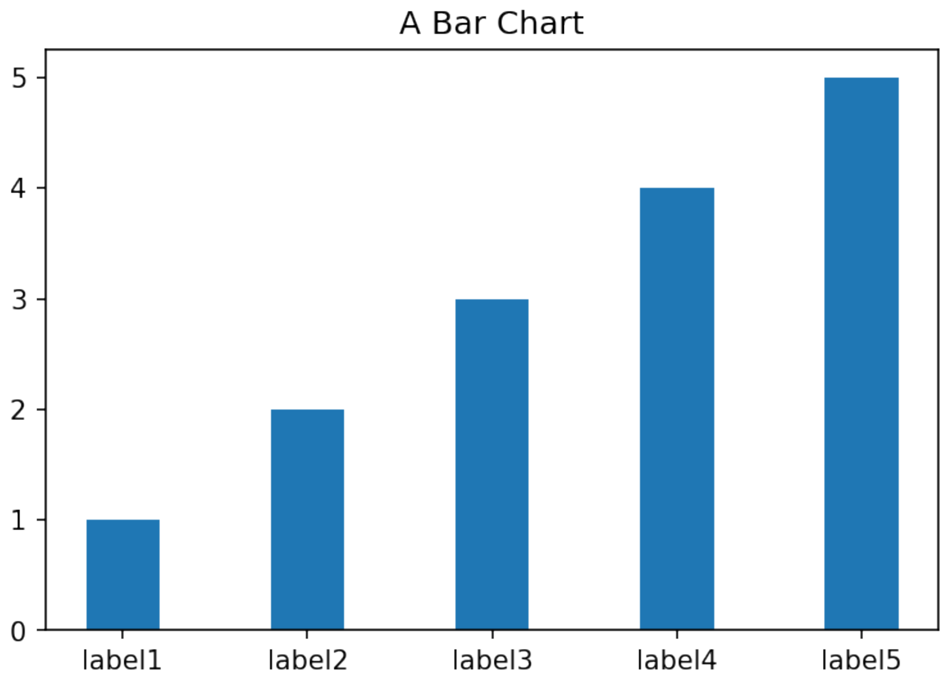
In [154]:
```
# Plot of an histogram of a random set of points generated in [xmin, xmax]
uniform_distr = []
xmin = 0
xmax = 10
num_pts = 1000
for i in range(num_pts):
    uniform_distr.append(random.uniform(xmin, xmax))

plt.figure(figsize = (5,3))
plt.title('Histogram representation for uniformly distributed random data', fontsize=

ylabel = 'Number of occurrences'
plt.ylabel(ylabel, fontsize=9)

xlabel = 'x'
plt.xlabel(xlabel, fontsize=9)

nbins = 40
plt.hist(uniform_distr, nbins, density=False, color='blue')


plt.show()
```



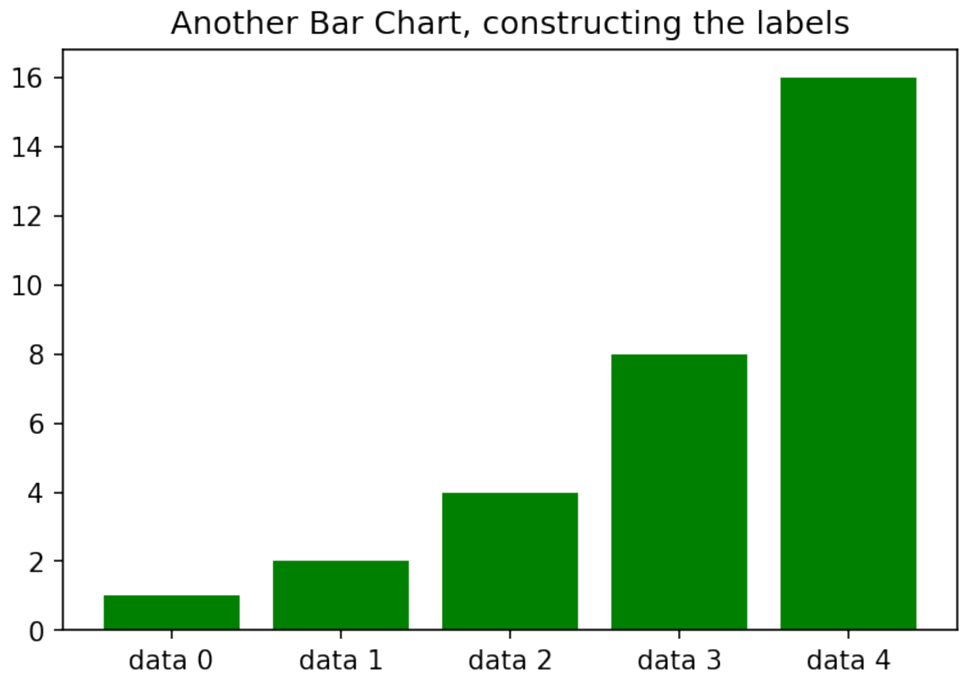

```
In [155]: xsize = 8
          ysize = 8
          plt.figure(figsize=(xsize, ysize))
          # Make the chart a circle, otherwise it'll be an ellipsis fir xsize different from y

          plt.title("A Pie Chart")
          plt.pie([20,30,50], labels=["20 label", "30 label", "50 label"], autopct="%.1f%%")
          plt.show()
```



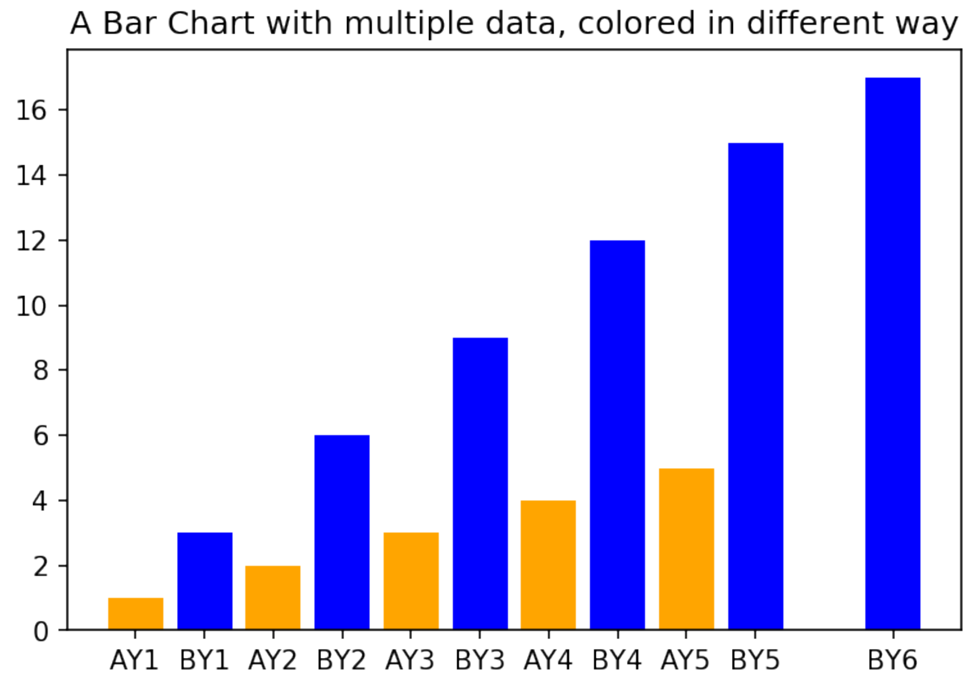

```
In [156]: plt.title("A Bar Chart")

          plt.bar(range(0,10,2), [1,2,3,4,5], # where the bars are located, value/height of th
                  tick_label=["label1", "label2", "label3", "label4", "label5"])

          plt.show()
```



In [157]:
```
plt.title("Another Bar Chart, constructing the labels")

dataset = [1, 2, 4, 8, 16]
labels = []
for i in range(len(dataset)):
    labels.append('data ' + str(i))

plt.bar(range(len(dataset)), dataset, tick_label=labels, color='green')
plt.show()
```



In [158]:
```
plt.title("A Bar Chart with multiple data, colored in different way")

dataset_1 = (list(range(0, 10, 2)), [1,2,3,4,5])
labels_1  = ["AY1", "AY2", "AY3", "AY4", "AY5"]

dataset_2 = (list(range(1, 13, 2)), [3, 6, 9, 12, 15, 17])
labels_2  = ["BY1", "BY2", "BY3", "BY4", "BY5", 'BY6']

plt.bar(dataset_1[0] + dataset_2[0],
        dataset_1[1] + dataset_2[1],
        tick_label = labels_1 + labels_2,
        color= ['orange']*len(dataset_1[1]) + ['b']*len(dataset_2[1]) )
        # a list of 'oragne' for each element in dataset_1, and a list
        # of 'b' for each element in dataset_2

plt.show()
```



In [ ]:
```
# this is the file price_list.csv used in the cell below
#(remove these comment lines if cut & paste the records in a file)
period,price,returns
1,0.708478,0.9137454899729256
2,0.6473685772450384,0.9285440434833032
3,0.6011102363391411,0.9361942709665733
4,0.5627559594800667,0.9602514986030374
5,0.5403872534385242,0.9568440158496524
6,0.5170663096940814,1.1190691645959412
7,0.5786329632300619,0.9125305382834752
8,0.5280202494048907,1.0185142901318816
9,0.5377961694978812,0.9759725855747301
10,0.5248743180570329,1.1172125413979572
11,0.5863961707910175,0.9565059222560891
12,0.5608914101499013,1.0067832670125132
13,0.5646960863499731,0.9318443058364185
14,0.526208832593333,0.8375749336931183
15,0.440739328068094,0.9292209686506898
16,0.4095442253498885,0.9465985482471304
17,0.38767396915920005,1.0239974112443788
18,0.396977140825854,0.9209232986865893
19,0.36558549803251617,1.1110040571558837
20,0.4061669715514798,1.0317439207471584
21,0.4190603037065233,0.9646915513312078
22,0.404263934483973,0.9871614583088331
```

```
23,0.39907377510686537,1.0595548157143824
24,0.4228405402397976,1.0177264391117575
25,0.43033599733034106,0.9339335361701707
26,0.40190521972804255,1.0578806626082478
27,0.42516776015161506,0.9867432637094707
28,0.4195314232760501,0.9702829303502888
29,0.40706417875031325,1.0419769996013712
30,0.42415151161944764,1.067397516919094
31,0.4527382703000786,1.0000366512988252
32,0.452754863745713,0.8972005921998502
33,0.40621193187401616,0.8897075660925364
34,0.3614098292253781,1.0424349508294097
35,0.3767462375578224,0.9611786348546107
36,0.3621204343024386,1.0174989856162964
37,0.368457174573664,1.14085237139653
38,0.4203552413704298,1.089772603278595
39,0.45809162569005546,1.010772778852884
40,0.4630265454679725,0.965953787133708
41,0.4472622451382261,0.966229234981223
42,0.4321578569558924,1.0135265589556979
43,0.4380034656861744,1.1560031178700896
44,0.5063333719711224,0.9765175271957773
45,0.4944434123339401,1.0549140921351428
46,0.5215953234344605,0.8237088505489465
47,0.4296426843179054,0.8653700529601334
48,0.3717999124821197,1.130414261970813
49,0.42028792366928824,0.961424140444192
50,0.40407495575281965,0.9311752009824759
51,0.3762645781351169,0.9482652292000402
52,0.35679861642515304,1.1240105995546334
53,0.40104542676829996,0.9521153090007225
54,0.38184149043082655,1.0001470413733171
55,0.38189763692796896,0.8241466536738264
56,0.31473965952012756,0.9911831201503634
57,0.31196463775822303,1.0200672659916
58,0.3182249151240904,1.0133978668047139
59,0.32248845015086436,0.9295617719453477
60,0.29977293515414644,1.1058811397285129
61,0.33151323518802905,1.0434231026987497
62,0.3459085684455936,0.9234320708561621
63,0.31942306568660495,1.0903243427549942
64,0.3482747441555329,1.0037892813264968
65,0.3495944551400519,0.9167605266255439
66,0.32049439679956404,1.2455376530664954
67,0.39918783881069114,1.0192778087790786
68,0.4068833056342173,0.9458835317099098
69,0.3848642181270961,1.1003857692039531
70,0.4234991087028626,0.8869456438961426
```

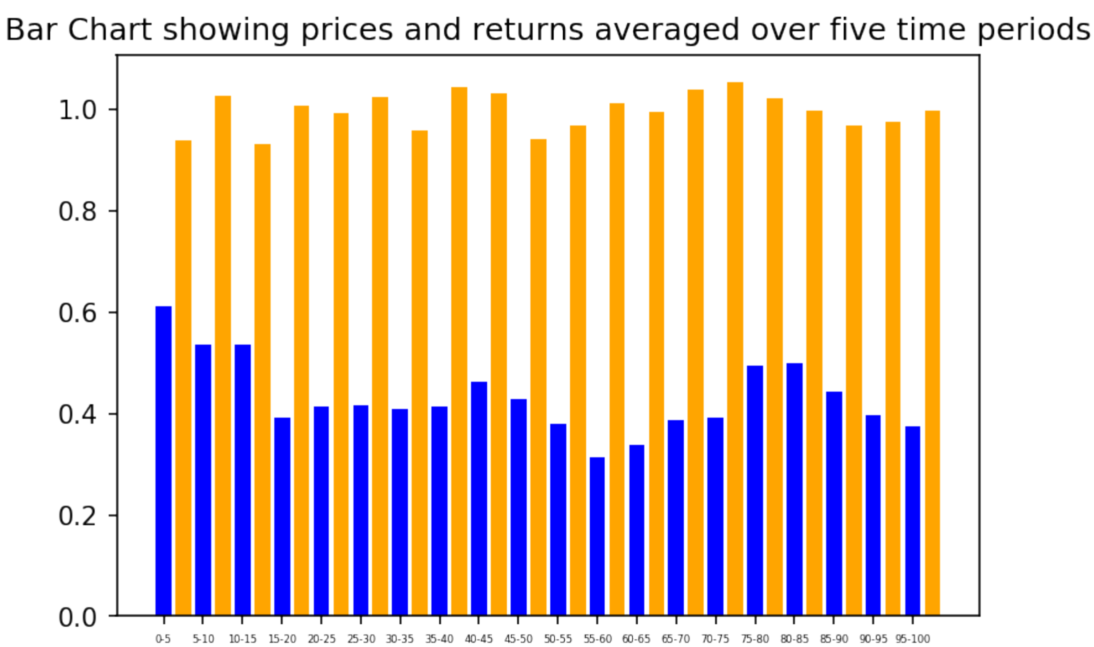
```
71,0.37562068965790296,0.9984843778025276
72,0.37505139060282755,1.1138490367468217
73,0.41775063015351543,0.9946444753991215
74,0.4155133563766958,0.9081527192316808
75,0.37734958447057876,1.2571996566697117
76,0.47440376804087,0.9123728162245036
77,0.4328331018749647,1.1140274065330846
78,0.48218793794343734,1.0738673992339596
79,0.5178059068613049,1.0886932616183174
80,0.5637318016260647,0.925859071607911
81,0.5219362024893633,0.9643744929980159
82,0.5033419606529895,0.9795922744441014
83,0.49306989605921536,0.9959575254120362
84,0.491076673534306,1.0046876440016292
85,0.49337866615733916,1.0512576535503853
86,0.5186680988963832,0.8907694922653504
87,0.4620137191081658,0.9804390364743906
88,0.4529762856003598,0.8403344424752491
89,0.38065157441448755,1.0736174669148324
90,0.40867417910002496,1.0553637997082639
91,0.4312999344976579,0.8772602032940622
92,0.37836226821813107,1.0132035690231203
93,0.3833580005422935,1.0030985487067157
94,0.384545853979083,1.0557218457192001
95,0.4059734587264635,0.9363858907026422
96,0.3801478187512119,0.9855789615226175
97,0.37466569242990766,0.9908696579608094
98,0.3712448665076724,1.0846678198960338
99,0.4026773600024711,0.8653485962293821
100,0.34845628821149194,1.0680379224190018
```

In [159]:
```python
import os

datafile = 'csv/price_list.csv'
if os.path.isfile(datafile):
    f = open(datafile, 'r')
else:
    print('File ', datafile, "doesn't exist!")
    assert(False)

# datafile is a csv file with field: period, price, returns
# where period is an integer, while the other fields are floats
periods = []
prices = []
returns = []

records = f.readlines()
records.pop(0)
```

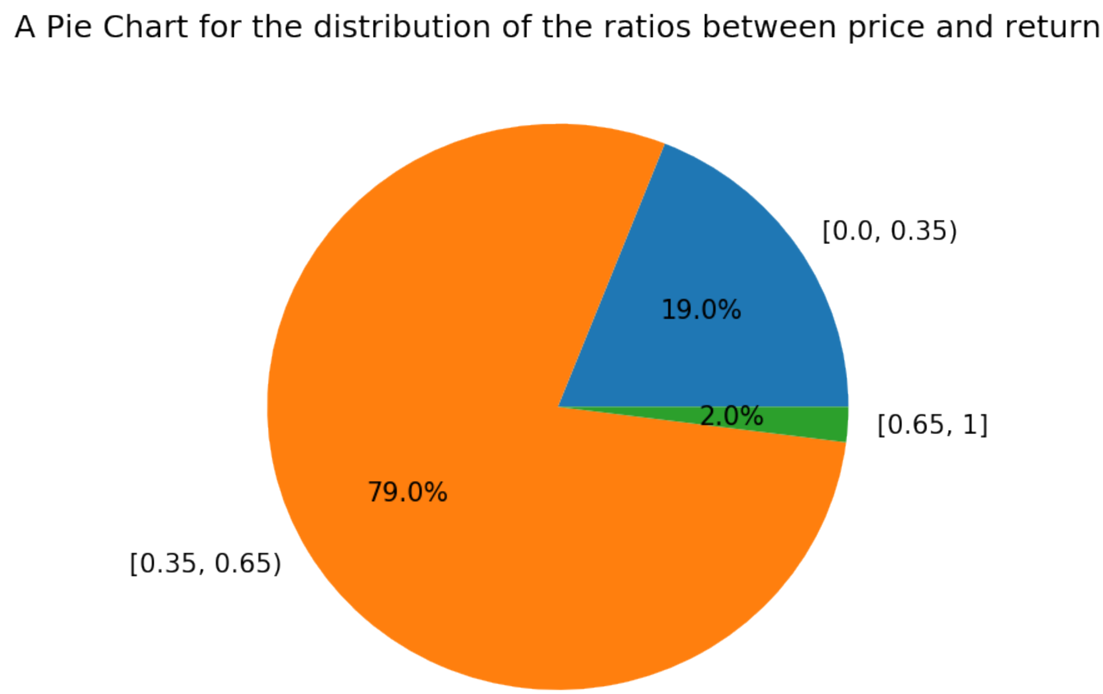
```
for r in records:
    fields = r.split(',')
    periods.append(int(fields[0]))
    prices.append(float(fields[1]))
    returns.append(float(fields[2]))

plt.title("Prices and Returns vs. Periods\n", fontsize=10)
plt.xlabel("Period")
plt.ylabel("Value")

p_legend, = plt.plot(periods, price, marker='.', color='red', markersize=2, linewidtl
r_legend, = plt.plot(periods, returns, marker='o', color='blue', markersize=2, linew

# plot the legends with the labels
plt.legend(handles=[p_legend, r_legend])
plt.show()
```


```
In [160]: plt.title("Bar Chart showing prices and returns averaged over five time periods")

          # prices and returns are clustered in 5-periods groups
          clustered_prices = []
```

```
clustered_returns = []

cluster_size = 5 #let's assume that we know that the number of periods is a multiple

# num of 5-periods cluster of valures
cluster_num = int(len(periods) / cluster_size)

# average the values of prices and returns over 5-period times
for i in range(cluster_num):
    clustered_prices.append( sum(prices[ cluster_size * i : cluster_size * (i+1)]) /
    clustered_returns.append( sum(returns[ cluster_size * i : cluster_size * (i+1)])

# create labels reporting the period intervals
labels = []
for i in range(cluster_num):
    labels.append('{}-{}'.format(i*cluster_size, (i+1)*cluster_size))

# reduce the fontsize otherwise labels overlap with each other
plt.xticks(fontsize=4)

# plot the bar charts for the two sets of data, the positions of the bars need to be
plt.bar(range(0, 2*cluster_num, 2), clustered_prices, tick_label = labels, color='blu
plt.bar(range(1, 2*cluster_num + 1, 2), clustered_returns, color='orange')

plt.show()
```



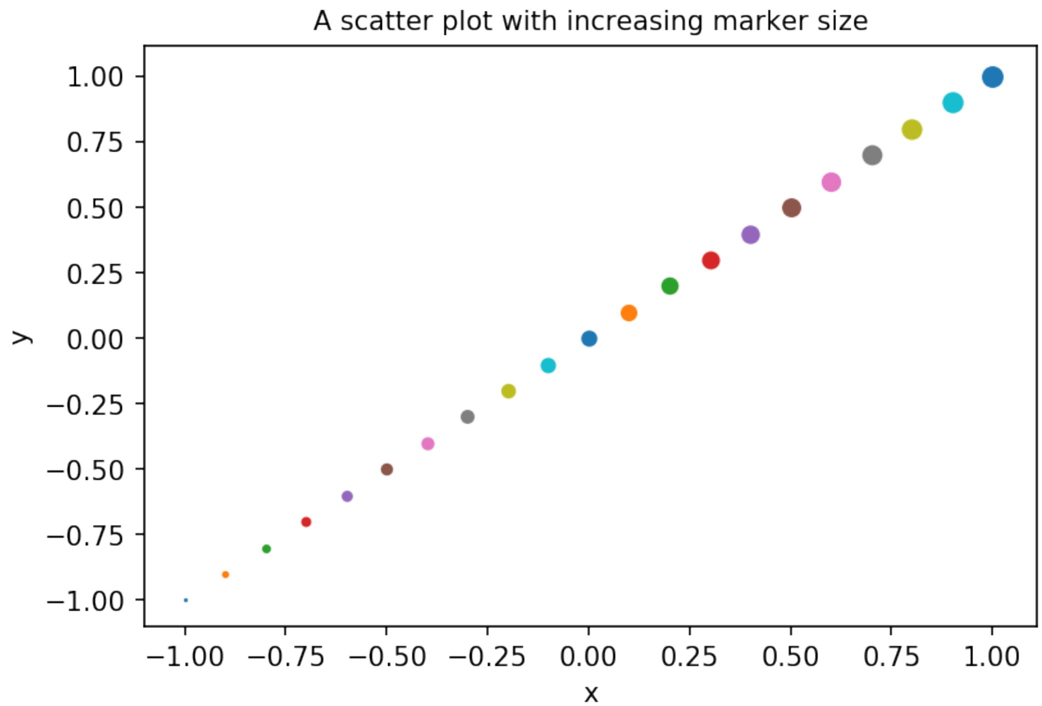
In [161]:
```
# Let's make a pie chart out of the data, showing how the ratios between prices and 
#
# Four classes are considered based on the values of the ratio pr = price/return.
#
# The four classes correspond to the following cases for the ratio:
# pr in [0, 0.35)
# pr in [0.35, 0.65)
# pr in (0.65, 1]

classes = [0]*3

for i in range(len(periods)):
    pr = prices[i] / returns[i]
    if pr < 0.35:
        classes[0] += 1
    elif pr < 0.65:
        classes[1] += 1
    else:
        classes[2] += 1

plt.figure(figsize=(5, 5))

plt.title("A Pie Chart for the distribution of the ratios between price and return")

plt.pie(classes, labels=["[0.0, 0.35)", "[0.35, 0.65)", "[0.65, 1]"], autopct="%.1f%

plt.show()
```



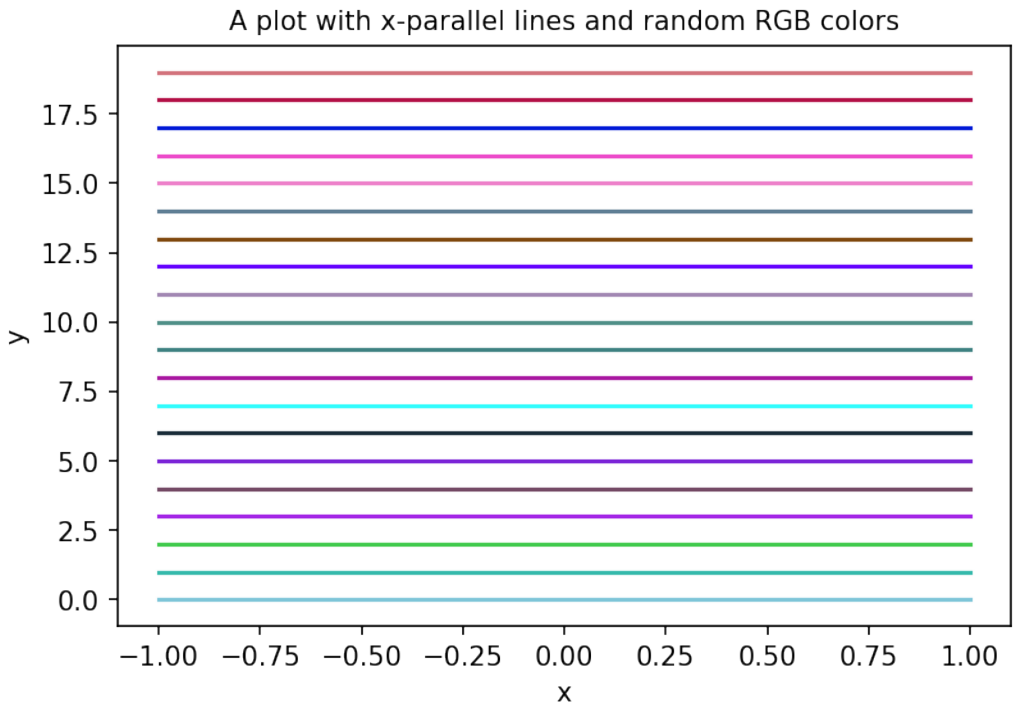
---

1 Optional material below (don't cover in class), just additional examples ...

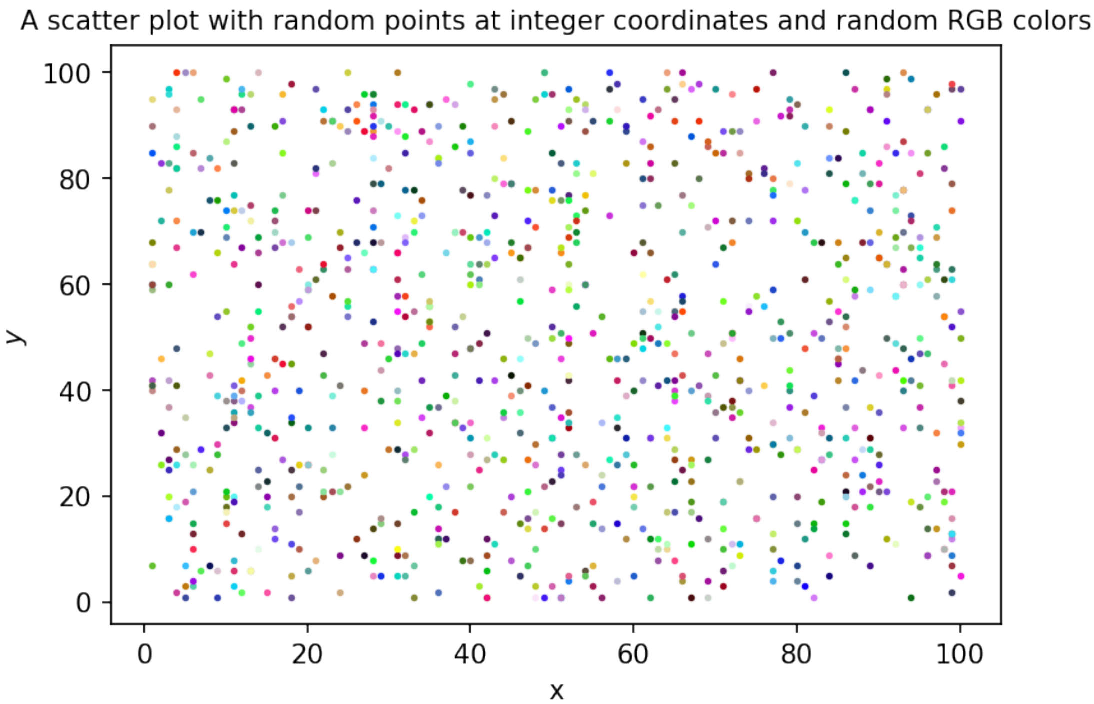
In [162]:
```
plt.title("A scatter plot with increasing marker size", fontsize=10)
plt.xlabel("x")
plt.ylabel("y")

x = np.arange(-1, 1.1, 0.1)  #  x coordinate points
y = x

size = 1
for i in range(len(x)):
    plt.scatter(x[i], y[i], marker='.', s=size) # increasing the marke size
    size += 10
plt.show()
```



In [163]:
```
import random

plt.title("A plot with x-parallel lines and random RGB colors", fontsize=10)
plt.xlabel("x")
plt.ylabel("y")
x = np.arange(-1, 1.1, 0.1)  #  x coordinate points
y = [1] * len(x)

lines = 20
for i in range(lines):
    y = [i] * len(x)  # x-parallel lines
    rgb = (random.uniform(0,1), random.uniform(0,1), random.uniform(0,1))
    plt.plot(x, y, marker='', color=rgb)
plt.show()
```

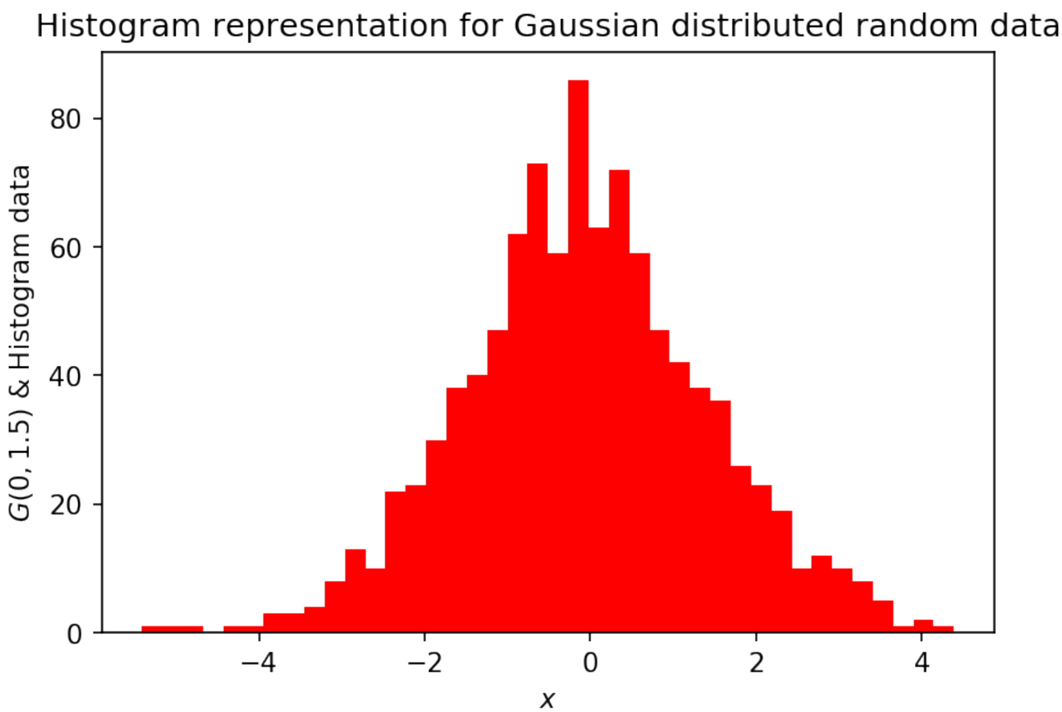


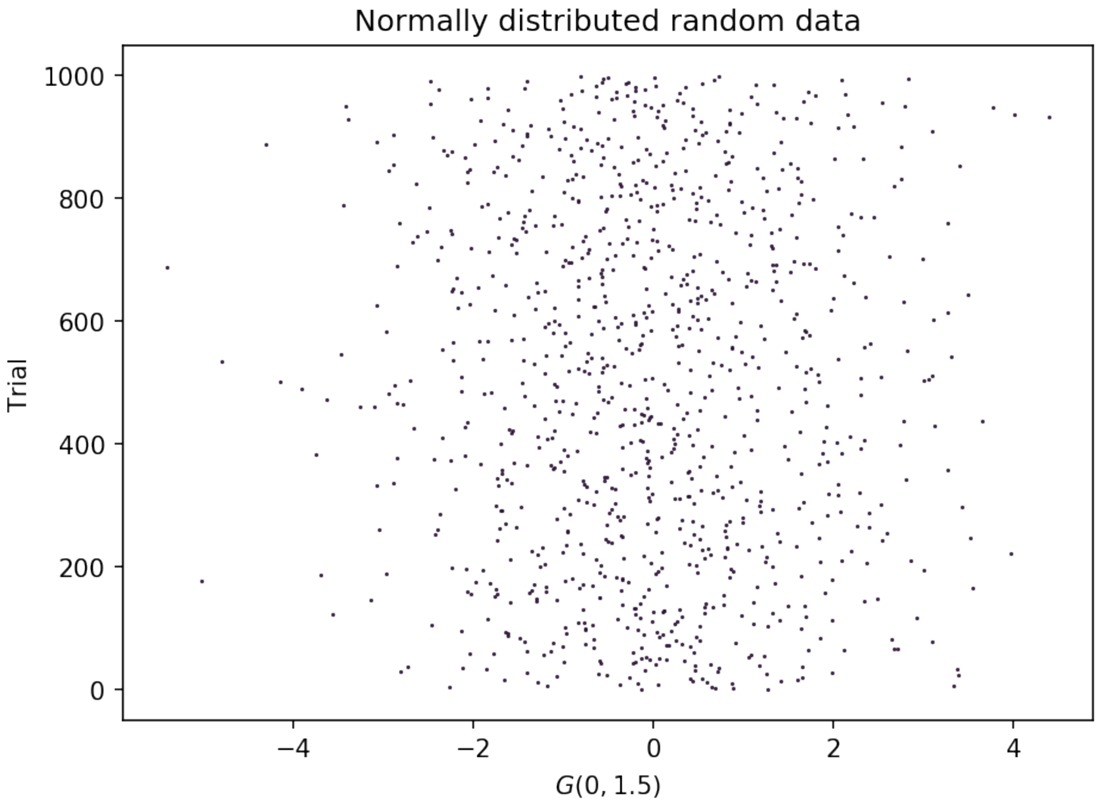
In [164]:
```
import random

plt.title("A scatter plot with random points at integer coordinates and random RGB co
plt.xlabel("x")
plt.ylabel("$y$")

num_pts = 1000


for i in range(num_pts):
    x = random.randint(1, 100)
    y = random.randint(1, 100)
    # the method randint(a, b) returns a random integer number uniformly generated
    # in the interval between a and b

    rgb = ( random.uniform(0,1), random.uniform(0,1), random.uniform(0,1))
    plt.scatter(x, y, marker='.', s=10, color=rgb)

plt.show()
```




```
In [165]: # Plot of an histogram of a random set of points distributed as a Gaussian
          gauss_distr = []
          mean = 0
          std_dev = 1.5
          generations = 1000
          for i in range(generations):
              gauss_distr.append(random.gauss(mean, std_dev))

          plt.figure(figsize = (6,4))
          plt.title('Histogram representation for Gaussian distributed random data')

          ylabel = '$G({}, {})$ & Histogram data'.format(mean, std_dev)
          plt.ylabel(ylabel)

          xlabel = '$x$'
          plt.xlabel(xlabel)

          nbins = 40
          plt.hist(gauss_distr, nbins, density=False, color='red')

          plt.show()

          # plot the data points
          plt.figure(figsize = (7,5))
```

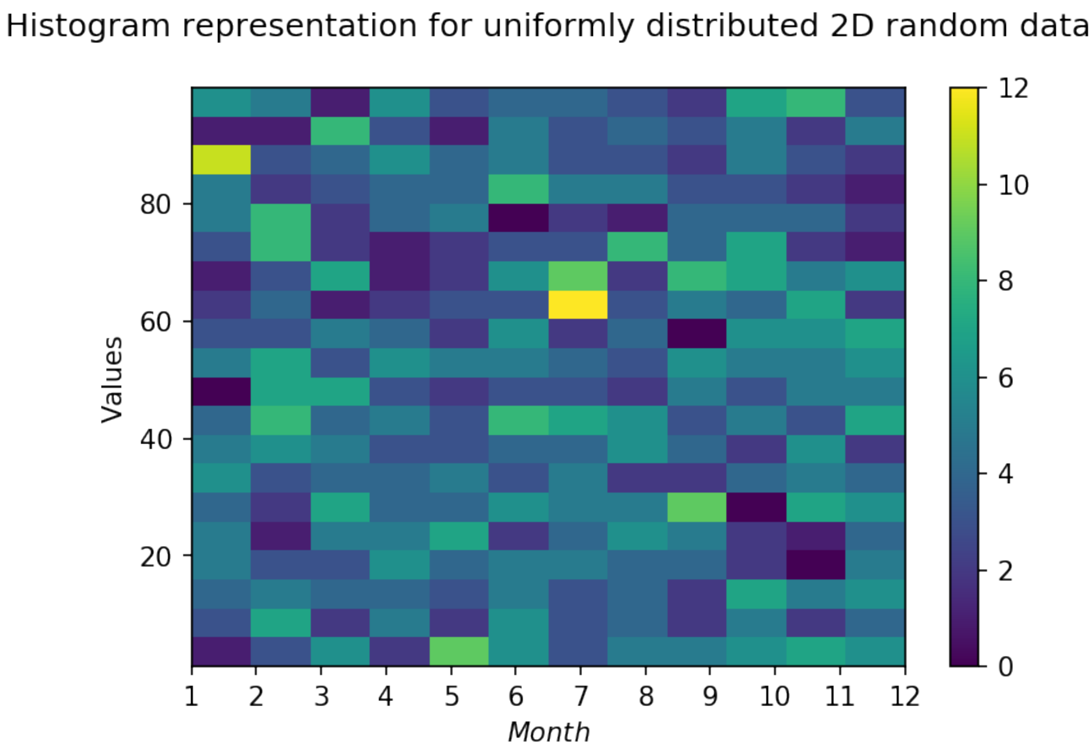
```
plt.title('Normally distributed random data')
xlabel = '$G({}, {})$'.format(mean, std_dev)
plt.xlabel(xlabel)
plt.ylabel('Trial')

rgb = (random.uniform(0,1), random.uniform(0,1), random.uniform(0,1))
plt.plot(gauss_distr, range(generations), linestyle='None', marker='.', markersize=1
plt.show()
```






```
In [166]: # Plot of a 2D histogram of a random set of (x,y) points
          x = []
          xmin = 1
          xmax = 12

          y = []
          ymin = 1
          ymax = 100

          num_pts = 1000

          for i in range(num_pts):
              x.append(random.randint(xmin, xmax))
              y.append(random.uniform(ymin, ymax))

          plt.figure(figsize = (6,4))
          plt.title('Histogram representation for uniformly distributed 2D random data\n')

          ylabel = 'Values'
          plt.ylabel(ylabel)
```

```
xlabel = '$Month$'
plt.xlabel(xlabel)
plt.xticks(np.arange(0, xmax+2, 1))
#plt.xlim(xmin-1, xmax+2)
plt.hist2d(x, y, bins=(12,20))  # cmap=plt.cm.jet #cmap=plt.cm.Reds
plt.colorbar()

plt.show()
```




```
In [ ]:
```