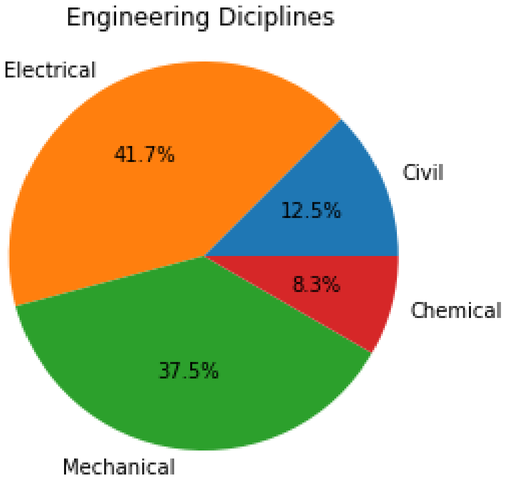
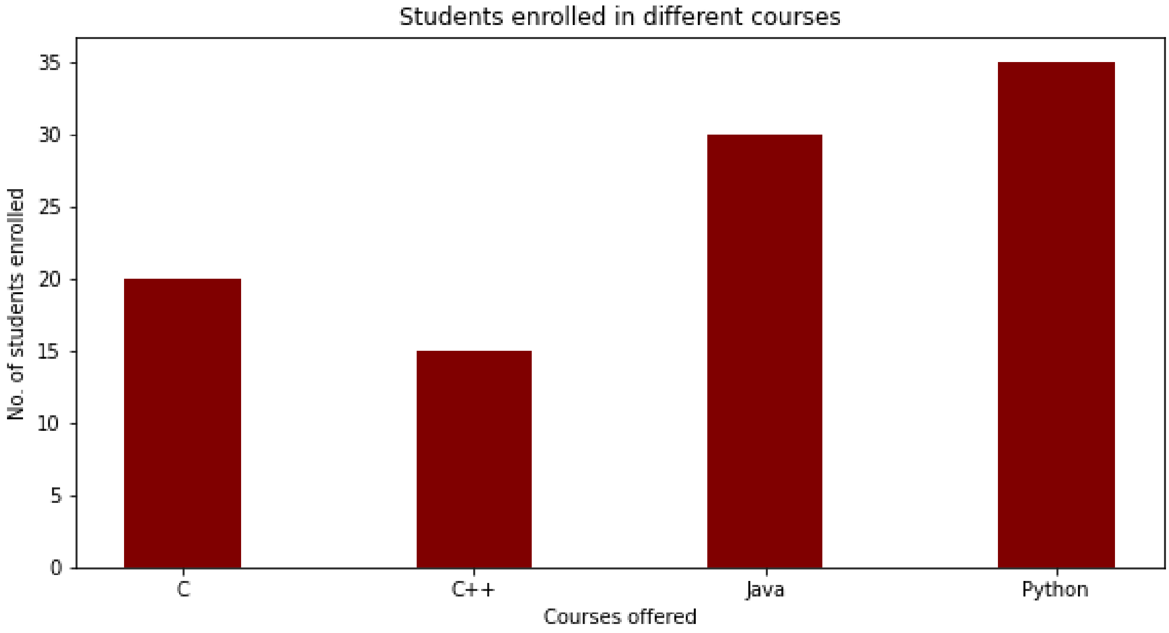
In [39]:

```
import numpy as np
import matplotlib.pyplot as plt
# if using a Jupyter notebook, include:
%matplotlib inline

# Pie chart, where the slices will be ordered and plotted counter-clockwise:
labels = ['Civil', 'Electrical', 'Mechanical', 'Chemical']
sizes = [15, 50, 45, 10]


fig, ax = plt.subplots()
ax.pie(sizes, labels=labels, autopct='%1.1f%%')
ax.axis('equal')  # Equal aspect ratio ensures the pie chart is circular.
ax.set_title('Engineering Diciplines')


plt.show()
```



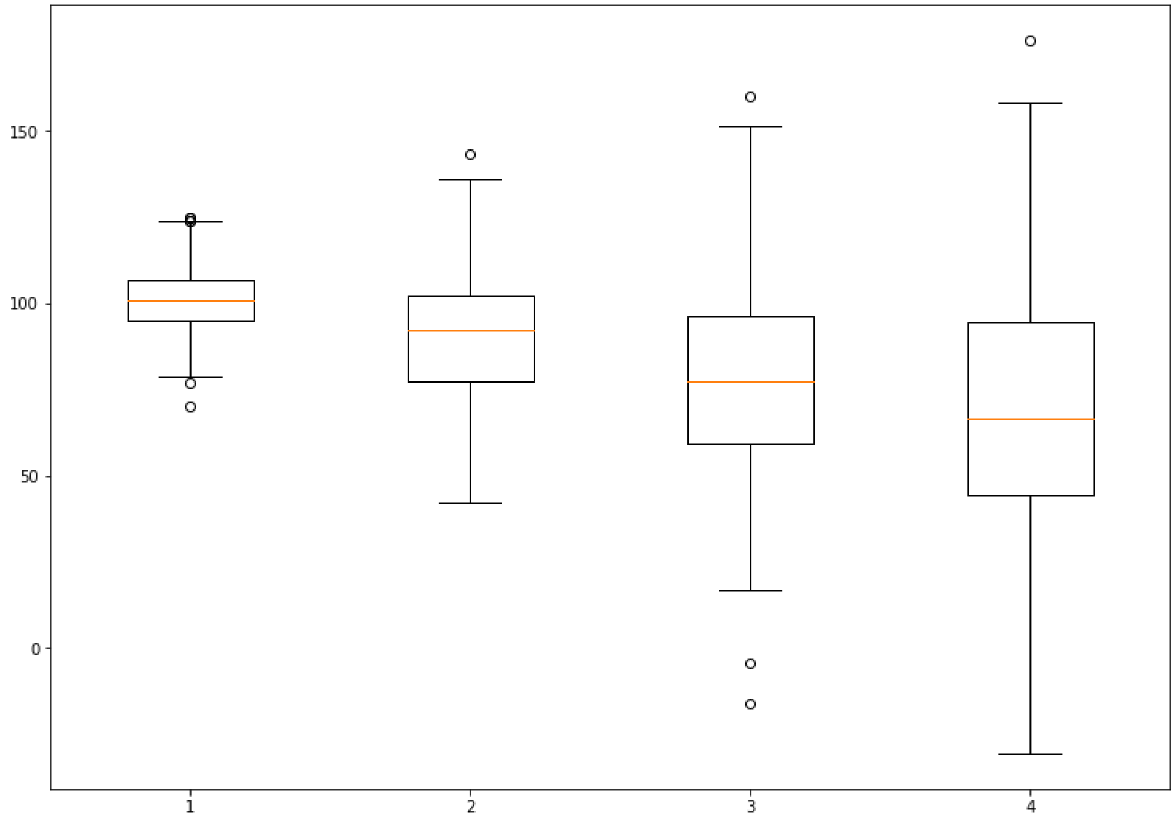
In [40]:

```
import numpy as np
import matplotlib.pyplot as plt


# creating the dataset
data = {'C':20, 'C++':15, 'Java':30,
        'Python':35}
courses = list(data.keys())
values = list(data.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(courses, values, color ='maroon',
        width = 0.4)

plt.xlabel("Courses offered")
plt.ylabel("No. of students enrolled")
plt.title("Students enrolled in different courses")
plt.show()
```



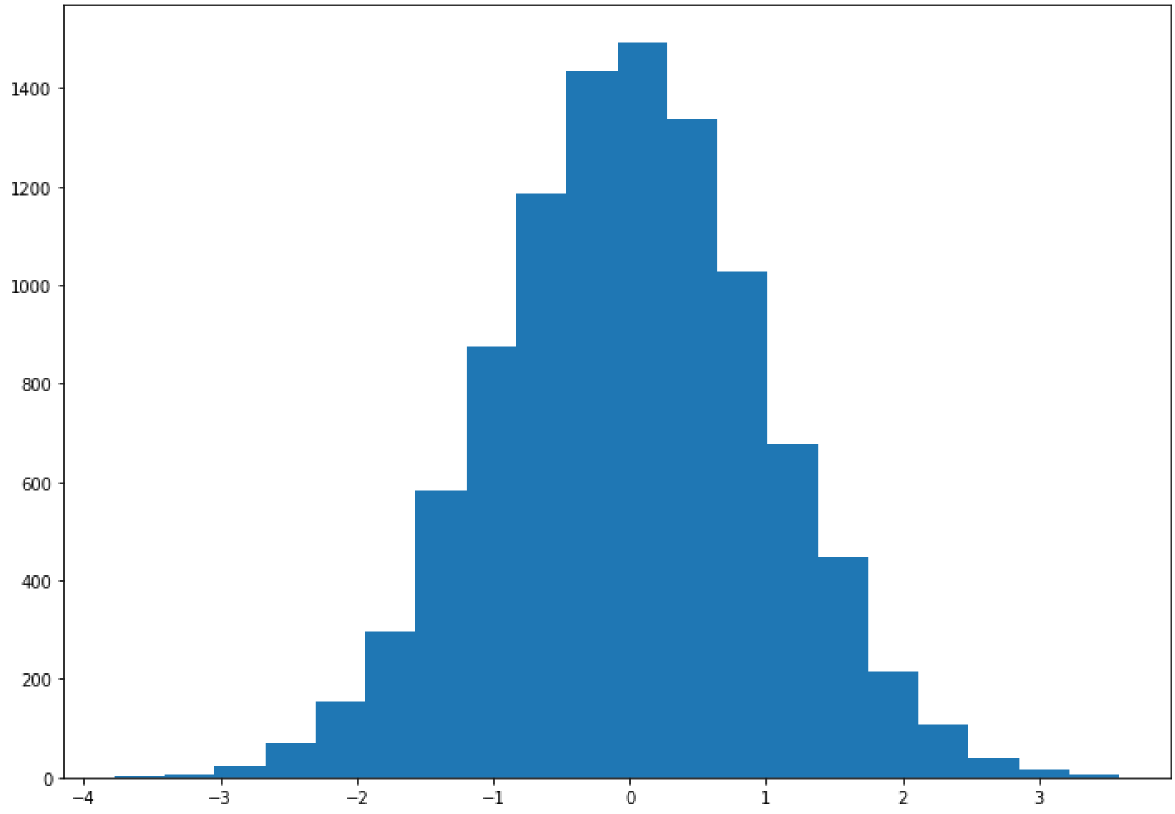
In [41]:

```
# Import libraries
import matplotlib.pyplot as plt
import numpy as np


# Creating dataset
np.random.seed(10)

data_1 = np.random.normal(100, 10, 200)
data_2 = np.random.normal(90, 20, 200)
data_3 = np.random.normal(80, 30, 200)
data_4 = np.random.normal(70, 40, 200)
data = [data_1, data_2, data_3, data_4]

fig = plt.figure(figsize =(10, 7))

# Creating axes instance
ax = fig.add_axes([0, 0, 1, 1])

# Creating plot
bp = ax.boxplot(data)

# show plot
plt.show()
```



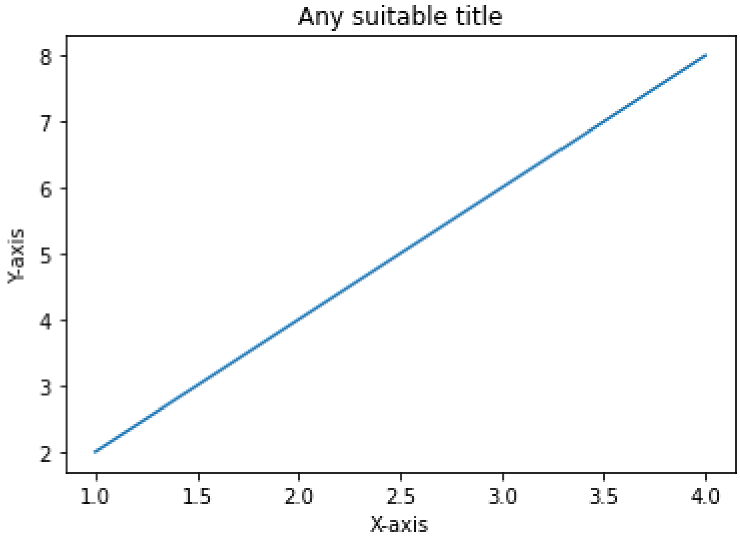
In [42]:

```
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import colors
from matplotlib.ticker import PercentFormatter

# Creating dataset
np.random.seed(23685752)
N_points = 10000
n_bins = 20

# Creating distribution
x = np.random.randn(N_points)
y = .8 ** x + np.random.randn(10000) + 25

# Creating histogram
fig, axs = plt.subplots(1, 1,
                        figsize =(10, 7),
                        tight_layout = True)

axs.hist(x, bins = n_bins)

# Show plot
plt.show()
```



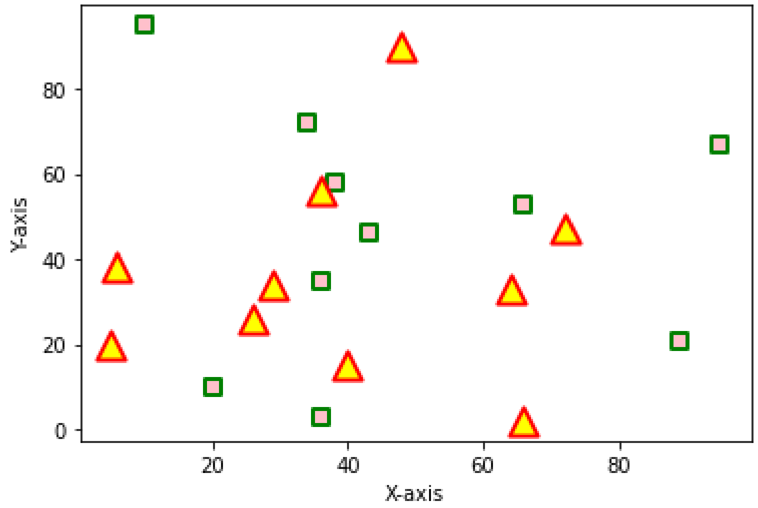
In [43]:

```
import matplotlib.pyplot as plt
import numpy as np


# Define X and Y variable data
x = np.array([1, 2, 3, 4])
y = x*2

plt.plot(x, y)
plt.xlabel("X-axis") # add X-axis label
plt.ylabel("Y-axis") # add Y-axis label
plt.title("Any suitable title") # add title
plt.show()
```



In [44]:

```
import matplotlib.pyplot as plt

# dataset-1
x1 = [89, 43, 36, 36, 95, 10,
    66, 34, 38, 20]

y1 = [21, 46, 3, 35, 67, 95,
    53, 72, 58, 10]

# dataset2
x2 = [26, 29, 48, 64, 6, 5,
    36, 66, 72, 40]

y2 = [26, 34, 90, 33, 38,
    20, 56, 2, 47, 15]

plt.scatter(x1, y1, c ="pink",
            linewidths = 2,
            marker ="s",
            edgecolor ="green",
            s = 50)

plt.scatter(x2, y2, c ="yellow",
            linewidths = 2,
            marker ="^",
            edgecolor ="red",
            s = 200)

plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```

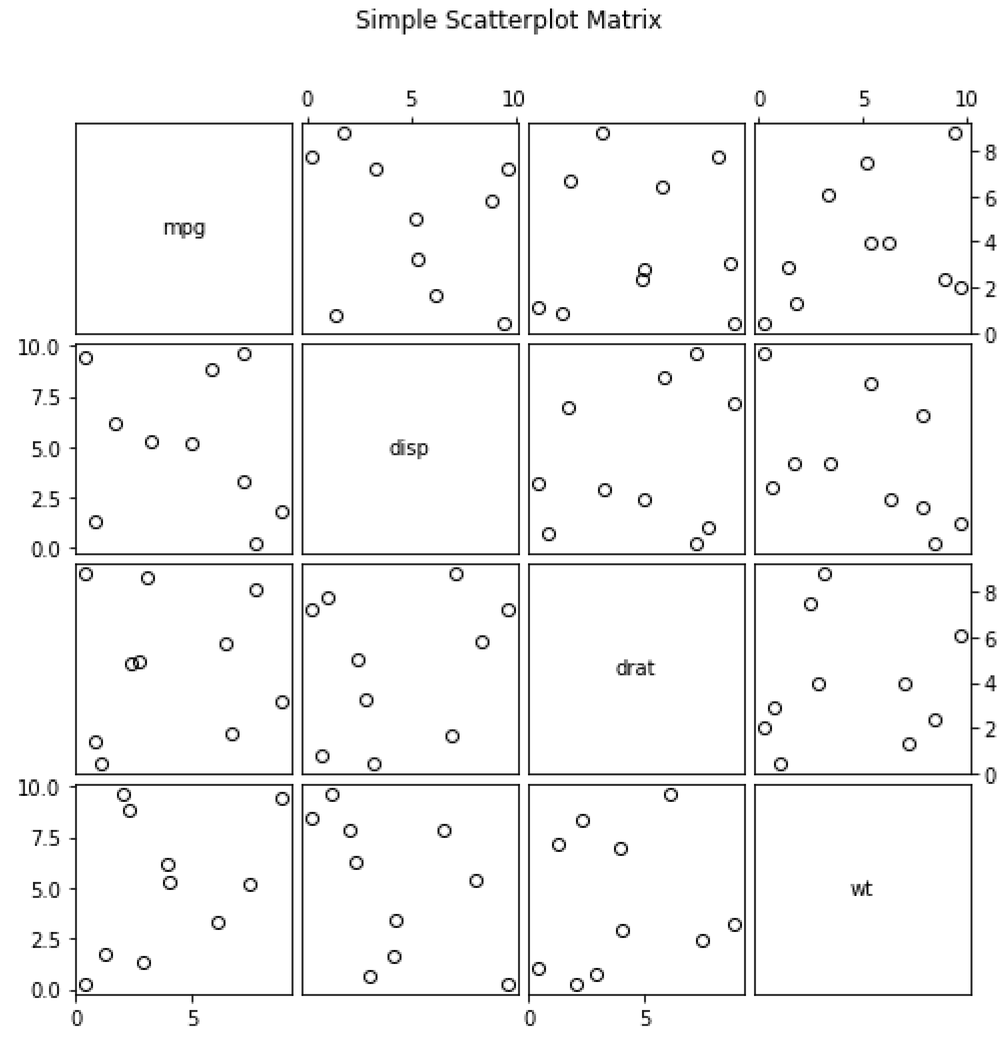


In [48]:

```
import itertools
import numpy as np
import matplotlib.pyplot as plt

def main():
    np.random.seed(1977)
    numvars, numdata = 4, 10
    data = 10 * np.random.random((numvars, numdata))
    fig = scatterplot_matrix(data, ['mpg', 'disp', 'drat', 'wt'],
            linestyle='none', marker='o', color='black', mfc='none')
    fig.suptitle('Simple Scatterplot Matrix')
    plt.show()

def scatterplot_matrix(data, names, **kwargs):
    """Plots a scatterplot matrix of subplots.  Each row of "data" is plotted
    against other rows, resulting in a nrows by nrows grid of subplots with the
    diagonal subplots labeled with "names".  Additional keyword arguments are
    passed on to matplotlib's "plot" command. Returns the matplotlib figure
    object containg the subplot grid."""
    numvars, numdata = data.shape
    fig, axes = plt.subplots(nrows=numvars, ncols=numvars, figsize=(8,8))
    fig.subplots_adjust(hspace=0.05, wspace=0.05)

    for ax in axes.flat:
        # Hide all ticks and labels
        ax.xaxis.set_visible(False)
        ax.yaxis.set_visible(False)

        # Set up ticks only on one side for the "edge" subplots...
        if ax.is_first_col():
            ax.yaxis.set_ticks_position('left')
        if ax.is_last_col():
            ax.yaxis.set_ticks_position('right')
        if ax.is_first_row():
            ax.xaxis.set_ticks_position('top')
        if ax.is_last_row():
            ax.xaxis.set_ticks_position('bottom')

    # Plot the data.
    for i, j in zip(*np.triu_indices_from(axes, k=1)):
        for x, y in [(i,j), (j,i)]:
            axes[x,y].plot(data[x], data[y], **kwargs)

    # Label the diagonal subplots...
    for i, label in enumerate(names):
        axes[i,i].annotate(label, (0.5, 0.5), xycoords='axes fraction',
                ha='center', va='center')

    # Turn on the proper x or y axes ticks.
    for i, j in zip(range(numvars), itertools.cycle((-1, 0))):
        axes[j,i].xaxis.set_visible(True)
        axes[i,j].yaxis.set_visible(True)

    return fig

main()
```

```
/tmp/ipykernel_6468/1909682182.py:30: MatplotlibDeprecationWarning:
```

```
The is_first_col function was deprecated in Matplotlib 3.4 and will
be removed two minor releases later. Use ax.get_subplotspec().is_fir
st_col() instead.
  if ax.is_first_col():
/tmp/ipykernel_6468/1909682182.py:32: MatplotlibDeprecationWarning:
The is_last_col function was deprecated in Matplotlib 3.4 and will b
e removed two minor releases later. Use ax.get_subplotspec().is_last
_col() instead.
  if ax.is_last_col():
/tmp/ipykernel_6468/1909682182.py:34: MatplotlibDeprecationWarning:
The is_first_row function was deprecated in Matplotlib 3.4 and will
be removed two minor releases later. Use ax.get_subplotspec().is_fir
st_row() instead.
  if ax.is_first_row():
/tmp/ipykernel_6468/1909682182.py:36: MatplotlibDeprecationWarning:
The is_last_row function was deprecated in Matplotlib 3.4 and will b
e removed two minor releases later. Use ax.get_subplotspec().is_last
_row() instead.
  if ax.is_last_row():
```


Simple Scatterplot Matrix
mpg
disp
drat
wt