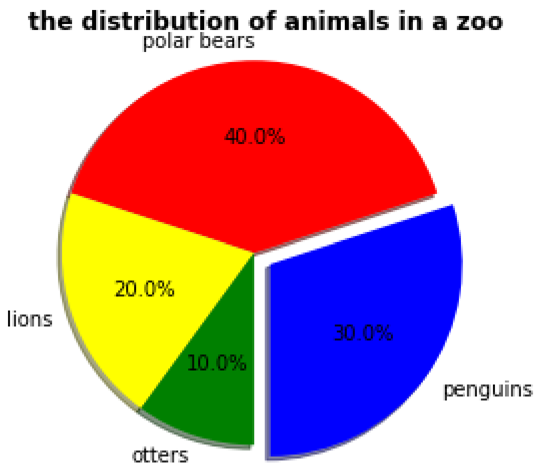
Thasina Tabashum

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
```

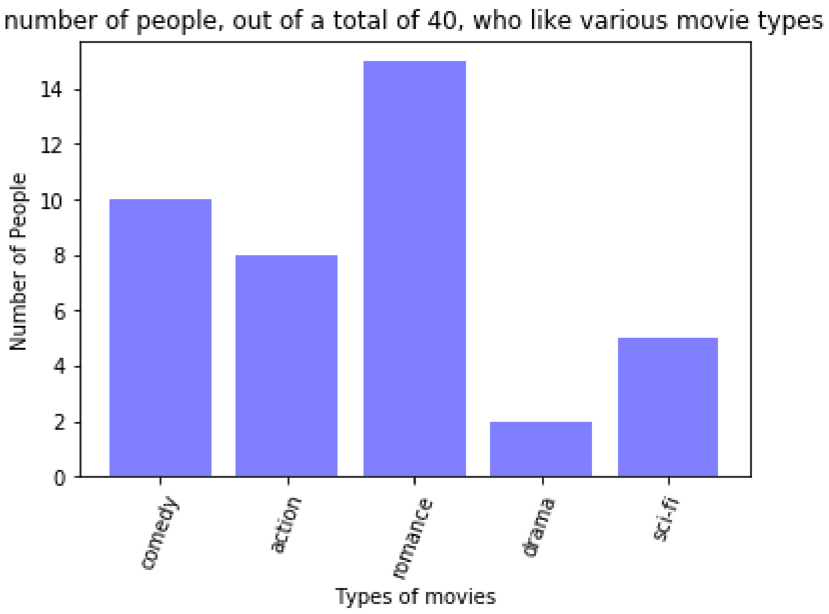
1. Pie Chart:

In [13]:
```
labels = ['penguins','polar bears','lions','otters']
sizes = [30,40,20,10]
colors = ['blue', 'red', 'yellow','green']
explode = [0.1,0,0,0]

#plotting the pie chart
plt.pie(sizes, explode=explode, labels=labels, colors=colors,autopct='%1.1f%%',sh
plt.title(' the distribution of animals in a zoo',fontweight='bold')
plt.axis('equal')
plt.show()
```



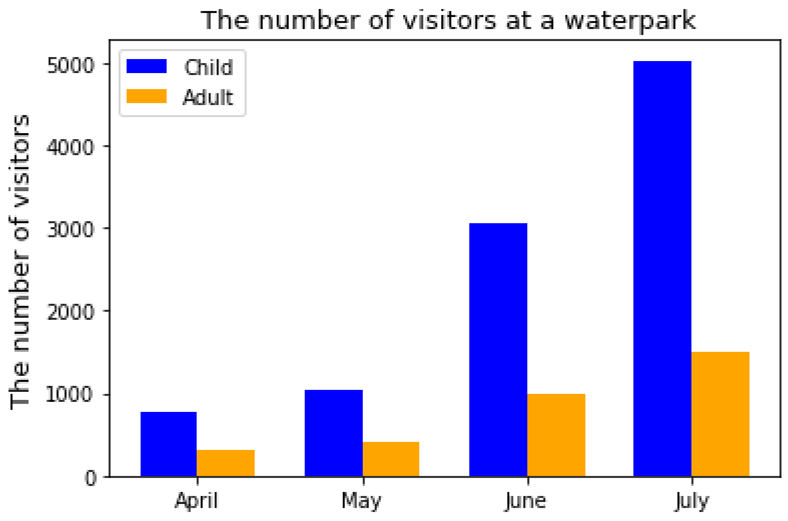
2. Bar Chart:

In [14]:
```
objects = ('comedy', 'action', 'romance', 'drama','sci-fi')
y_pos = np.arange(len(objects))
performance = [10,8,15,2,5]

#Create bar plot with labels, a title, and color
plt.bar(y_pos, performance, align='center', alpha=0.5,color='blue')
plt.xticks(y_pos, objects, rotation='70')
plt.ylabel('Number of People')
plt.xlabel('Types of movies')
plt.title('number of people, out of a total of 40, who like various movie types'

#Display the bar plot
plt.show()
```

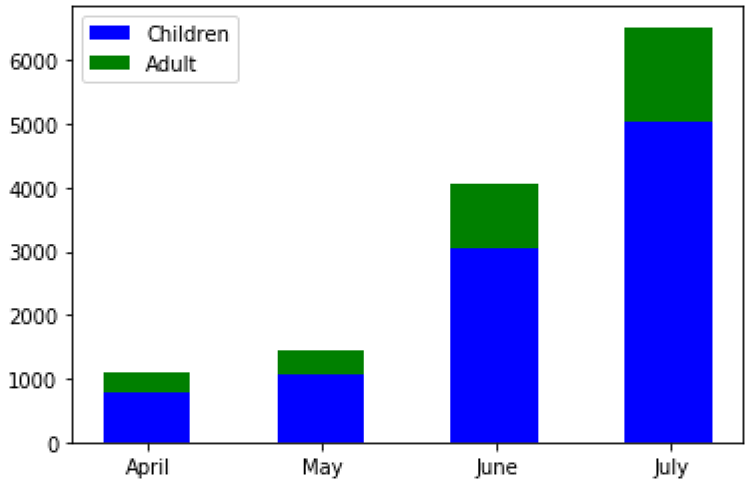


3. Grouped Bar Plots:

In [15]:
```
#First set of data
children = (780, 1050, 3056, 5025)
ind = np.arange(4)  # the x locations for the groups
width = 0.35       # the width of the bars
fig, ax = plt.subplots()

#Creating the first set of bars for the first set of data
rects1 = ax.bar(ind, children, width, color='blue')

#Second set of data-- creation of bars
adult = (315, 400, 1000, 1500)
rects2 = ax.bar(ind + width, adult, width, color='orange')

# Add some text for labels, title and axes ticks
ax.set_ylabel('The number of visitors',fontsize=13)
ax.set_title(' The number of visitors at a waterpark',fontsize=13)
ax.set_xticks(ind + width / 2)
ax.set_xticklabels(('April', 'May', 'June', 'July'),fontsize=10)

#Create a legend
ax.legend((rects1[0], rects2[0]), ('Child', 'Adult'))

plt.show()
```

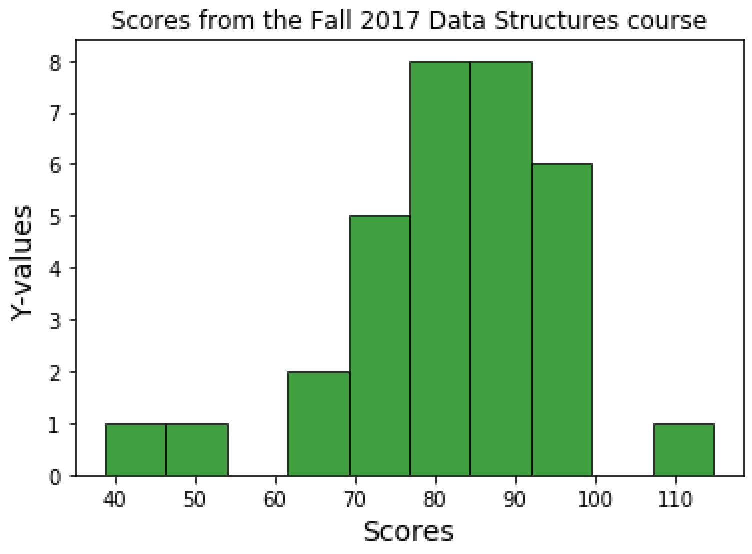


In [16]:
```
#Data for the first and second bar plots
children = (780, 1050, 3056, 5025)
adult = (315, 400, 1000, 1500)

#Setting the width of the bars and tick positions
width = 0.5      # the width of the bars
ind = np.arange(1,5)  # the x locations for the groups
tick_pos = [i + (width/50) for i in ind]

#Creating the bars
p1 = plt.bar(ind, children, width, color='blue',align='center')
p2 = plt.bar(ind,adult,width,bottom=children,color='green',align='center')

# Add some text for labels, title and axes ticks
ax.set_ylabel('The number of visitors',fontsize=13)
ax.set_title(' The number of visitors at a waterpark',fontsize=13)
plt.xticks(tick_pos,('April', 'May', 'June', 'July'), fontsize=10)
ax.set_xticklabels(('April', 'May', 'June', 'July'),fontsize=10)
plt.legend((p1[0], p2[0]), ('Children', 'Adult'),loc="best")


plt.show()
```



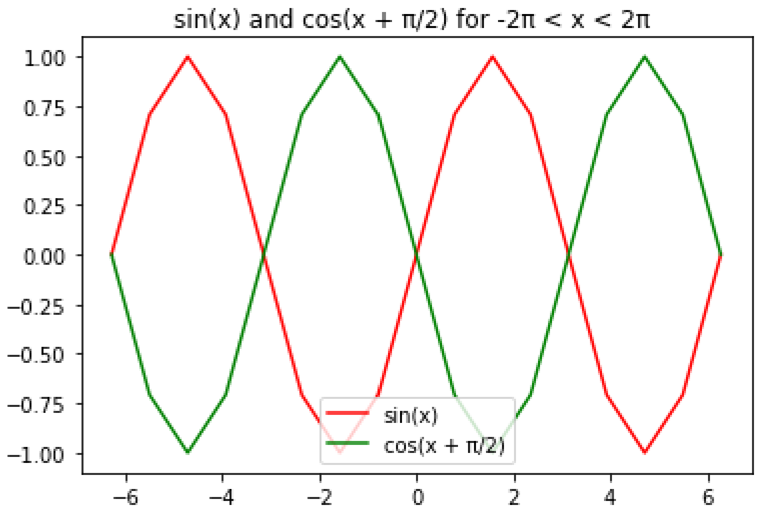
4. Histogram:

In [7]:
```
x = [114.8, 98.8, 97.3, 96, 94.1, 93.1, 93.1, 91.6, 91.5, 91.3, 90.3, 89.2, 87.5
     72, 71, 64.6, 63.3, 47.2, 38.7]
num_bins =10
n, bins, patches = plt.hist(x, num_bins, facecolor='green', alpha=0.75, edgecolo

plt.title('Scores from the Fall 2017 Data Structures course')
plt.xlabel('Scores',fontsize=14)
plt.ylabel('Y-values',fontsize=14)
plt.savefig("D:\Courses\BigData\histogram.pdf",bbox_inches='tight')
plt.savefig("D:\Courses\BigData\histogram.png",bbox_inches='tight')
plt.show()
```



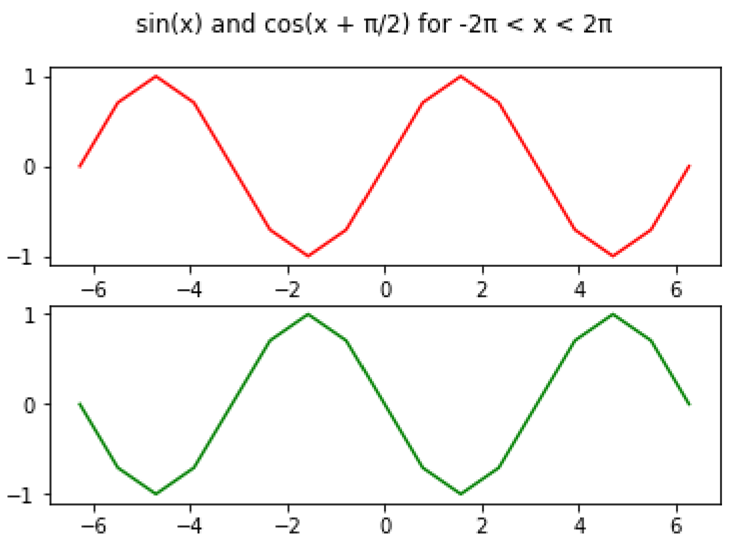
1. Line Plot:

In [18]:
```
#Varying y-values and x-values
x = [ -6.283, -5.498, -4.712, -3.927, -3.142, -2.356,
              -1.571, -.7854, 0, .7854, 1.571, 2.356, 3.142, 3.927, 4.712, 5.498


sin_x= [ 0, .70711, 1, .70711, 0, -.70711, -1, -.70711, 0, .70711, 1, .70711, 0,
cos_x = [0, -.70711, -1, -.70711, 0, .70711, 1, .70711, 0, -.70711, -1, -.70711,


#Plotting the two different lines
plt.plot(x,sin_x,color='r')
plt.plot(x,cos_x,color='g')

#Putting a title and labels
plt.title("sin(x) and cos(x + π/2) for -2π < x < 2π ")

plt.legend(['sin(x)','cos(x + π/2)'],loc = 'lower center')

plt.show()
```



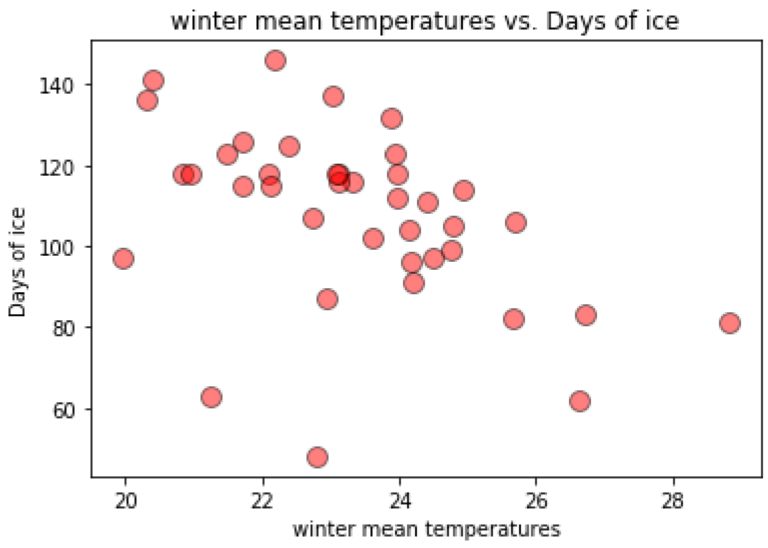
In [19]:
```
fig, axs = plt.subplots(2)
fig.suptitle('sin(x) and cos(x + π/2) for -2π < x < 2π')
axs[0].plot(x, sin_x,color='r')
axs[1].plot(x, cos_x,color='g')
```

Out[19]:
```
[<matplotlib.lines.Line2D at 0x1b9567529c8>]
```



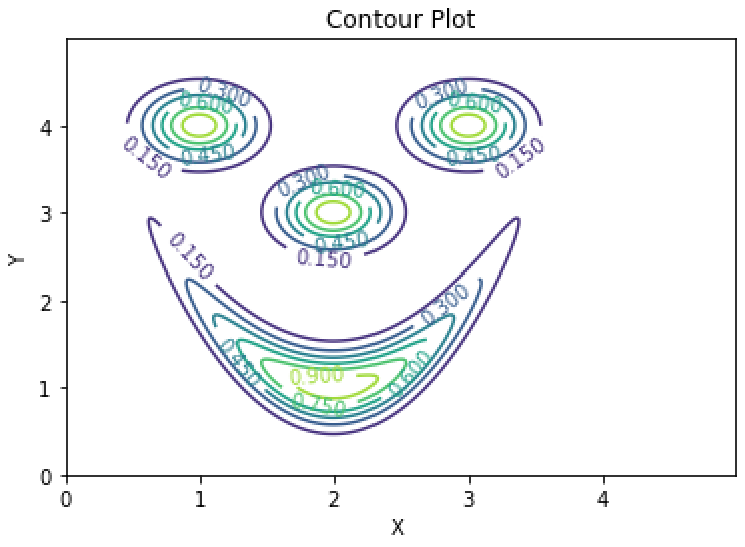
2. Scatter Plot:

In [20]:
```
# Fixing random state for reproducibility
np.random.seed(19680801)

#Getting x and y values
temperatures = [22.94, 23.02, 25.68, 19.96, 24.80, 23.98, 22.10, 20.30, 24.20, 2
                24.16, 24.94, 22.40, 22.14, 20.84, 25.66, 21.73, 24.49, 24.13, 2
                21.73, 20.41, 24.41, 23.95, 20.95, 26.71, 22.81, 23.11, 23.33, 2
                23.11, 21.47, 23.97, 24.75, 23.61, 23.08, 21.24, 26.63, 23.88]
days = [ 87, 137, 106, 97, 105, 118, 118, 136, 91, 107,
         96, 114, 125, 115, 118, 82, 115, 97, 104, 146, 126,
         141, 111, 123, 118, 83, 48, 118, 116, 81, 116, 123, 112, 99, 102, 1

#Creating the scatter plot
plt.scatter(temperatures, days, s=100, facecolors ='r', edgecolors='black', alph

#Labelling the plot
plt.xlabel('winter mean temperatures')
plt.ylabel('Days of ice')
plt.title('winter mean temperatures vs. Days of ice')

#Outputting the scatter plot
plt.show()
```

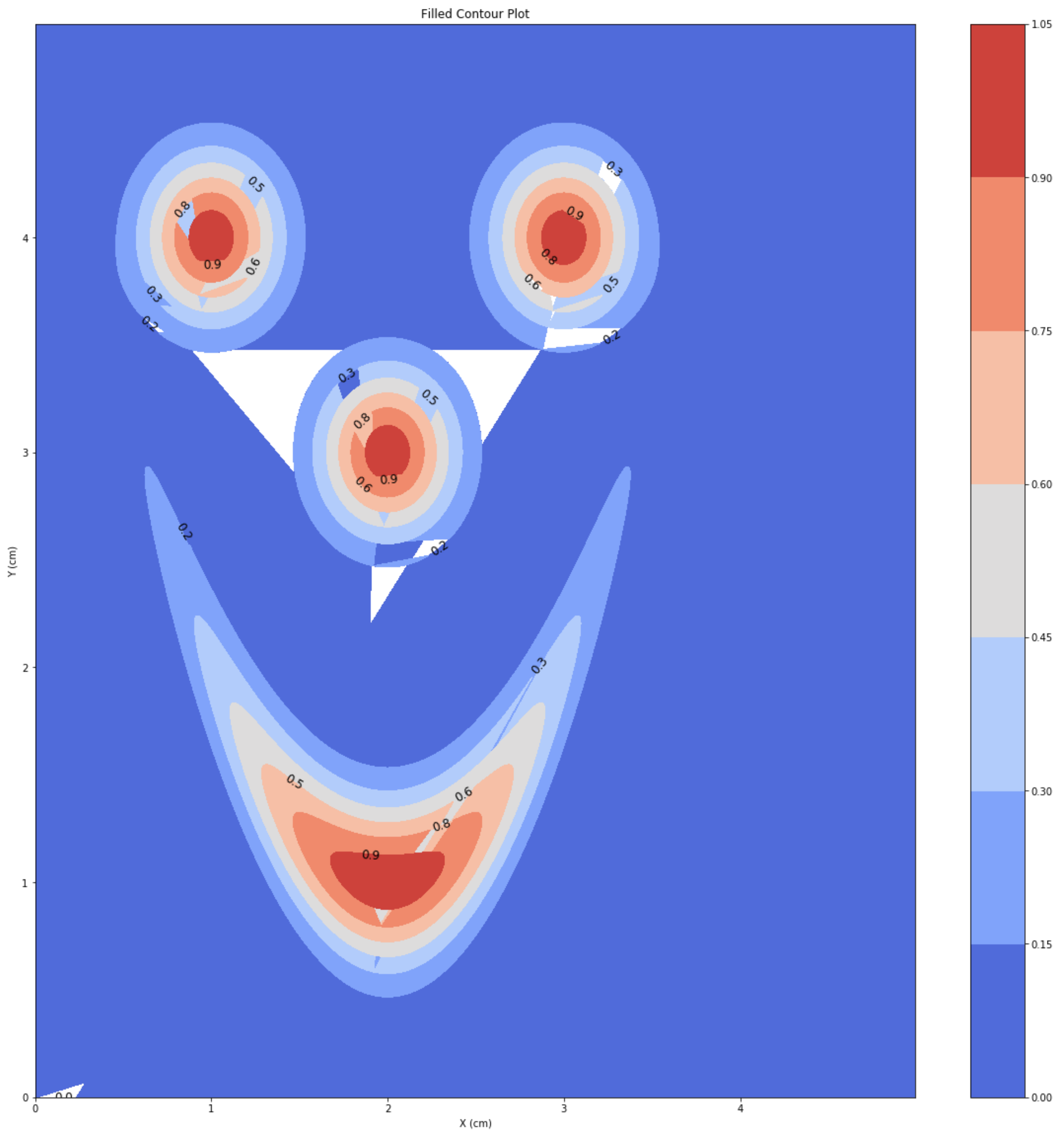


1. Contour plot

In [21]:
```
#Creates an unfilled contour with a cool & warm color mapping
import matplotlib.mlab as mlab
#Following line is for color mapping
from matplotlib import cm
#Creating the data for the contour plot(s)
delta = 0.015
x = np.arange(0.0,5.0,delta)
y = np.arange(0.0,5.0,delta)
#meshgrid makes rectangular arrays that
#cover every combination of x and y values
X, Y = np.meshgrid(x,y)


Z = np.exp(-((X-1)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-3)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-2)**2+(Y-3)**2)/0.15) + \
    np.exp(-(X-2)**2) * np.exp(-(Y - ((X-2)**2+1))**2/0.15)



#Creating an unfilled contour plot
plt.figure()
ax.set_ylim(Y.min(),Y.max())
ax.set_xlim(X.min(),X.max())
#Can change the color mappings and line style
cp = plt.contour(X, Y, Z)

#Labels and titles for the plot
plt.clabel(cp, inline=True, fontsize=10)
plt.title('Contour Plot')
plt.xlabel('X ')
plt.ylabel('Y ')
plt.show()
```

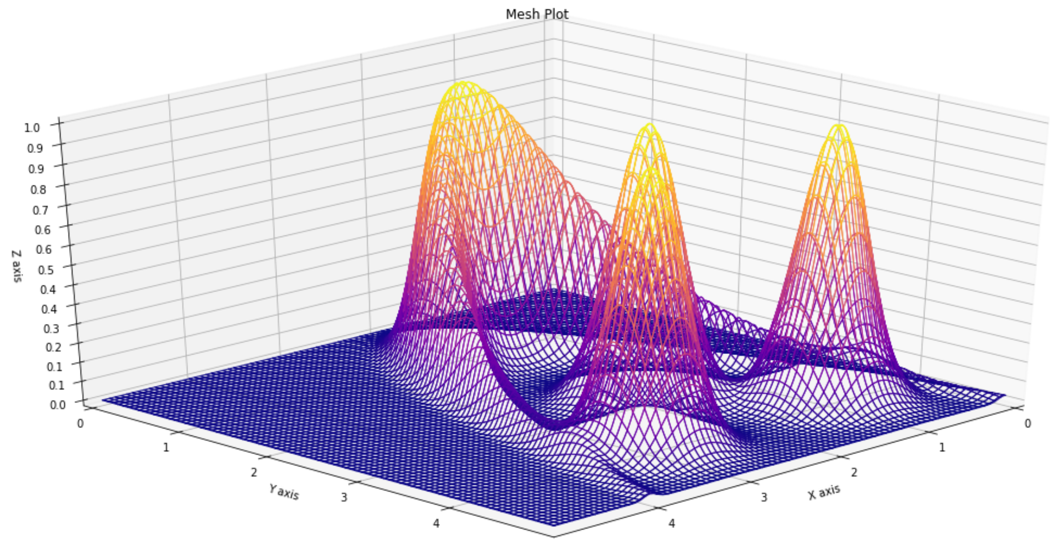


In [22]:
```
#Creates a filled contour with a cool & warm color mapping
fig = plt.figure(figsize=(20,20))
#Creating the data for the contour plot(s)
delta = 0.015
x = np.arange(0.0,5.0,delta)
y = np.arange(0.0,5.0,delta)
#meshgrid makes rectangular arrays that
#cover every combination of x and y values
X, Y = np.meshgrid(x,y)


Z = np.exp(-((X-1)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-3)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-2)**2+(Y-3)**2)/0.15) + \
    np.exp(-(X-2)**2) * np.exp(-(Y - ((X-2)**2+1))**2/0.15)

#print(Z) -- Use this statement if you want to get the Z-values
#Creating a filled contour plot

#Creating the contour plot with the data values and creating labels
#Filling the contour plot with designated colors
contour_filled = plt.contourf(X,Y,Z,cmap=cm.coolwarm)

#Creating a legend, title, and labels
plt.clabel(contour_filled, colors='k',fmt = '%2.1f',fontsize=12)
plt.colorbar(contour_filled)
plt.title('Filled Contour Plot')
plt.xlabel('X (cm)')
plt.ylabel('Y (cm)')
plt.show()
```

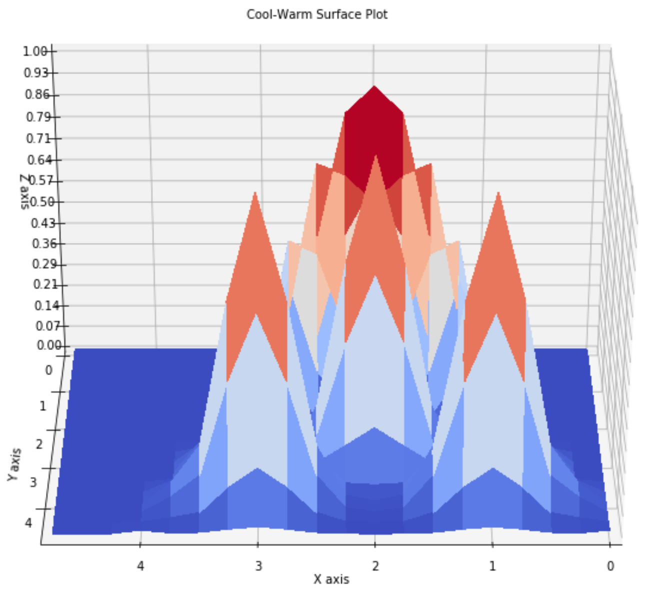


2. Surface plots (or mesh plots)

In [37]:
```
import numpy as np
from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
fig = plt.figure(figsize=(20,10))
delta = 0.015
x = np.arange(0.0,5.0,delta)
y = np.arange(0.0,5.0,delta)
#meshgrid makes rectangular arrays that
#cover every combination of x and y values
X, Y = np.meshgrid(x,y)


Z = np.exp(-((X-1)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-3)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-2)**2+(Y-3)**2)/0.15) + \
    np.exp(-(X-2)**2) * np.exp(-(Y - ((X-2)**2+1))**2/0.15)
Z = (Z-Z.min())/(Z.max()-Z.min())

#Color mapping
colors = cm.plasma(Z) #Can customize with various color mapping styles,rcount,cc

# Plot a basic wireframe.
#rstride = arrayrowstride(step size)
#cstrode = arraycolumnstride(step size)
ax = fig.gca(projection='3d')

#Setting rotation angle of plot to 45 degrees
ax.view_init(azim=45)

#rccount= max amount of rows
#ccount = max amount of columns

surf= ax.plot_surface(X, Y, Z, rcount=100, ccount=100,facecolors=colors, shade=F
#wire = ax.plot_wireframe(X, Y, Z, rstride=100, cstride=100)
#Setting the tick locators-- choosing tick locations
ax.set_zlim(Z.min(), Z.max())
ax.set_ylim(Y.min(),Y.max())
ax.set_xlim(X.min(),X.max())
#Creates evenly spaced ticks from min to max
ax.zaxis.set_major_locator(LinearLocator(15))
#sprintf format string
ax.zaxis.set_major_formatter(FormatStrFormatter('%.1f'))

#Add labels for x,y, and z axes
ax.set_xlabel('X axis', fontsize=10,labelpad=4.2)
ax.set_ylabel('Y axis', fontsize = 10,labelpad=4.2)
ax.set_zlabel('Z axis',fontsize=10,labelpad = 4.2)
ax.set_title('Mesh Plot')
surf.set_facecolor((0,0,0,0))
plt.show()
```


Mesh Plot
Z axis
1.0
0.9
0.9
0.8
0.7
0.6
0.6
0.5
0.4
0.4
0.3
0.2
0.1
0.1
0.0
Y axis
0
1
2
3
4
X axis
0
1
2
3
4

In [41]:
```
#Demonstrates plotting a 3D surface colored with the cool-warm color map.
#Surface plot is viewed at a 45-degree angle

#Creating the figure
fig = plt.figure(figsize=(20,10))
ax = fig.gca(projection='3d') #creating axes

#Setting rotation angle of plot


delta = 0.25
x = np.arange(0.0,5.0,delta)
y = np.arange(0.0,5.0,delta)
#meshgrid makes rectangular arrays that
#cover every combination of x and y values
X, Y = np.meshgrid(x,y)


Z = np.exp(-((X-1)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-3)**2+(Y-4)**2)/0.15) + \
    np.exp(-((X-2)**2+(Y-3)**2)/0.15) + \
    np.exp(-(X-2)**2) * np.exp(-(Y - ((X-2)**2+1))**2/0.15)
Z = (Z-Z.min())/(Z.max()-Z.min())
surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,linewidth=10, antialiased=False
ax.view_init(azim=90)
# Customize the z axis.
ax.set_zlim(Z.min(), Z.max())
ax.set_ylim(Y.min(),Y.max())
ax.set_xlim(X.min(),X.max())
#Setting the tick locators-- choosing tick locations
#Creates evenly spaced ticks from min to max
ax.zaxis.set_major_locator(LinearLocator(15))
#sprintf format string
ax.zaxis.set_major_formatter(FormatStrFormatter('%.02f'))

# Add a color bar which maps values to colors.
fig.colorbar(surf,shrink=0.01,aspect =5,pad=0.09)

#Add labels for x,y, and z axes
ax.set_xlabel('X axis',fontsize=10)
ax.set_ylabel('Y axis',fontsize=10)
ax.set_zlabel('Z axis',fontsize=10)
ax.set_title('Cool-Warm Surface Plot',fontsize=10)
plt.show()
```


Cool-Warm Surface Plot
Z axis
1.00
0.93
0.86
0.79
0.71
0.64
0.57
0.50
0.43
0.36
0.29
0.21
0.14
0.07
0.00
Y axis
0
1
2
3
4
X axis
4
3
2
1
0