In [257]:
```python
import numpy as np
from matplotlib import pyplot as plt
import scipy as sp
import pandas as pd
```

1. Use `matplotlib.pyplot.plot` to produce a plot of the functions

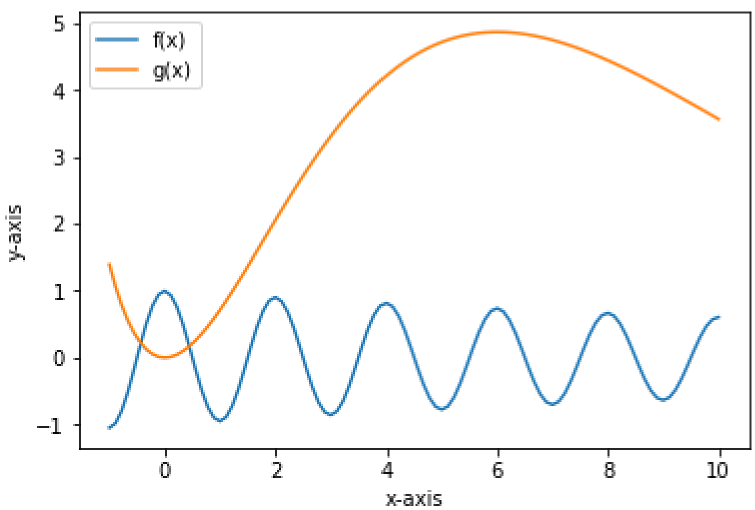
$$f(x) = e^{-x/20}cos(\pi x)$$

and

$$g(x) = x^2 e^{-x/3}$$

over the interval [-1, 10]. Include labels for the x- and y-axes, and a legend explaining which line is which plot.

In [258]:
```python
x = np.linspace(-1, 10, 100)
y1 = (np.exp(-x/20))*(np.cos(np.pi*x))
#plt.xlabel('$x1$')
#plt.ylabel('$(np.exp(-x/20))*(np.cos(np.pi*x))$')

y2 = x*x*(np.exp(-x/3))

plt.plot(x, y1, x, y2)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.legend(['f(x)', 'g(x)'])
plt.show()
```

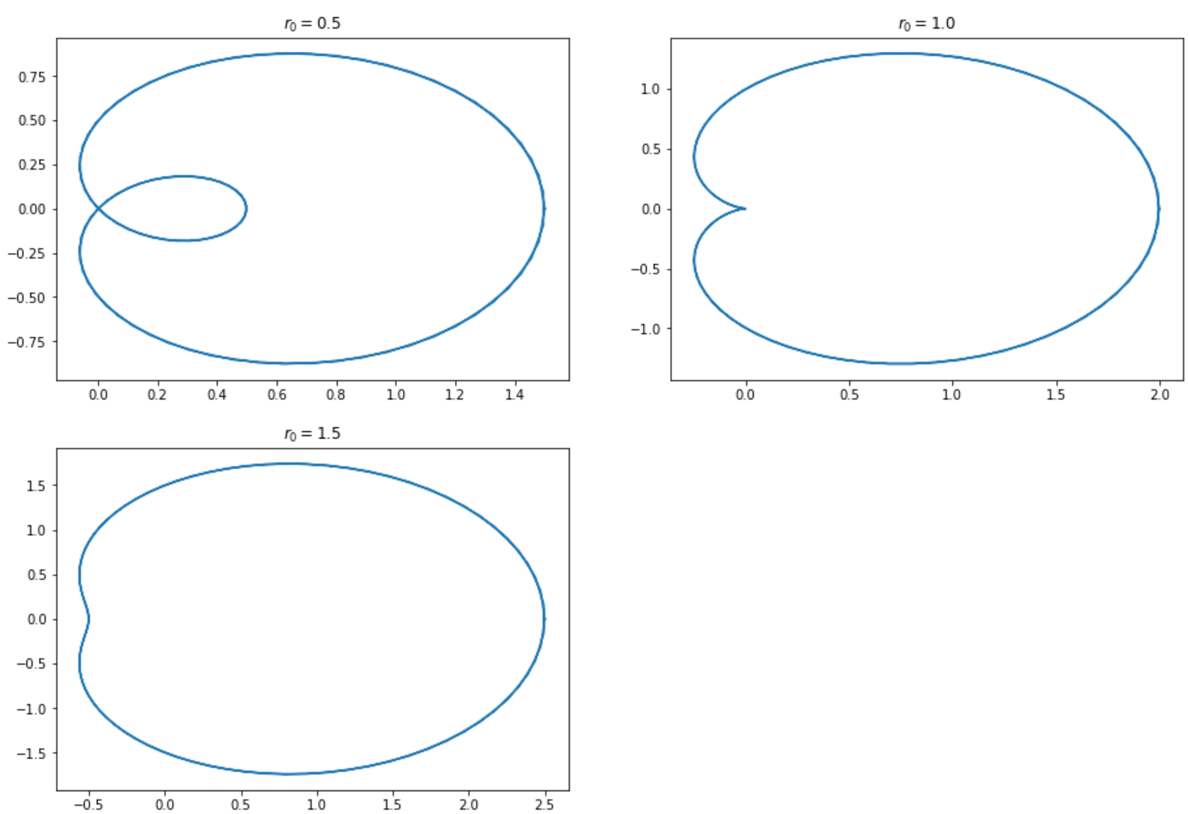


2. The shape of a limacon can be defined parametrically as

$$r = r_0 + cos\theta$$
$$x = rcos\theta$$
$$y = rsin\theta$$

When $r_0 = 1$, this curve is called a cardioid. Use this definition to plot the shape of a lima¸con for $r_0 = 0.5$, $r_0 = 1.0$, and $r_0 = 1.5$. Put these three plots in three subgraphs (with 2*2 layout), give them titles ($r_0 = 0.5$, $r_0 = 1.0$, and $r_0 = 1.5$), adjust the spaces between subgraphs to make sure the texts don't overlap with each other.

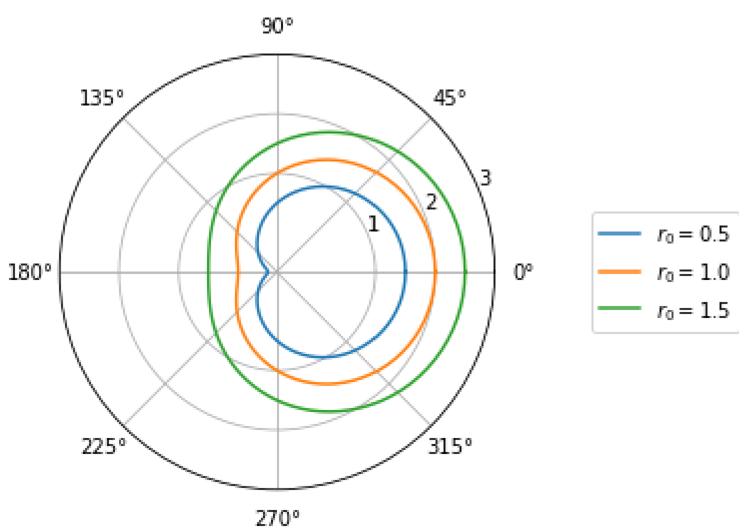
In [259]:
```
theta = np.linspace(-2*np.pi, 2*np.pi, 100)

r01 = 0.5
r1 = r01 + np.cos(theta)
x1 = r1*np.cos(theta)
y1 = r1*np.sin(theta)
plt.subplot(2, 2, 1)
plt.plot(x1, y1)
plt.gca().set_title('$r_0 = 0.5$')

r02 = 1.0
r2 = r02 + np.cos(theta)
x2 = r2*np.cos(theta)
y2 = r2*np.sin(theta)
plt.subplot(2, 2, 2)
plt.plot(x2, y2)
plt.gca().set_title('$r_0 = 1.0$')

r03 = 1.5
r3 = r03 + np.cos(theta)
x3 = r3*np.cos(theta)
y3 = r3*np.sin(theta)
plt.subplot(2, 2, 3)
plt.plot(x3, y3)
plt.gca().set_title('$r_0 = 1.5$')

plt.subplots_adjust(left=10, right=12, bottom=10, top=12)
plt.show()
```



Draw another graph with those three limacon in one **polar** coordinate system.

In [260]:

```
theta = np.linspace(0, 2*np.pi, 100)

r01 = 0.5
r1 = r01 + np.cos(theta)

r02 = 1.0
r2 = r02 + np.cos(theta)

r03 = 1.5
r3 = r03 + np.cos(theta)

plt.subplot(111, projection='polar')
plt.plot(theta, r1, theta, r2, theta, r3)
plt.gca().set_rmax(3)
plt.gca().set_rticks([1.0, 2.0, 3.0])

#plt.gca().set_yticklabels([0, 0.5, 1.0, 1.5, 2.0, 2.5])
#for label in plt.gca().get_yticklabels()[::2]:
#    label.set_visible(True)

plt.legend(['$r_0 = 0.5$', '$r_0 = 1.0$', '$r_0 = 1.5$'],
           loc='center left', bbox_to_anchor=(1.2, 0.5))
plt.show()
```



3. Given the following dataset:

In [261]:
```
titanic = pd.read_csv("titanic.csv")

titanic.head()
```

Out[261]:

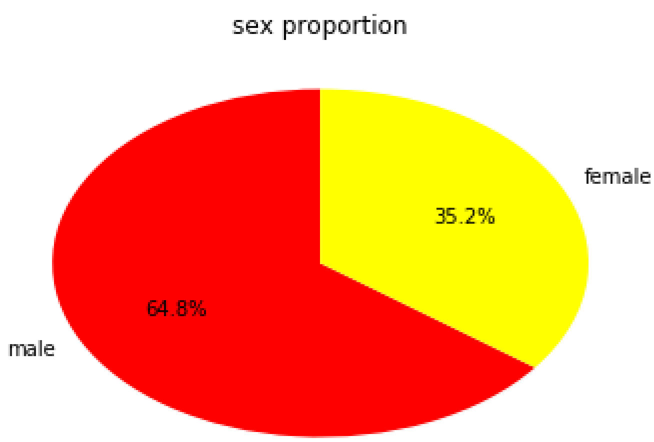
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

Use the `plt.pie` method to make a pie chart of the sex proportion of all the passengers,

Here's the requirements:

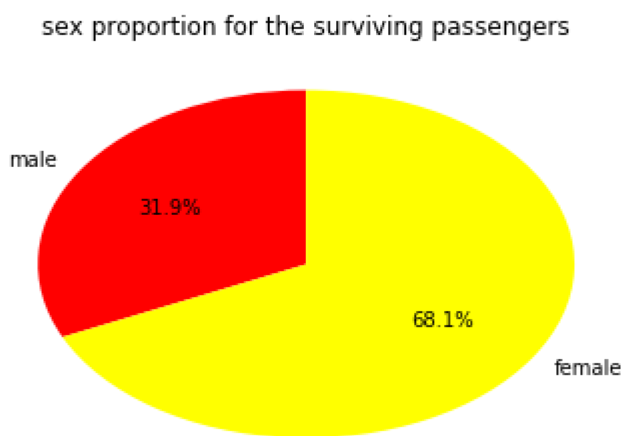
- Label each portion of the pie chart(male, female)
- Male color: red, female color: yellow
- Starting angle at 90 degree
- Percentage number listed on the pie chart.
- Give it a title "sex proportion"

In [262]:
```
female = 0
male = 0

for person in titanic['Sex']:
    if person == 'female':
        female += 1
    if person == 'male':
        male += 1

genders = 'male', 'female'
number = [male, female]
pie_color = ('red', 'yellow')
plt.pie(number, labels=genders, autopct='%1.1f%%',
        colors=pie_color, startangle=90)
plt.gca().set_title("sex proportion")
plt.show()
```



4. Make another pie chart of sex proportion for the survived passengers

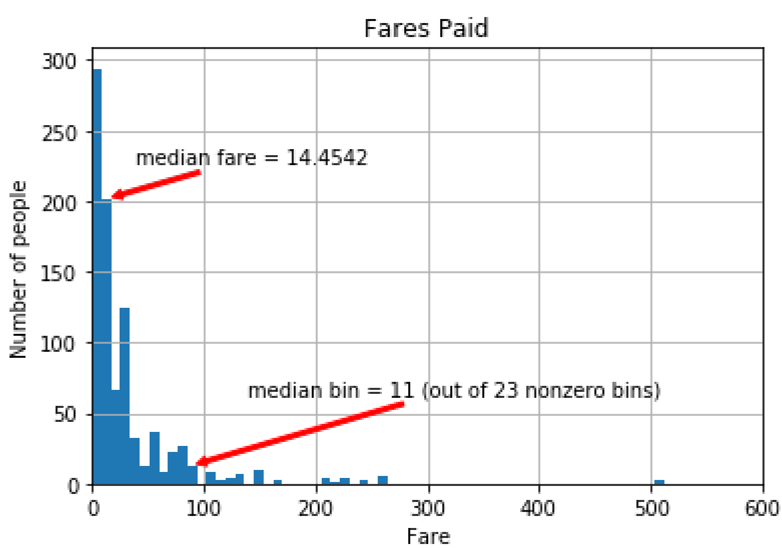
In [263]:
```
female = 0
male = 0

survivors = titanic.loc[titanic['Survived'] == 1]

for person in survivors['Sex']:
    if person == 'female':
        female += 1
    if person == 'male':
        male += 1

genders = 'male', 'female'
number = [male, female]
pie_color = ('red', 'yellow')
plt.pie(number, labels=genders, autopct='%1.1f%%',
        colors=pie_color, startangle=90)
plt.gca().set_title("sex proportion for the surviving passengers")
plt.show()
```



5. Make a histogram of the fares paid,

- Use bins from 0 to 600, binsize=10
- Give x, y labels, and a title
- annotate the median bin with text and arrow, text='median = (number)'

In [264]:
```
fares = titanic['Fare'].values
medianValue = np.median(fares)
loc = (medianValue, 202)
bin_size = 10;

myPlot = plt.hist(fares, 60)
plt.grid(True)
plt.xlim([0, 600])
plt.xlabel('Fare')
plt.ylabel('Number of people')
plt.gca().set_title("Fares Paid")
plt.annotate('median fare = ' + str(medianValue), xy=loc,
             xytext=loc+np.array([25, 25]),
            arrowprops=dict(color='red', arrowstyle='simple'))

howManyBars = 0
for i in range(60):
    if myPlot[0][i] != 0:
        #used to manually find each y value (height)
        #plt.text(myPlot[1][i], myPlot[0][i], str(myPlot[0][i]))
        howManyBars += 1

#where = 10
#plt.text(myPlot[1][where], myPlot[0][where], str(myPlot[0][where]))

medianBin = int(howManyBars/2)
mbloc = (bin_size * (medianBin - 2), 13)
plt.annotate('median bin = ' + str(medianBin) + ' (out of ' + str(howManyBars)
+ ' nonzero bins)', xy=mbloc,
            xytext=mbloc+np.array([50, 50]),
            arrowprops=dict(color='red', arrowstyle='simple'))

plt.show()
```



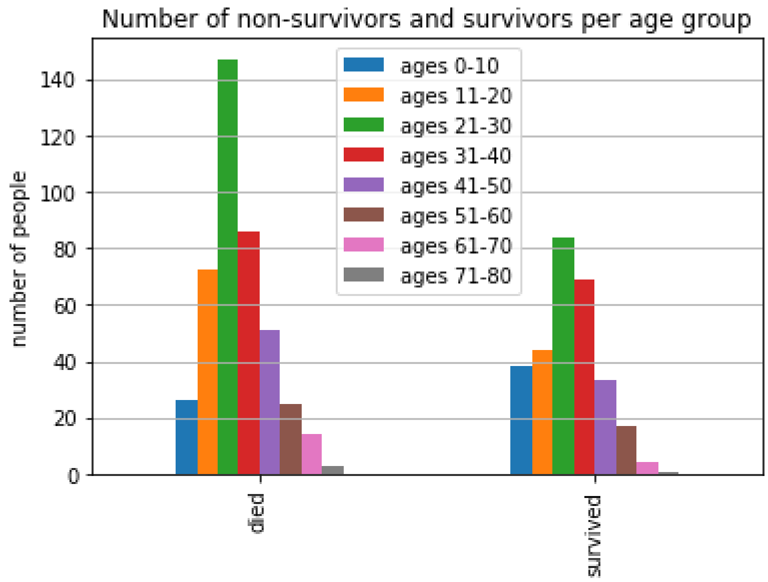
6. Clean the data by setting the PassengerID as row index, and then make a bar figure that compares the survival rate of different age range(e.g. 0-10 years old, 11-20, 21-30...).

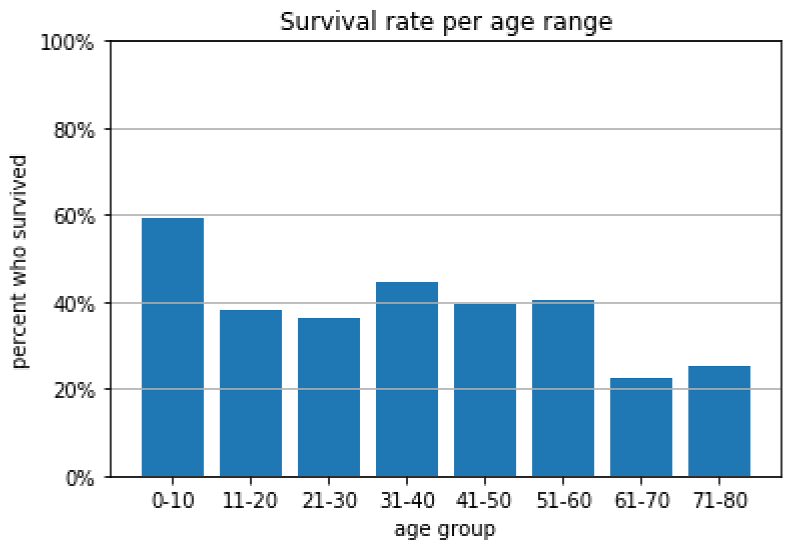
In [265]:

```
#titanic.reset_index(drop=True)
titanic.set_index(keys='PassengerId', inplace = True)

#oldest is 80
zero = titanic.loc[titanic['Age'] >= 0]
zero = zero[zero['Age'] < 11]
eleven = titanic.loc[titanic['Age'] >= 11]
eleven = eleven[eleven['Age'] < 21]
twentyone = titanic.loc[titanic['Age'] >= 21]
twentyone = twentyone[twentyone['Age'] < 31]
thirtyone = titanic.loc[titanic['Age'] >= 31]
thirtyone = thirtyone[thirtyone['Age'] < 41]
fortyone = titanic.loc[titanic['Age'] >= 41]
fortyone = fortyone[fortyone['Age'] < 51]
fiftyone = titanic.loc[titanic['Age'] >= 51]
fiftyone = fiftyone[fiftyone['Age'] < 61]
sixtyone = titanic.loc[titanic['Age'] >= 61]
sixtyone = sixtyone[sixtyone['Age'] < 71]
seventyone = titanic.loc[titanic['Age'] >= 71]
seventyone = seventyone[seventyone['Age'] < 81]

zeroY = 0.0
zeroN = 0.0
for person in zero['Survived']:
    if person == 1:
        zeroY += 1
    if person == 0:
        zeroN += 1
zeroSR = zeroY / (zeroY + zeroN)

elevenY = 0.0
elevenN = 0.0
for person in eleven['Survived']:
    if person == 1:
        elevenY += 1
    if person == 0:
        elevenN += 1
elevenSR = elevenY / (elevenY + elevenN)

twentyoneY = 0.0
twentyoneN = 0.0
for person in twentyone['Survived']:
    if person == 1:
        twentyoneY += 1
    if person == 0:
        twentyoneN += 1
twentyoneSR = twentyoneY / (twentyoneY + twentyoneN)

thirtyoneY = 0.0
thirtyoneN = 0.0
for person in thirtyone['Survived']:
    if person == 1:
        thirtyoneY += 1
    if person == 0:
        thirtyoneN += 1
thirtyoneSR = thirtyoneY / (thirtyoneY + thirtyoneN)
```

```
fortyoneY = 0.0
fortyoneN = 0.0
for person in fortyone['Survived']:
    if person == 1:
        fortyoneY += 1
    if person == 0:
        fortyoneN += 1
fortyoneSR = fortyoneY / (fortyoneY + fortyoneN)

fiftyoneY = 0.0
fiftyoneN = 0.0
for person in fiftyone['Survived']:
    if person == 1:
        fiftyoneY += 1
    if person == 0:
        fiftyoneN += 1
fiftyoneSR = fiftyoneY / (fiftyoneY + fiftyoneN)

sixtyoneY = 0.0
sixtyoneN = 0.0
for person in sixtyone['Survived']:
    if person == 1:
        sixtyoneY += 1
    if person == 0:
        sixtyoneN += 1
sixtyoneSR = sixtyoneY / (sixtyoneY + sixtyoneN)

seventyoneY = 0.0
seventyoneN = 0.0
for person in seventyone['Survived']:
    if person == 1:
        seventyoneY += 1
    if person == 0:
        seventyoneN += 1
seventyoneSR = seventyoneY / (seventyoneY + seventyoneN)


data = {'ages 0-10': {'survived':zeroY, 'died':zeroN},
        'ages 11-20': {'survived':elevenY, 'died':elevenN},
       'ages 21-30': {'survived':twentyoneY, 'died':twentyoneN},
        'ages 31-40': {'survived':thirtyoneY, 'died':thirtyoneN},
       'ages 41-50': {'survived':fortyoneY, 'died':fortyoneN},
        'ages 51-60': {'survived':fiftyoneY, 'died':fiftyoneN},
       'ages 61-70': {'survived':sixtyoneY, 'died':sixtyoneN},
        'ages 71-80': {'survived':seventyoneY, 'died':seventyoneN}}
df = pd.DataFrame(data)
df.plot(kind='bar')
plt.gca().set_title('Number of non-survivors and survivors per age group')
plt.ylabel('number of people')
plt.axes()
plt.gca().yaxis.grid()
plt.show()

percentages = [zeroSR, elevenSR, twentyoneSR, thirtyoneSR,
              fortyoneSR, fiftyoneSR, sixtyoneSR, seventyoneSR]
N = len(percentages)
```

```python
x = range(N)
width = 1/1.5
plt.bar(x, percentages)
#fig = plt.gcf()
plt.gca().set_title('Survival rate per age range')
plt.xlabel('age group')
plt.ylabel('percent who survived')
plt.gca().set_xticklabels([None, '0-10', '11-20', '21-30', '31-40',
                           '41-50', '51-60', '61-70', '71-80'])
plt.gca().set_yticklabels(['0%', '20%', '40%', '60%', '80%', '100%'])
plt.axes()
plt.gca().set_ylim([0, 1])
plt.gca().yaxis.grid()
plt.show()

titanic.head()
```


Number of non-survivors and survivors per age group
number of people
0
20
40
60
80
100
120
140
ages 0-10
ages 11-20
ages 21-30
ages 31-40
ages 41-50
ages 51-60
ages 61-70
ages 71-80
died
survived


Survival rate per age range
percent who survived
0%
20%
40%
60%
80%
100%
0-10
11-20
21-30
31-40
41-50
51-60
61-70
71-80
age group

Out[265]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
PassengerId										
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	Nal
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C8
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	Nal
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C12
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	Nal