Solve the ODE below by newton-linearization scheme

$$y'' - (y')^2 - y^2 + y + 1 = 0$$

In [10]:

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

In [11]:

```
def get_a(x, h):
    return 1/(h**2) - 1/(4.0 * h**2)
def get_b(x, h):
    return - 2/(h**2) + B(x)
def get_c(x, h):
    return 1/(h**2) + A(x)/(2.0 * h)
```

In [31]:

```
epsilon = 0.00001
def ThomasAlgorithm(a, b, c, d, n):
    c_dash = np.zeros(n-1)
    d_dash = np.zeros(n-1)
    c_dash[0] = c[0]/b[0]
    d_dash[0] = d[0]/b[0]
    for itr in range(1, n-1):
        c_dash[itr] = c[itr] / (b[itr] - a[itr] * c_dash[itr-1])
        d_dash[itr] = (d[itr] - a[itr]*d_dash[itr-1]) / (b[itr] - a[itr] * c_dash[it

    y = np.zeros(n-1)
    y[n-2] = d_dash[n-2]

    for itr in reversed(range(n-2)):
        y[itr] = d_dash[itr] - c_dash[itr] * y[itr+1]

    return y
```

In [99]:

```
x0 = 0
xn = np.pi
y0 = 0.5
yn = -0.5
def func(x0, xn, h = 0.1):
    lst = np.arange(x0, xn, h)
    lst = np.append(lst, xn)
    return lst

def BVP(x0, xn, y0, yn, step, epsilon = 0.0001):
    '''Keeping the initialization y = 0.5cos(x) '''
    x = func(x0, xn, step)
    print(x)
    y = 0.5*(np.cos(func(x0, xn, step)))
#     y = np.zeros(x.shape[0])
# #     y[0] = 0.5
#     y[-1] = -0.5
    print(y.shape)
#     a = [1/step**2 - 2*(y[i+1] - y[i-1])/(4*step**2)for i in range(1, len(y)-1)]
#     b = [-2/step**2 + -2*y[i] + 1 for i in range(1, len(y)-1)]
#     c = [1/step**2 + 2*(y[i+1] - y[i-1]) for i in range(1, len(y) -1)]
#     d = [-(y[i]**2 - y[i] - 1 + (y[i+1] -y[i-1])**2/(4*step**2) - (y[i-1] - 2*y[i
    delta_y = np.ones(y.shape)
    while(np.amax(np.absolute(delta_y))>epsilon):
        a = [1/step**2 + 2*(y[i+1] - y[i-1])/(4*step**2)for i in range(1, len(y)-1)]
        b = [-2/step**2 - 2*y[i] + 1 for i in range(1, len(y)-1)]
        c = [1/step**2 - 2*(y[i+1] - y[i-1])/(4*step**2) for i in range(1, len(y) -1
        d = [y[i]**2 - y[i] - 1 + (y[i+1] -y[i-1])**2/(4*step**2) - (y[i-1] - 2*y[i
        delta_y = ThomasAlgorithm(a, b, c, d, len(y)-1)
        delta_y = np.insert(delta_y, 0, 0)
        delta_y = np.append(delta_y, 0)
        y = y + delta_y
    print(y)
    return y

y_new = BVP(x0, xn, y0, yn, step=0.2, epsilon = 0.001)
print(y_new)
```

```
[0.         0.2        0.4        0.6        0.8        1.
 1.2        1.4        1.6        1.8        2.         2.2
 2.4        2.6        2.8        3.         3.14159265]
(17,)
[  0.          -1.97769364  -3.90678151  -5.70926742  -7.31102328
  -8.64500281  -9.65428216 -10.29472426 -10.53709057 -10.36847394
  -9.79298213  -8.8316532   -7.52162476  -5.91461128  -4.0747687
  -2.07604584   0.        ]
[0.         0.98550929 1.94626708 2.84428489 3.64233336 4.30709233
 4.81015551 5.12951713 5.25058545 5.16690595 4.88050279 4.40183814
 3.74937548 2.94900371 2.03215719 1.03730161 0.        ]
[0.         0.48457368 0.9580719  1.40047805 1.79401495 2.12204331
 2.37057736 2.52870985 2.58919675 2.54880254 2.40847284 2.17331608
 1.85243671 1.45829007 1.00736596 0.51498825 0.        ]
[0.         0.2759549  0.51240895 0.72195077 0.90284425 1.04976984
 1.15713104 1.22043637 1.23681564 1.2052787  1.12682172 1.00443381
 0.84302631 0.6494889  0.43240949 0.20531465 0.        ]
[0.         0.04602887 0.12639225 0.21262965 0.29394717 0.36514909
 0.42283811 0.4644477  0.48809863 0.49263108 0.4776435  0.44348588
 0.39117464 0.32212639 0.23752805 0.13576352 0.        ]
```

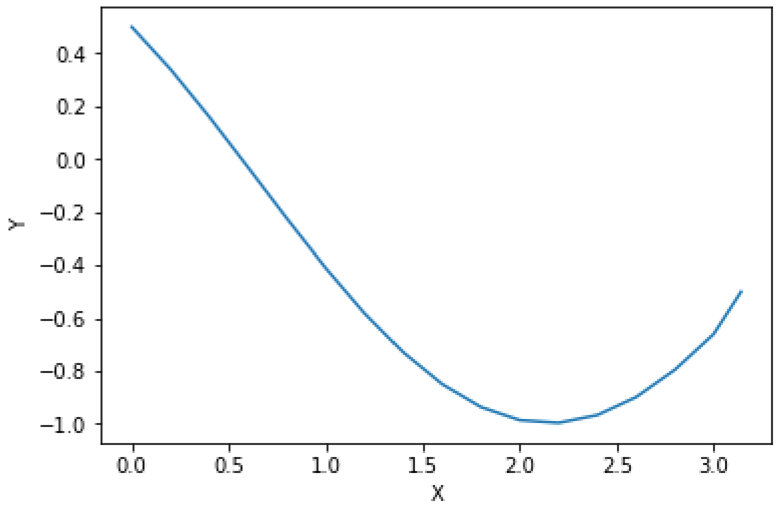
```
[0.         0.03450184 0.06037367 0.08188633 0.09978059 0.11373877
 0.12330146 0.12810868 0.12796293 0.1228533  0.11296758 0.09869788
 0.08064749 0.05966515 0.03699036 0.01488642 0.        ]
[0.         0.00097501 0.00247259 0.00403656 0.00549854 0.00676979
 0.00778867 0.00850912 0.00889858 0.00893804 0.00862249 0.00796086
 0.00697481 0.0056946  0.00414711 0.00231626 0.        ]
[0.00000000e+00 1.15136454e-05 2.08918753e-05 2.90367836e-05
 3.59896238e-05 4.15611413e-05 4.55445760e-05 4.77773075e-05
 4.81611910e-05 4.66708662e-05 4.33575025e-05 3.83498997e-05
 3.18545565e-05 2.41605469e-05 1.56721355e-05 7.06376509e-06
 0.00000000e+00]
[ 0.5         0.33989476  0.15975631 -0.03130433 -0.22421508 -0.410246
95
 -0.58126558 -0.72996405 -0.85008419 -0.9366187  -0.98598128 -0.996132
76
 -0.96665433 -0.89876267 -0.79526604 -0.66046432 -0.5       ]
```

In [100]:

```
x = func(x0, xn, 0.2)
plt.xlabel('X')
plt.ylabel('Y')
plt.plot(x, y_new, '-')
```

Out[100]:

```
[<matplotlib.lines.Line2D at 0x114cc8390>]
```



In [ ]: