Problem 1

Get polygon's area

In [1]:

```
import numpy as np

def PolyArea(x,y):
    xy_ = np.dot(x, np.roll(y,1)) - np.dot(y,np.roll(x,1))
    area = 0.5 * np.abs(xy_)
    return area
```

In [3]:

```
x = np.arange(0, 1+0.01, 0.01)
y = np.sqrt(1 - x**2)

print(PolyArea(x,y))
```

```
0.28510425794475935
```

Problem 2

get points of f(x) = x, g(x) = x^2

In [5]:

```
import numpy as np

def f(x):
    return x

def g(x):
    return x**2
```

In [22]:

```
N = int(input("N: "))
E = float(input("E: "))

interval = 8 / N # -4 ~ 4를 N 구간으로 divide
x = np.arange(-4, 4+interval, interval) # x의 범위: -4~4
```

```
N: 400
E: 0.01
```

In [45]:

```
point = [] # E 범위에 만족하는 point가 저장될 list

def CalcPoint(x):
    for _ in x:
        val = f(_) - g(_)
        if(np.abs(val) < E):
            point.append(_)
    return
```

In [46]:

```
CalcPoint(x)
print("오차에 만족하는 근삿값:",point)
```

오차에 만족하는 근삿값: [3.552713678800501e-15, 1.0000000000000044]

Problem 3

PI Scheme - Leibniz vs Euler

In [4]:

```
def LeibnizPI(n):
    sum = 0.0
    for k in range(n):
        sum += 1 / ((4*k + 1) * (4*k + 3))
    return 8 * sum

def EulerPI(n):
    sum = 0.0
    for k in range(1,n):
        sum += 1 / (k ** 2)
    return (6 * sum) ** 0.5
```

In [5]:

```
N = int(input("iterations(N): "))
print("Leibniz:",LeibnizPI(N))
print("Euler:  ",EulerPI(N))
```

```
iterations(N): 100
Leibniz: 3.1365926848388144
Euler:   3.1319807472443624
```

In [8]:

```
import pylab as plt
import math

LeibnizResult = []; EulerResult = []
t = range(1, N+1)

for k in t:
    LeibnizResult.append(LeibnizPI(k))
    EulerResult.append(EulerPI(k))

plt.plot(t, LeibnizResult, 'r')
plt.plot(t, EulerResult, 'g')
plt.axhline(y=math.pi)
plt.xlabel('iterations(N)'); plt.ylabel('acc - pi')
plt.ylim(3.05, 3.15)
plt.legend(['Leibniz','Euler'])
plt.show()
```