Aproximación de Ecuaciones Diferenciales

Luis Eduardo Robles Jiménez

0224969

Input

In [25]:
```
# yp, a, b, n, c = 'y - t**2 + 1', 0, 2, 10, 0.5
#yp, a, b, n, c = '-2*t**3 + 12*t**2 - 20*t + 8.5', 0, 4, 8, 1
yp, a, b, n, c = 'y - t**2 + 1', 0, 2, 10, 0.5
```

Method

In [14]:
```
Euler(yp, a, b, n, c)
```

Out[14]:
```
[5.25000000000000,
 5.87500000000000,
 5.12500000000000,
 4.50000000000000,
 4.75000000000000,
 5.87500000000000,
 7.12500000000000,
 7.00000000000000]
```

In [34]:
```
RunggeKutta(yp, a, b, n, c)
```

Out[34]:
```
[0.829293333333333,
 1.21407621066667,
 1.64892201704160,
 2.12720268494794,
 2.64082269272875,
 3.17989417023223,
 3.73234007285498,
 4.28340949831841,
 4.81508569457943,
 5.30536300069265]
```

Euler

In [15]:
```
def Euler(fun, a, b, n, c):
    f = parse_expr(fun)
    h = (b - a)/n
    tT, yV, p = a, c, []
    for i in range(1, n+1):
        yV += h*N(f.subs([(t, tT), (y, yV)]))
        tT += h
        p.append(yV)
    return p
```

Rungge Kutta (Cuarto Grado)

In [33]:
```
def RunggeKutta(fun, a, b, n, c):
    f = parse_expr(fun)
    h = (b - a)/n
    tT, yV, p = a, c, []
    for i in range(n):
        ku = h*N(f.subs([(t, tT), (y, yV)]))
        kd = h*N(f.subs([(t, tT + h/2), (y, yV + ku/2)]))
        kt = h*N(f.subs([(t, tT + h/2), (y, yV + kd/2)]))
        kc = h*N(f.subs([(t, tT + h), (y, yV + kt)]))
        yV += (ku + 2*kd + 2*kt + kc)/6
        tT += h
        p.append(yV)
    return p
```

Run first

In [5]:
```
from sympy import *
t, y = symbols("t"), symbols("y")
```