Aproximación de Ecuaciones Diferenciales

Luis Eduardo Robles Jiménez

0224969

Input

In [ ]:
```
#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
yp, a, b, n, c = '-5*y + 5*t**2 + 2*t', 0, 1, 10, 1/3
```

Method

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

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

Euler

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

Rungge Kutta (Cuarto Grado)

```
In [ ]: def RunggeKutta(fun, a, b, n, c):
            f = parse_expr(fun)
            print("\tf(x) =", f, end = "\n\n")
            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)
                print(tT, yV, sep = "\t")
            return (tT, yV)
```

Run first

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