Algoritmos y Complejidad
Python siendo Python

Pythonic Complexers

SymPy

La computación simbolica pelea con el calculo de objetos matematicos (valga la redundancia) simbolicamente. Las estructuras y objetos matematicos son representados **exactamente** y no aproximadamente (como suele pasar en analisis númerico).

```
>> import math
>> math.sqrt(8)
2.82842712475

>> import sympy
>> sympy.sqrt(8)
2*sympy.sqrt(2)
```

Listing 1: Intente

```
x, y = symbols('x y')
expr = x + 2*y
print(expr +1)
print(expr -x)
print(x*expr)
```

Listing 2: Cambiele el nombre a las variables

```
x, y, z = symbols('y z x')
3*x**2-2*x+5*x*y*z-2*z-3*x**2 *z**2 +4*y**2 +8
```

Listing 3: Obten el LaTeX

```
x, y, z = symbols('y z x')
latex(3x**2 -2*x +5*x*y*z-2z-3*x**2 *z**2 + 4*y**2 +8)
```

Listing 4: O inicia una sesion

```
sympy.init_session() # sympy.init_printing()
x, y, z = symbols('y z x')
3x**2 -2*x +5*x*y*z-2z-3*x**2 *z**2 + 4*y**2 +8
```

NumPy

NumPy es una extensión de Python, que le agrega mayor soporte para vectores y matrices, constituyendo una biblioteca de funciones matemáticas de alto nivel para operar con esos vectores o matrices.

Implemente el producto Matrix-Vector(columna)

```
import numpy as np
def matrixCrossVector(m, v):
    if m.shape[1] == v.shape[0]:
        result = np.zeros(m.shape[1])
        for r in range(m.shape[1]):
            result += [ m[r][c]*v[c][0] for c in range(v.shape[0])]
        return result
    else:
        raise AssertionError("Shapes not aligned")
```

SciPy

SciPy nos da herramientas y algoritmos matematicos, por ejemplo optimizacion, algebra lineal, integracion, interpolacion, Fourier, etc.

Matplotlib

Listing 5: Cute functions doing cute plots

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
x = np.arange(0, 2*np.pi, 0.01)

ax.plot(x, np.sin(x), label="$\sin(x)$")
ax.plot(x, np.cos(x), label="$\cos(x)$")
ax.plot(x, np.exp(np.sin(x)), label="$\exp(\sin(x))$")
ax.plot(x, np.exp(-(np.tan(x)**2)), label="$\exp(-tan^2(x))$")

ax.legend()
plt.grid(True)
plt.xlabel('x')
plt.ylabel('y')
plt.title("Las Funciones")
plt.show()
```

Ejercicios

1. Simplifique/Expanda/Factorize/Colecte/Cancele/Déje bonitas las siguientes expresiones
   - Simplifique

     $$\frac{\Gamma(x)}{\Gamma(x-2)}$$

   - Expanda

     $$(x+1)^2$$

   - Factorize

     $$\cos^2(x) + 2 \cdot \cos(x) \cdot \sin(x) + \sin^2(x)$$

- Simplifique

$$\frac{\sqrt{6}\cdot x^{-1}+3\cdot\sqrt[3]{x^4}\cdot\sqrt[3]{x^{-1}}+2}{x^{\frac{1}{2}}\cdot x^{\frac{1}{3}}\cdot x^{\frac{1}{6}}+1}$$

- Factorize

$$3x^7y+96x^2y^{11}$$

- Consiga los factores de

$$zx^2+4xyz+4zy^2$$

- Factorize

$$\frac{(x^4y^5)^3}{x^2y}-\frac{11(x^3y^4)^2}{xy}+28$$

- Junte los coeficientes de

$$xy+x-3+2x^2-zx^2+x^3$$

- Simplifique

$$\sin^4(x)-2\cos^2(x)\sin^2(2)+\cos^4(x)$$

- Simplifique

$$\cosh^2(x)+\sinh^2 x$$

2. Obtenga los siguientes limites:

- $$\lim_{x\to-2}\frac{x^4+5x^3+6x^2}{x^2(x+1)-4(x+1)}$$

- $$\lim_{x\to0}\frac{\sin(x)}{x}$$

3. Reescriba:

- La tangente en funcion del seno.
- El factorial en funcion de la Gamma.

4. Obtenga las siguientes derivadas:

- $$\frac{d}{dx}3e^x-4\cos(x)-\frac{\ln x}{4}$$

- $$\frac{\partial^7}{\partial x\partial y^2\partial z^4}e^{xyz}$$

- $$\frac{d}{dx}\sin\left(\sum_{i=1}^{20}x^i\right)\cos\left(\prod_{i=1}^{20}\log(x+i)\right)$$

5. Obtenga las siguientes integrales:

   - $$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-x^2-y^2}\,dx\,dy$$
   - $$\int \log^2 x\,dx$$
   - $$\int_{0}^{\pi} \sin(x^2)\,dx$$
   - $$\int_{-\infty}^{\infty} e^{-(x^2)}\,dx$$
   - $$\int_{0}^{1}\int_{0}^{1} e^{-(x+y)^2}\,dx\,dy$$

6. Descomponga en fracciones parciales:

   - $$\frac{4x^3+21x^2+10x+12}{x^4+5x^3+5x^2+4x}$$

7. Obtenga la Serie de Taylor:

   - Centrada en $x^\star = 0$ de:

     $$\sin(x)$$

   - Centrada en $x^\star = x^\star$ de:

     $$\frac{e^x}{\cos(x)}$$

8. Resuelva las siguientes sistemas de ecuaciones:

   - $$\begin{aligned} x_1 + 1 &= x_2 \\ x_1 + x_2 &= 10 \end{aligned}$$
   - $$x^3 - 4x^2 + 6x - 24 = 0$$
   - $$y'' + y = 0$$

- $$f'(t) = f^2(t) + 1$$

9. Encuentre usando el metodo de la biseccion los ceros de las siguientes funciones:
   - $x^3 + x - 1$
   - $sin(x)$
10. Calcule las propiedades descriptivas basicas del siguiente set $\{25, 35, 10, 17, 29, 14, 21, 31\}$
    - Suma
    - Promedio
    - Mediana
    - Desviacion estandar muestral
    - Grafico de Torta
    - Grafico de Caja-y-Bigote
11. Encuentre los optimos de las siguientes funciones:
    - Maximize $x(1-x)e^x$
    - Minimize $x^2 + 2y^2$
12. Interpole los siguientes set de puntos:
    - $(25, 35); (10, 17); (29, 14); (21, 31)$
13. Obtenga la inversa de las matrices:
    - Con $a, b, c \in R$
      $\begin{pmatrix} a & b \\ c & a \end{pmatrix}$
14. Implemente la interpolacion de Vandermonde e interpole:
    - $(25, 35); (10, 17); (29, 14); (21, 31)$

Extra

Sweet Python

Lambdas vs Function definitions

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

g = lambda x: x**2
```

List Comprehensions

```
[ [r*4+c for c in range(4)] for r in range(4)]
```

Generators vs Lists

```
def num_list(n):
    result = []
    for i in range(n):
        result.append(i)
    return result

print(num_list(3))

############################

def num_gen(n):
    for i in range(n):
        yield i

nums = num_gen(3)
print(next(nums), next(nums), next(nums))
```

Curry-ing

```
def a(a):
    def b(b):
        def c(c):
            def d(d):
                def e(e):
                    def f(f):
                        return f*e*d*c*b*a
                    return f
                return e
            return d
        return c
    return b

a(1)(2)(3)(4)(5)(6)
```

Function Closures

```
def counter():
    d = {'i': 0}
    def f():
        d['i'] += 1
        return d['i']
    return f

c = counter()
print(c(), c(), c(), c())
```

Functors

```
class F:
    def __init__(self, mult, incr):
        self.mult = mult
        self.incr = incr

    def __call__(self, x):
        return self.mult*x + self.incr

f = F(2, 4)

print(f(0), f(1), f(2), f(3))
```

Jupyter Notebooks

WIP.

Please Don't

Las siguientes son historias reales y aterradoras de Algoco, por favor no lo haga, da miedo.

Min3

Listing 6: http://shitcode.net/178

```
int min(int a,int b,int c) //Function to return the minimum.
{
  if(a < b)
  {
    if(a < c)return a;
    else if(a > c)return c;
    else return a;
  }
  else if(a > b)
  {
    if(b < c)return b;
    else if(b > c)return c;
    else return b;
  }
  else
```

```
16    {
17      if(a < c) return a;
18      else if(a > c) return c;
19      else return a;
20    }
21  }
```

Min3 revisited

Listing 7: http://shitcode.net/179

```
1  int minimum(int a, int b, int c){
2    int mini =a*b*c;
3    int iterator=0;
4    int test[3];
5    test[0]=a;
6    test[1]=b;
7    test[2]=c;
8    for (iterator=0;iterator<3;iterator++){
9      if (test[iterator]<mini){
10       mini=test[iterator];
11     }
12   }
13   return mini;
14 }
```

Absolute

Listing 8: http://shitcode.net/105

```
1  def absolute_value(value):
2      if str(value)[0]=='-':
3          value = -1 * value
4          return value
5      else:
6          return value
```