Definir Funções

Funções nativas e Funções Criadas

Algumas Funções nativas Basicas são as que seguem

In [ ]:
```
sin(pi/2)
```

In [ ]:
```
cos(pi/2)
```

In [ ]:
```
tan(pi/4)
```

In [ ]:
```
csc(pi/4)
```

In [ ]:
```
cosh(13)
```

In [ ]:
```
n(cosh(1))
```

In [ ]:
```
n((e+e^(-1))/2)
```

In [ ]:
```
ln(e^12)
```

In [ ]:
```
log(1024,2)
```

No Sage(como no Python) temos que declarar as variaveis que serao usadas antes de fazer calculos com elas. Veja

In [ ]:
```
f(x)=2*x
```

In [ ]:
```
f(2)
```

Funcao lambda

In [ ]:

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

In [ ]:

```
g(3)
```

In [ ]:

```
g('Marcelo')
```

In [ ]:

```
m=matrix([[1,0],[0,2]])
```

In [ ]:

```
show(m)
```

In [ ]:

```
g(m)
```

Funções DEF

Podemos também definir uma função usando o metodo def conforme segue

In [ ]:

```
def elevar_ao_quadrado(x):
    y=x^2
    return y
```

In [ ]:

```
elevar_ao_quadrado(13)
```

In [ ]:

```
elevar_ao_quadrado(m)
```

O def permite mais opcoes e pode ser usado para customizar varias funcoes, veja o exemplo a seguir

In [ ]:

```
def elevar_a_potencia(x,n=2):
    y=x^n
    return y
```

In [ ]:

```
elevar_a_potencia(5)
```

In [ ]:

```
elevar_a_potencia(5,3)
```

Calculo Diferencial e Integral

Limites

Para Calcular limites podemos usar o metodo $\mathbf{limit}$

In [ ]:

```
limit??
```

In [ ]:

```
f(x)=(x^5-32)/(x-2)
```

In [ ]:

```
limit(f,x=2)
```

Podemos já usar a propria formula dentro do limite como segue abaixo,(no caso em que as variaveis não são x lembre-se de definir-las)

In [ ]:

```
t=var('t')
limit(sin(t)/t,t=0)
```

Tambem é possivel a sintaxe a seguir

In [ ]:

```
f(t)=(1-cos(t))/t;
f.limit(t=0)
```

In [ ]:

```
h=var('h')
limit(((2+h)^2-2^2)/h, h=0)
```

Limites laterais e limites no Infinito tambem podem ser calculados seguindo o mesmo padrão.

In [ ]:

```
f=(1+1/x)^x
```

In [ ]:

```
limit(f,x=+infinity)
```

In [ ]:

```
limit(x^x,x=0, dir='+')
```

In [ ]:

```
limit(tan(x), x=pi/2, dir='-')
```

In [ ]:

```
limit(tan(x), x=pi/2, dir='+')
```

In [ ]:

```
limit((1 + sin(2*x))^(1/x),x=0)
```

In [ ]:

```
limit(cos(1/x^2),x=0)
```

In [ ]:

```
limit(1/x^2,x=0)
```

Calcule o seguintes limites:

1.$\lim_{x\to 1} \frac{x^3-x^2+x-1}{x-1}$

2.$\lim_{x\to 1^+} \frac{\sqrt{x}-1}{x-1}$

3.$\lim_{x\to 1} \arcsin(\frac{1-\sqrt{x}}{1-x})$

4.$\lim_{x\to \infty}(\frac{x^2-1}{x^2+1})$

5.$\lim_{x\to +\infty}(x-\sqrt{(x^2-x)})$

Derivadas

Podemos usar o Sage para calcular derivadas de varios modos, abaixo seguem algumas.

In [ ]:

```
f(x)=x^2
```

In [ ]:

```
f(x=2)
```

In [ ]:

```
diff(f,x)
```

In [ ]:

```
f.diff(x)
```

Derivadas de Ordem Superior podem ser calculadas simplesmente indicando a ordem. Vejamos:

In [ ]:

```
g(x)=x^10
```

In [ ]:

```
diff(g,x),diff(g,x,2),diff(g,x,3)
```

Ou usando a seguinte sintaxe.

In [ ]:

```
diff(g,x),diff(g,x,x),diff(g,x,x,x)
```

Derivadas implicitas também podem ser calculas com o metodo $\mathbf{implicit_derivative}$.

In [ ]:

```
x,y=var('x,y')
f=x^4-y^4
show(f.implicit_derivative(y,x))
```

Tambem é possivel calcular derivadas implicitas usando diretamente o metodo $\mathbf{diff}$. Vejamos o exemplo que segue:

In [ ]:

```
x=var('x')
y=function('y')(x)
show(diff(x^4-y^4==0,x))
```

In [ ]:

```
diff(x^4-y^4==0,x)
```

Derivadas parciais são calculadas indicando a variavel na qual está se tomando a derivada.

In [ ]:

```
x,y,t,z,w=var('x,y,t,z,w')
```

In [ ]:

```
F(x,y,z)=e^x+cos(x*y)+2*z^(17)
```

In [ ]:

```
diff(F,z)
```

Aqui uma opção seria calcular o gradiente.

In [ ]:

```
F.gradient()
```

Além disso, o Sage possui implementados os metodos **jacobian** e **hessian** para funções de varias variaveis.

In [ ]:

```
G=(x^2+y^2,x+cos(y),z+2*x+3*y)
```

In [ ]:

```
show(jacobian(G,(x,y,z)))
```

In [ ]:

```
show(F.hessian())
```

In [ ]:

```
show(F.hessian()(x,y,z))
```

Obtendo as regras de derivacao.

In [ ]:

```
reset()
```

In [ ]:

```
f=function('f')(x);
```

In [ ]:

```
g=function('g')(x);
```

In [ ]:

```
h=function('h')(x);
```

In [ ]:

```
show(diff(f*g,x))
```

In [ ]:

```
latex(diff(f*g,x))
```

In [ ]:

```

```

Exercício

Exercicio. Encontre formulas para as seguintes derivadas.

1.(f.g.h)'

2.(f/g)'

3.(f.g/h)

Criando uma implementação do teste da segunda derivada

O Metodo solve serve para resolver equações, vejamos

In [ ]:

```
reset()
```

In [ ]:

```
solve(x^2-4,x)
```

Como pegar somente os numeros deta lista?

In [ ]:

```
r,t=var('r t')
```

In [ ]:

```
eq=x^2==r+t
```

In [ ]:

```
eq.rhs()
```

In [ ]:

```
eq.lhs()
```

In [ ]:

```
lista=solve(x^2-4,x); show(lista)
```

In [ ]:

```
lista2=[k.rhs() for k in lista]; show(lista2)
```

Crivo de Erastostenes

In [ ]:

```
Lista3=[2,3,4,5,6,7,8,9,10,11,12,13];
```

In [ ]:

```
for k in Lista3:
    if is_prime(k):
        print '%d eh primo'%k
    else:
        print '%d NAO eh primo'%k
```

In [ ]:

```
for k in Lista3:
    if (is_prime(k) and k<9):
        print '%d eh primo menor que 9'%k
    elif (is_prime(k) and k>=9):
        print '%d eh primo maior ou igual a 9'%k
    else:
        print '%d NAO eh primo'%k
```

In [ ]:

```

```

Veja a implementação abaixo para o Teste da Segunda Derivada.

In [ ]:

```
def Teste_da_derivada_segunda_x(f):
    derivada=diff(f,x)
    pontos_criticos=solve(derivada==0,x)
    lista=[pontos_criticos[k].rhs() for k in range(len(pontos_criticos))]
    for k in lista:
        if diff(f,x,2)(x=k)>0:
            print '%d eh um ponto de minimo local'%(k)
        elif diff(f,x,2)(x=k)<0:
            print '%d eh um ponto de maximo local'%(k)

        else:
            print 'para %d o teste da derivada segunda foi inconclusivo'%(k)
```

In [ ]:

```

```

In [ ]:

```
f=x^4-1
```

In [ ]:

```
diff(f,f.variables()[0],2)
```

In [ ]:

```
Teste_da_derivada_segunda_x(f)
```

In [ ]:

```
f=1/3*x^3-5/2*x^2+6*x+3
```

In [ ]:

```
plot(f,(1.5,3.5))
```

In [ ]:

```
Teste_da_derivada_segunda_x(f)
```

Se a funcao eh funcao de outra variavel, vamos ver o que acontece:

In [ ]:

```
v=var('v')
```

In [ ]:

```
g=1/3*v^3-5/2*v^2+6*v+3
```

In [ ]:

```
Teste_da_derivada_segunda_x(g)
```

Como resolver este problema?

In [ ]:

```
j=g(v=x)
```

In [ ]:

```
Teste_da_derivada_segunda_x(j)
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
t=var('t')
```

In [ ]:

```
h=integral((t-1)*(t-2)*((t-3)^3)*(t-4),t)
```

Observe que no ponto 1 a derivada de h troca de positiva para negativa.

In [ ]:

```
plot(diff(h), t, (0.95, 4.05))
```

In [ ]:

```
h
```

In [ ]:

```
diff(h)
```

In [ ]:

```
Teste_da_derivada_segunda_x(h(t=x))
```

In [ ]:

```
f=1-x^2
```

In [ ]:

```
Teste_da_derivada_segunda_x(f)
```

In [ ]:

```
g=x^3
```

In [ ]:

```
Teste_da_derivada_segunda_x(g)
```

In [ ]:

```

```

In [ ]:

```

```

Exercicio Modifique a funcao anterior Programando o teste da Primeira Derivada.

Exercicio(Avancado) Programe o Teste da segunda Derivada Para uma funcoes de duas variaveis x e y.

INTEGRAIS

Integrais Simples.

Um exemplo de integral indefinida simples:

In [ ]:

```
integral(x^2,x)
```

Um exemplo de integral definida simples:

In [ ]:

```
integral(x^2,(x,0,1))
```

In [ ]:

```
integral((x^2)*cos(x),x)
```

In [ ]:

```
show(integral(e^(-x^2),x))
```

In [ ]:

```

```

Mesmo integrais impropria podemos calcular deste modo.

In [ ]:

```
show(integral(e^(x^2),(x,0,+infinity)))
```

In [ ]:

```
show(integral(e^(-x^2),(x,0,+infinity)))
```

In [ ]:

```
show(integral(e^(-x^2),(x,0,+oo)))
```

Para calcular integrais duplas(ou multiplas simplesmente iteramos as integrais).

In [ ]:

```
integral(integral(x*y,(y,0,10)),(x,0,10))
```

Podemos usar esta ideia, mesmo quando os limites de integracao são funções. Isto ajuda a calcular integrais entre curvas.

In [ ]:

```
#area da circunferencia de raio 2
integral(integral(1,(y,-sqrt(4-x^2),sqrt(4-x^2))),(x,-2,2))
```

In [ ]:

```

```

In [ ]:

```
```

Como calcular a area entre a reta $y = x$ e a parabola $y = x^2$

In [ ]:

```
(f1, f2) = x, x^2
plot([f1, f2], 0, 1, fill={1: [0]}, fillcolor='blue', fillalpha=.25)
```

In [ ]:

```
f=x; g=x^2
```

In [ ]:

```
limites_de_integracao=solve(f==g,x)
```

In [ ]:

```
show(limites_de_integracao)
```

In [ ]:

```
integral(1,(x,limites_de_integracao[0].rhs(),limites_de_integracao[1].rhs()))
```

Exercicios Calcule as seguintes Integrais:

$\int_0^1 \sqrt{(1-x^2)}dx$

$\int \frac{1}{\sqrt{(1-x^2)}}dx$

$\int_0^1 \sqrt{(1-x^2)}dx$

$\int e^{\sin(\theta)}\cos(\theta)d\theta$

Integral Numerica.

Integrais Trancendentes podem ser exploradas com o metodo $\mathbf{numerical\,integral}$

In [ ]:

```
show(integral(cos(x^2)))
```

In [ ]:

```
show(integral(cos(x^2)),x,0,pi)
```

In [ ]:

```
show(numerical_integral(cos(x^2),0,pi))##Saida= (valor da Integral, Precisao da Aproximacao)
```

In [ ]:

```
```

In [ ]:

```
```

Exercicio Usando o Metodos, **numericalintegral** e **integral** e calcule as seguintes integrais entre 1 e 10.

1. $\int e^{x^2} dx$
2. $\int \frac{e^x}{x} dx$
3. $\int e^{e^x} dx$ depois recalcule este item no intervalo $[0, 1]$
4. $\int \frac{\sin(x)}{x} dx$
4. $\int \frac{\ln(x)}{x^2+1} dx$

In [ ]:

```
show(integral(exp(x^2),x,1,10))
```

In [ ]:

```
show(n(integral(exp(x^2),x,1,10)))
```

In [ ]:

```
```

In [ ]:

```
numerical_integral(exp(x^2),1,10)
```

In [ ]:

```
show(integral(exp(x)/x,x,1,10))
```

In [ ]:

```
show(integral(exp(exp(x)),x,1,10))
```

In [ ]:

```
show(integral(sin(x)/x,x,1,10))
```

In [ ]:

```
show(integral(ln(x)/(x^2+1),x,1,10))
```

In [ ]:

```
```

In [ ]:

```
```

In [ ]:

```
```

Serie de Taylor e Maclaurin.

Para calcular a serie de Taylor de uma uma funcao podemos usar os metodos $\mathbf{taylor}$,$\mathbf{series}$

In [ ]:

```
show(taylor(exp(x), x, 0, 10))
```

In [ ]:

```
show(taylor(exp(x), x, 1, 10))
```

In [ ]:

```
show((exp(x)).series(x, 10))
```

Exercicio

1.Calcule a serie de Taylor de $cos(x)$.

2.Calcule a serie de Taylor de $sin(x)$.

3.Calcule a serie de Taylor de $exp(ix)$.

1. Verifique a formula de Euler $e^{ix} = cos(x) + i * sin(x)$.

In [ ]:

```
```

In [ ]:

```
```

In [ ]:

```
```

In [ ]:

```
```