# MATH 152 Lab 1

Put team members' names and section number here (double-click this line to edit).

```
import sympy as sp
from sympy.plotting import (plot,plot_parametric)
```

Instructions: Complete the lab assignment in your assigned groups. Unless stated otherwise, your answers should be obtained using Python code.

Do not modify the cell above, as it contains all the packages you will need. It is highly recommended to not use any additional packages.

## Question 1

### 1a

```
# Enter your code here
x = sp.Symbol('x', real=True)
f = x**2 * sp.exp(x)

domain = sp.calculus.util.continuous_domain(f, x, sp.S.Reals)
print(domain)
```

```
Reals
```

### 1b

```
# Enter your code here
x = sp.Symbol('x', real = True)
f = x**2 * sp.exp(x)

x_int = sp.solve(sp.Eq(f,0) , x)

y_int = f.subs(x,0)

print("x-interceps:" , x_int)
print("y-intercept:" , y_int)
```

```
x-interceps: [0]
y-intercept: 0
```

### 1c

```
# Enter your code here
x = sp.Symbol('x' , real=True)
f = x**2 * sp.exp(x)

fp = sp.diff(f,x)
print("f'(x) =" , fp)

crit = sp.solve(sp.Eq(fp,0) , x)
print("crit vals", crit)
```

```
f'(x) = x**2*exp(x) + 2*x*exp(x)
crit vals [-2, 0]
```

## 1d

```
# Enter your code here
x = sp.Symbol("x" , real = True)
f = x**2 * sp.exp(x)
fp = sp.diff(f,x)

crit = sp.solve(sp.Eq(fp, 0) , x)
crit_sorted = sorted(crit)

tests = [crit_sorted[0] - 1, (crit_sorted[0] + crit_sorted[1]) / 2, crit_sorted[1] + 1]

signs = []
for test in tests:
    value = sp.N(fp.subs(x, test))
    signs.append(value)
    print(f"f'({test}) = {value}")

print("Intervals:")
print("(-oo, -2):", "increasing" if signs[0] > 0 else "decreasing")
print("(-2, 0):", "increasing" if signs[1] > 0 else "decreasing")
print("(0, oo):", "increasing" if signs[2] > 0 else "decreasing")

print("Local extrema:")
print("At x = -2:", "local max" if signs[0] > 0 and signs[1] < 0 else "not a max")
print("At x = 0:", "local min" if signs[1] < 0 and signs[2] > 0 else "not a min")

print("Values at extrema:")
print("f(-2) =", sp.simplify(f.subs(x, -2)))
print("f(0)  =", sp.simplify(f.subs(x, 0)))
```

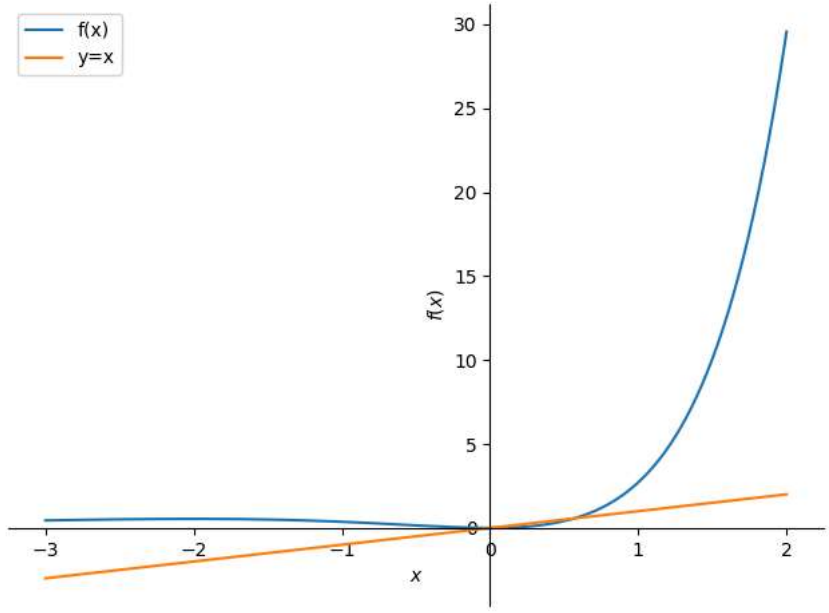
```
f'(-3) = 0.149361205103592
f'(-1) = -0.367879441171442
f'(1) = 8.15484548537714
Intervals:
(-oo, -2): increasing
(-2, 0): decreasing
(0, oo): increasing
Local extrema:
At x = -2: local max
At x = 0: local min
Values at extrema:
f(-2) = 4*exp(-2)
f(0)  = 0
```

## 1e

```
# Enter your code here
x = sp.Symbol('x', real=True)
f = x**2 * sp.exp(x)

p1 = plot(f, (x, -3,2), show=False, legend=True, label="f(x)")
p2 = plot(x, (x, -3, 2), show=False, legend=True, label='y=x')
p1.extend(p2)
p1.show()

eq = sp.Eq(f, x)
print("Fixed point:", 0)
root_pos = sp.nsolve(f - x, 0.6)
print("Fixed point (decimal):", root_pos)
```




```
Fixed point: 0
Fixed point (decimal): 0.567143290409784
```

## Question 2

### 2a

```
# Enter your code here
x, y = sp.symbols('x y', real=True)
f = 3*x - 1

inverse = sp.solve(sp.Eq(f, y), x)[0]
print("f^{-1}(y) =", inverse)

F = sp.integrate(f, x)
print("F(x) =", F)


laisant = sp.simplify(y * inverse - F.subs(x, inverse))
print("Using Laisant formula:", laisant)

direct = sp.integrate(inverse, y)
print("Direct integrate:", direct)

diff = sp.simplify(laisant - direct)
print("Difference (should be constant):", diff)
```

```
f^{-1}(y) = y/3 + 1/3
F(x) = 3*x**2/2 - x
Using Laisant formula: y**2/6 + y/3 + 1/6
Direct integrate: y**2/6 + y/3
Difference (should be constant): 1/6
```

### 2b

```
# Enter your code here
x, y = sp.symbols('x y', real=True)

f = sp.tan(x)
inv = sp.atan(y)

F = sp.integrate(f, x)
print("F(x) =", F)


laisant = sp.simplify(y * inv - F.subs(x, sp.atan(y)))
print("Using Laisant formula:", laisant)
```

```
direct = sp.integrate(sp.atan(y), y)
print("Direct integrate:", direct)

diff = sp.simplify(laisant - direct)
print("Difference (should be constant):", diff)
```

```
F(x) = -log(cos(x))
Using Laisant formula: y*atan(y) - log(y**2 + 1)/2
Direct integrate: y*atan(y) - log(y**2 + 1)/2
Difference (should be constant): 0
```

## 2c

```
# Enter your code here

x, y = sp.symbols('x y', real=True)

a = 0
b = sp.pi/4
c = sp.tan(a)
d = sp.tan(b)

right_hand = sp.simplify(b*d - a*c)
tan = sp.integrate(sp.tan(x), (x, a, b))

ans = sp.simplify(right_hand - tan)

print("bd - ac =", right_hand)
print("∫0 to pi/4 tan(x) dx =", tan)
print("Using identity, ∫0 to 1 arctan(y) dy =", ans)
```

```
bd - ac = pi/4
∫0 to pi/4 tan(x) dx = -log(sqrt(2)/2)
Using identity, ∫0 to 1 arctan(y) dy = -log(2)/2 + pi/4
```

# Question 3

## 3a

```
# Enter your code here

t, x = sp.symbols('t x', real=True)

A = 1
omega_c = sp.pi/2
m = (1/5)*sp.sin(sp.pi*x/2)
integrated = sp.integrate(m, (x, 0, t))
print("Integral term:", integrated)

wave1 = sp.sin(omega_c*t + integrated)
print("FM wave:", wave1)
```

```
Integral term: -0.4*cos(pi*t/2)/pi + 0.4/pi
FM wave: sin(pi*t/2 - 0.4*cos(pi*t/2)/pi + 0.4/pi)
```

## 3b

```
# Enter your code here
m = 3*sp.sin( sp.pi * x/2)
integrated = sp.integrate(m, (x, 0, t))
print("Integral term:", integrated)

wave2 = sp.sin(omega_c*t + integrated)
print("FM wave:", wave2)
```

```
Integral term: -6*cos(pi*t/2)/pi + 6/pi
FM wave: sin(pi*t/2 - 6*cos(pi*t/2)/pi + 6/pi)
```

## 3c

```
# Enter your code here
carrier = sp.sin(omega_c*t)
p2 = plot(wave1, (t, 0, 12), show=False, label='FM low amp')
p3 = plot(wave2, (t, 0, 12), show=False, label='FM high amp')

p1.extend(p2)
p1.extend(p3)
p1.show()
```