Name : Mudit sand

Roll NO. : 203100068

```
import numpy as np
import matplotlib.pyplot as plt
```

Change the input data in below cell to run further cells for different function

```
## Function 1
#change the input values here
##### Take one function values and comment others
# global h, y_init,a,b
# h = 0.1
# ## inital value
# y_init = -1*(1/np.log(2))
# ## Time endpoints
# a = 1
# b = 2
# ## write function below
# func1 = '(y**2)/(1+t)'
# sol_analytic = '-1/np.log(1+t)'

# ## Function 2
# #change the input values here
# global h, y_init,a,b
# h = 0.1
# ## inital value
# y_init = 1
# ## Time endpoints
# a = 0
# b = 1
# ## write function below
# func1 = '(2-2*y*t)/(t**2 +1)'
# sol_analytic = '(2*t + 1)/(1+t)'

# ## Function 3
# #change the input values here
# global h, y_init,a,b
# h = 0.2
# ## inital value
# y_init = -2
# ## Time endpoints
# a = 1
# b = 3
# ## write function below
# func1 = '(y**2 + y)/(t)'
# sol_analytic = '2*t/(1-2*t)'

## Function 4
#change the input values here
global h, y_init,a,b
h = 0.1
## inital value
y_init = 1
## Time endpoints
a = 0
b = 1
## write function below
func1 = '(-1*t*y) + (4*t/y)'
sol_analytic = 'np.sqrt(4-3*np.exp(-t**2))'
```

```
def func(function, y,t):
  return eval(function)
```

```
## Output plot data
y = [y_init]
t = np.arange(a,b,h)
y_imp = [y_init]
```

```
##analytic solution
y_analytic = []
```

```
for i in t:
  y_analytic.append(func(sol_analytic,0,i))
```

```
def eulexplicit(function, h=h,y_init = y_init):
  for i in t:
    y_val = y_init + h*func(function,y_init,i)
    y_init = y_val
    y.append(y_val)
  return y[:-1]
```

```
explicit_y =eulexplicit(func1)
```

```
def implicit(function,  h=h,y_init = y_init):
  for i in t:
    error = 10
    acc = .01
    ynew = y_init + error
    while error > acc:
      yiter = y_init + h*func(function,ynew,i)
      error = abs(ynew-yiter)
      ynew = yiter
    y_imp.append(ynew)
    y_init = ynew
  return y_imp[:-1]
```

```
implicit_y =implicit(func1)
```

```
plt.plot(t,explicit_y, label='explicit answer')
plt.plot(t,implicit_y, label='implicit answer')
plt.plot(t,y_analytic, label = 'actual value')
plt.legend()
plt.xlabel('t')
plt.ylabel('y')
plt.show()
```