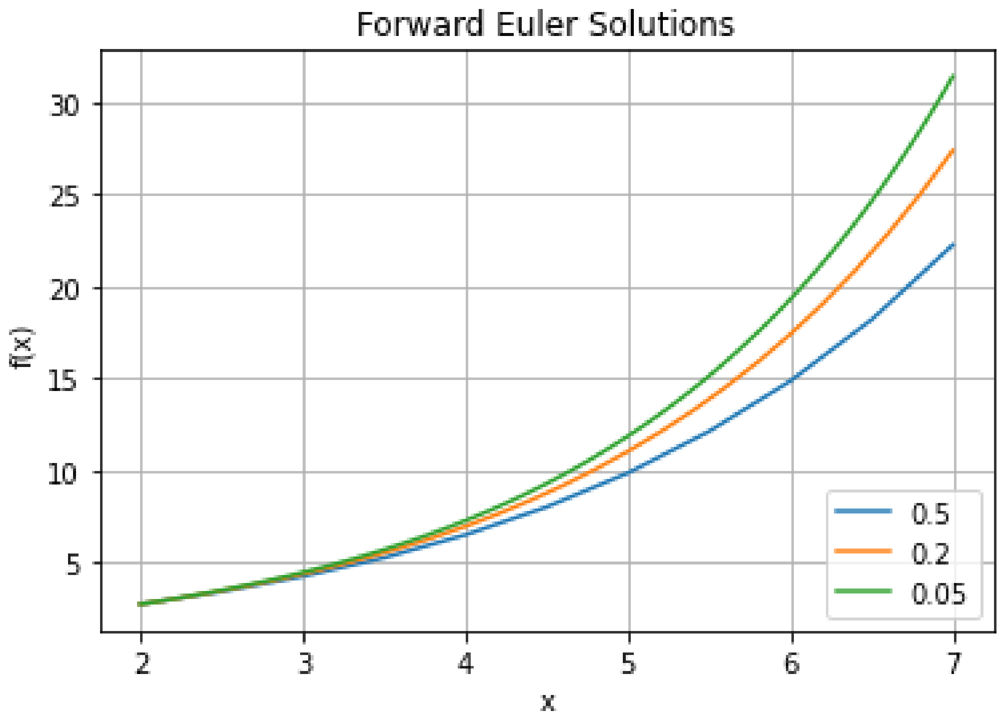
AS7 Notebook

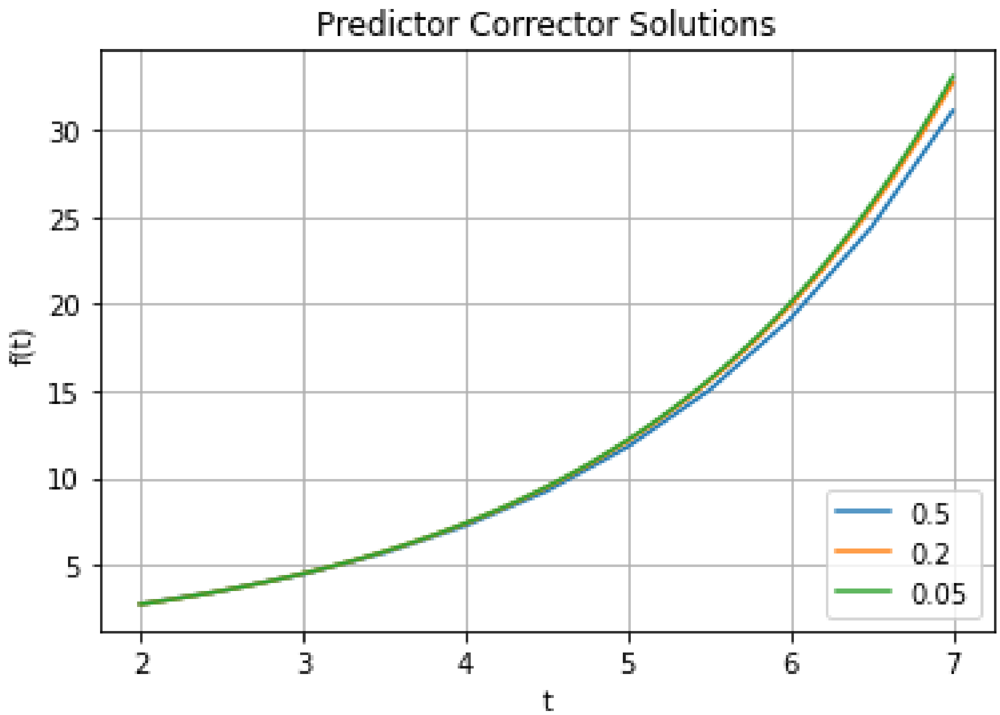
November 23, 2021

[12]:

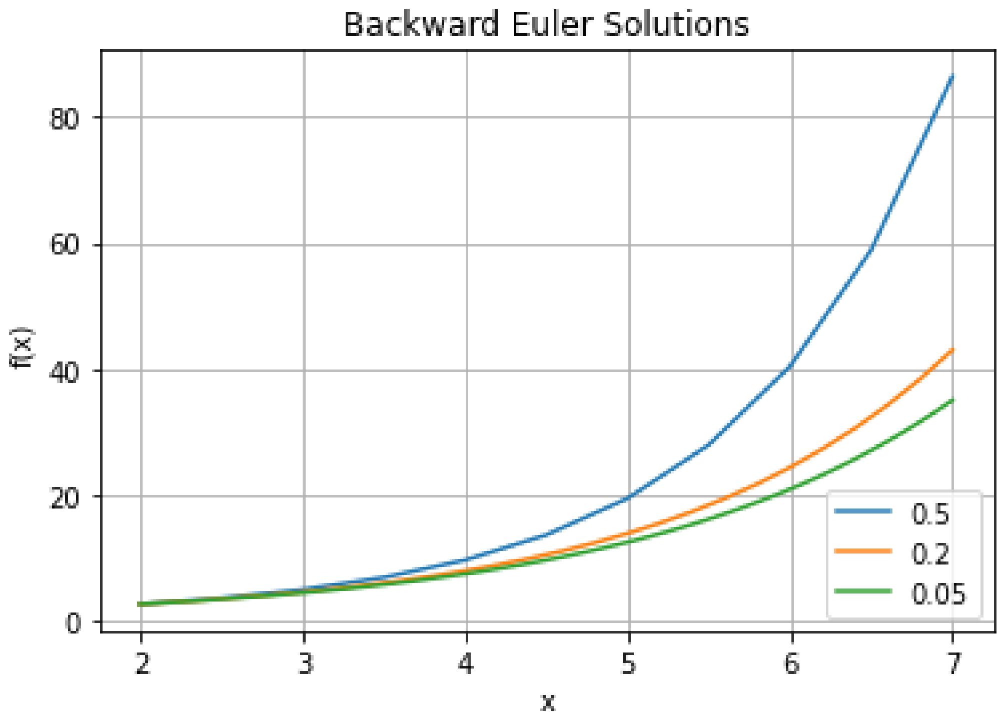
```
#QUESTION 1
import math
import matplotlib.pyplot as plt
from myfunctions import *

f = lambda x, y: (y*math.log(y))/x

#Forward Euler
t1, s1 = deqForwardEuler(f, 2, 7, 0.5, 2.71828)
t2, s2 = deqForwardEuler(f, 2, 7, 0.2, 2.71828)
t3, s3 = deqForwardEuler(f, 2, 7, 0.05, 2.71828)

plt.plot(t1, s1, label='0.5')
plt.plot(t2, s2, label = '0.2')
plt.plot(t3, s3, label = '0.05')
plt.title('Forward Euler Solutions')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.grid()
plt.legend(loc='lower right')
plt.show()
```



[7]:
```
#QUESTION 1
#Predictor Corrector
import math
import matplotlib.pyplot as plt
from myfunctions import *
f = lambda y, x: (y*math.log(y))/x

t1, s1 = deqPredictorCorrector(f, 2, 7, 0.5, 2.71828)
t2, s2 = deqPredictorCorrector(f, 2, 7, 0.2, 2.71828)
t3, s3 = deqPredictorCorrector(f, 2, 7, 0.05, 2.71828)

plt.plot(t1, s1, label='0.5')
plt.plot(t2, s2, label = '0.2')
plt.plot(t3, s3, label = '0.05')
plt.title('Predictor Corrector Solutions')
plt.xlabel('t')
plt.ylabel('f(t)')
plt.grid()
plt.legend(loc='lower right')
plt.show()
```

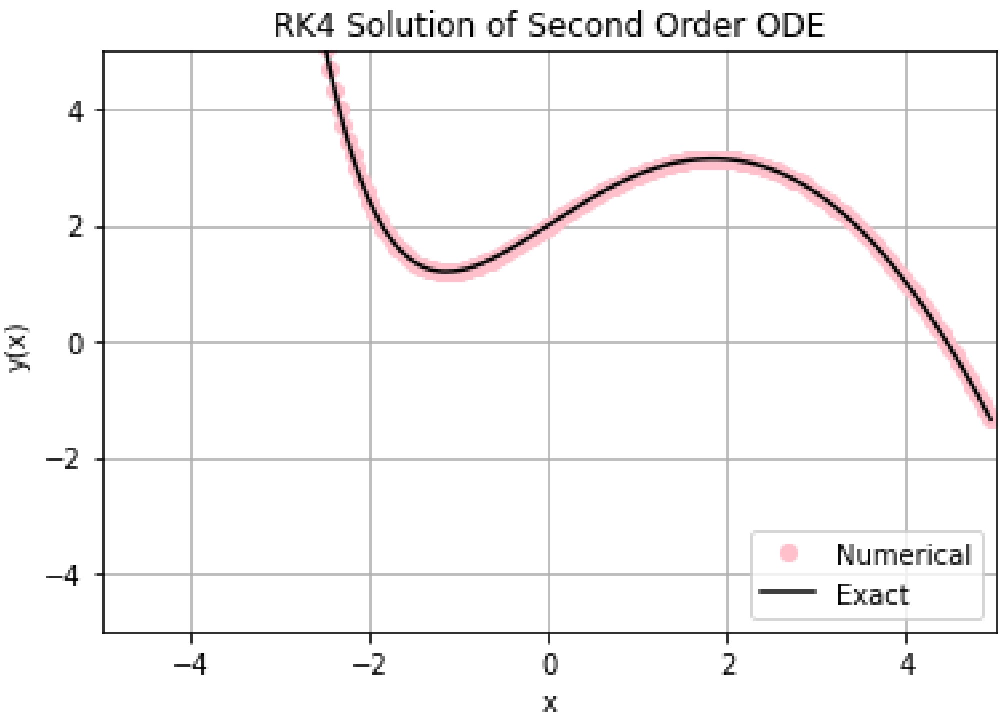


[11]:
```
#QUESTION 1
#Backward Euler
t1, s1 = deqBackwardEuler(2, 7, 0.5, 2.71828)
t2, s2 = deqBackwardEuler(2, 7, 0.2, 2.71828)
t3, s3 = deqBackwardEuler(2, 7, 0.05, 2.71828)

plt.plot(t1, s1, label='0.5')
plt.plot(t2, s2, label = '0.2')
plt.plot(t3, s3, label = '0.05')
plt.title('Backward Euler Solutions')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.grid()
plt.legend(loc='lower right')
plt.show()
```



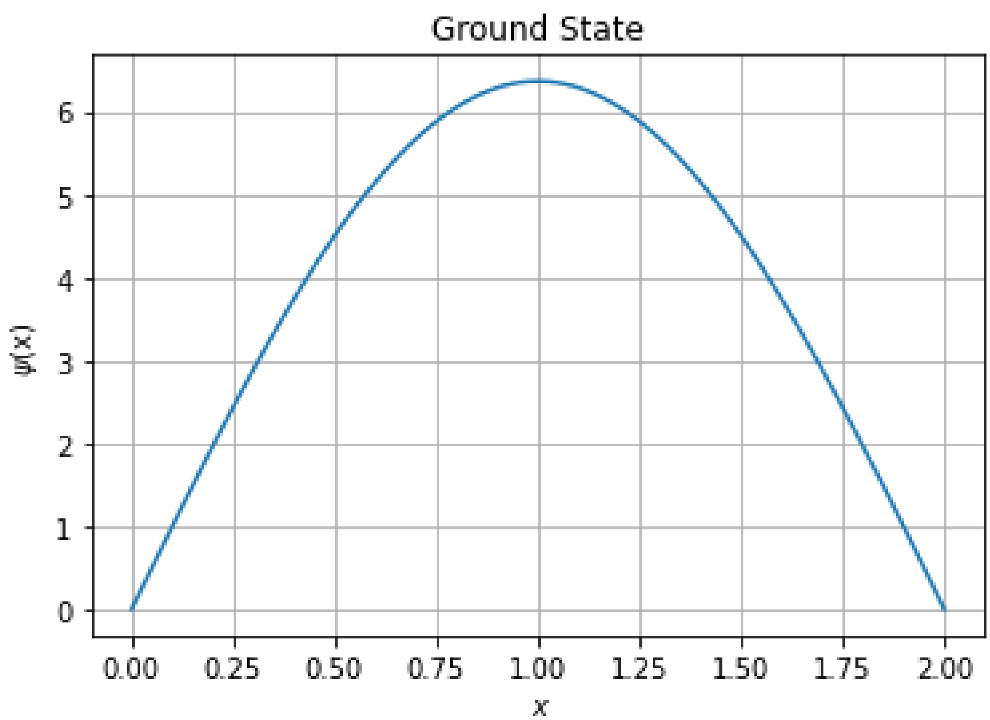
[24]:
```
#QUESTION 2
import math
import matplotlib.pyplot as plt
from myfunctions import *

def func1(y, u, x):
    return u
def func2(y, u, x):
    return (1-u-x)
x_list, y_list = deqRK4(func1, func2, 0, 5.0, 0.05, 2, 1)
x_back, y_back = deqRK4(func1, func2, 0, -5, -0.05, 2, 1)

x_min = -5.0
x_max = 5.0
step_size = 0.05
xExact = []
yExact = []

for i in range(math.ceil((x_max-x_min)/step_size)):
    xExact.append(x_min)
    x_min = x_min + step_size
for i in range(len(xExact)):
```

```
    yExact.append(2.0*xExact[i] - (1/2)*xExact[i]**2 + (math.exp(-xExact[i])) +␣
 ↪1.0)

plt.xlim(-5, 5)
plt.ylim(-5, 5)
plt.plot(x_list, y_list, 'o', color = 'pink', label = 'Numerical')
plt.plot(x_back, y_back, 'o', color = 'pink')
plt.plot(xExact, yExact, color = 'black', label = 'Exact')
plt.title('RK4 Solution of Second Order ODE')
plt.xlabel('x')
plt.ylabel('y(x)')
plt.grid()
plt.legend(loc='lower right')
plt.show()
```

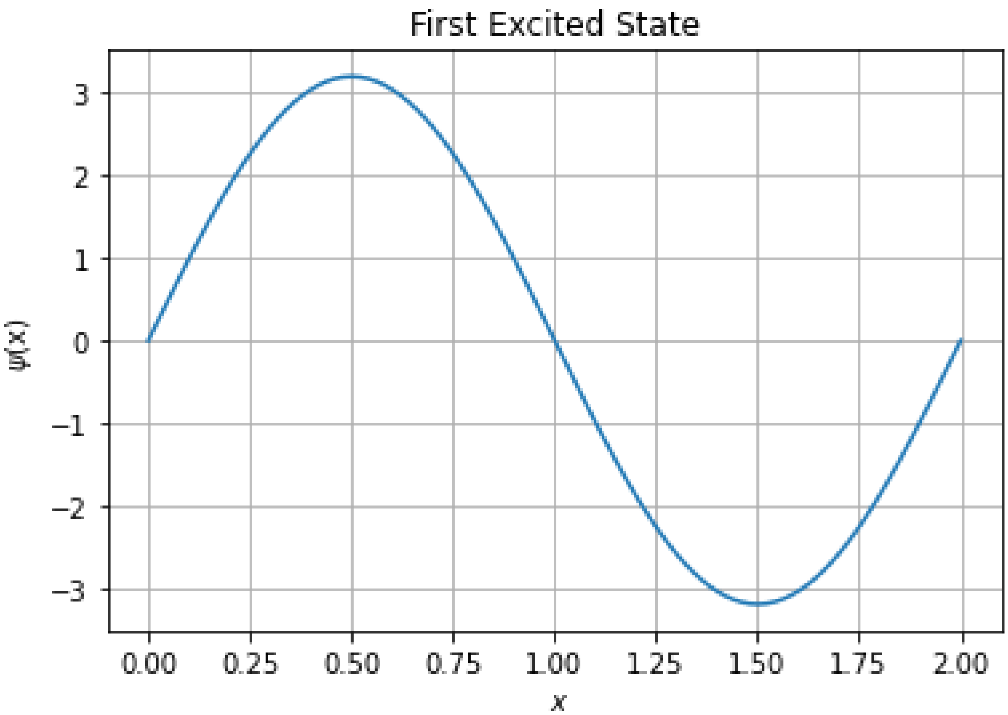


[22]:
```
#QUESTION 3
import math
import matplotlib.pyplot as plt
from myfunctions import *

# def shooting_method(d2ydx2, dydx, x0, y0, xf, yf, z_guess1, z_guess2,␣
 ↪step_size, tol=1e-6):
# this is func for d2y/dt2 =func
```

```python
def d2ydt2(t, y, z):
    return -((math.pi)**2)*y/4
# z = dy/dt
def dydt(t, y, z):
    return z

# Defining boundary values
x_initial = 0
x_final = 2
y_initial = 0
y_final = 0

x, y, z = deqShooting(d2ydt2, dydt, x_initial, y_initial, x_final, y_final, 10, 100, step_size=0.01)

plt.plot(x,y)
plt.xlabel(" $x$")
plt.ylabel("$\psi$(x)")
plt.title("Ground State")
plt.grid()
plt.show()
```



[23]:
```python
#QUESTION 3
import math
import matplotlib.pyplot as plt
from myfunctions import *
# def shooting_method(d2ydx2, dydx, x0, y0, xf, yf, z_guess1, z_guess2,␣
↪step_size, tol=1e-6):
# this is func for d2y/dt2 =func
def d2ydt2(t, y, z):
    return -((math.pi)**2)*y
# z = dy/dt
def dydt(t, y, z):
    return z

# Defining boundary values
x_initial = 0
x_final = 2
y_initial = 0
y_final = 0

x, y, z = deqShooting(d2ydt2, dydt, x_initial, y_initial, x_final, y_final, 10,␣
↪100, step_size=0.01)

plt.plot(x,y)
plt.xlabel(" $x$")
plt.ylabel("$\psi$(x)")
plt.title("First Excited State")
plt.grid()
plt.show()
```


First Excited State
ψ(x)
x
3
2
1
0
−1
−2
−3
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00