# Euler

Για την αριθμητική επίλυση ενός προβλήματος αρχικών τιμών (Π.Α.Τ.)

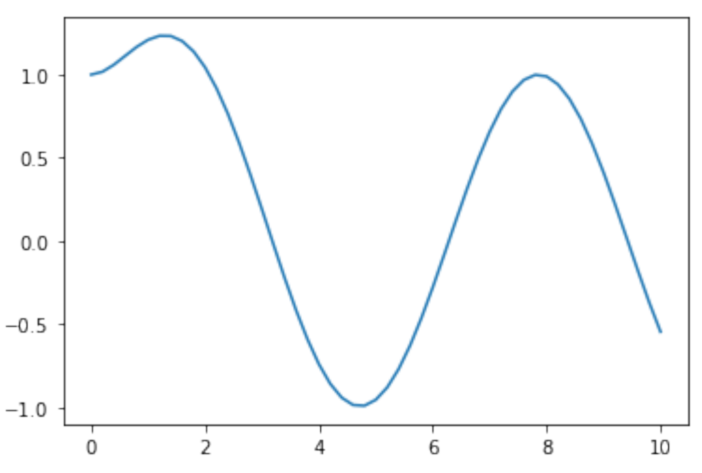
$$y'(t) = f(t, y(t)), \quad t \in [a,b], \quad y(0) = y_0$$

θεωρήσαμε τη μέθοδο του Euler. Έστω ένας ομοιόμορφος διαμερισμός του $[a,b]$, στα σημεία $t_n = a + nh$, $n = 0, \dots, N$, με βήμα $h = \frac{b-a}{N}$, υπολογίζουμε τις τιμές $y_n$ που αποτελούν προσεγγίσεις στις τιμές $y(t_n)$, $n = 0, \dots, N$, όπου

$$y_{n+1} = y_n + hf(t_n, y_n), \quad n = 0, \dots, N-1.$$

**Παράδειγμα 1:** Έστω $y(t) = e^{-t} + \sin(t)$, στο $[0, 10]$. Δημιουργείστε μια διαμέριση του $[0, 10]$ με 51 σημεία, $t_n$, $n = 0, 1, \dots, 50$, και χρησιμοποιήστε τη βιβλιοθήκη matplotlib για να σχηματίσετε το γράφημα της $y(t)$.

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt

def y(t):
    s=np.exp(-t)+np.sin(t)
    return s

N=50
t=np.linspace(0,10,N+1)

plt.plot(t,y(t))
plt.show()
```



**Παράδειγμα 2:** Έστω $y(t) = e^{-t} + \sin(t)$, στο $[0, 10]$, η οποία είναι λύση στο

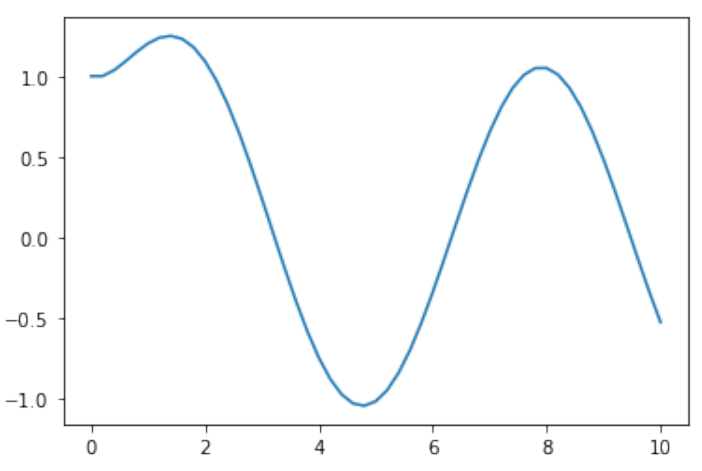
$$y'(t) = -y(t) + \cos(t) + \sin(t), \quad t \in [0, 10], \quad y(0) = 1.$$

1. Για βήμα $h = 0.5$ και αρχική τιμή $y_0 = 1$, υπολογίστε με τη μέθοδο του Euler την προσέγγιση $y_{10}$.
2. Για $N = 50$ κατασκευάστε τις προσεγγίσεις $y_n$, που δίνει η μέθοδος του Euler και δημιουργείστε τη γραφική παράσταση της προσεγγιστικής λύσης.

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt

def y_exact(t):
    s=np.exp(-t)+np.sin(t)
    return s

def f(t,y):
    s=-y+np.cos(t)+np.sin(t)
    return s

h=0.5 #Βήμα
N=10 #  10η προσέγγιση

# Δημιουργώ 10+1 θέσεις που θα αποθηκεύσω τις τιμές
y=np.zeros(N+1)

# Αρχική τιμή
y[0]=1

# Επαναλήψεις για να βρω την τιμή που ζητάω
for i in range(N):
    ti=i*h # Το σημείο ti
    y[i+1]=y[i]+h*f(ti,y[i])

print('y_10=',y[10])
print('t_10=',N*h)
print('Η ακριβής τιμή στο t_10:',y_exact(N*h))
```

```
y_10= -1.1410153656727977
t_10= 5.0
Η ακριβής τιμή στο t_10: -0.952186327664053
```

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt

def f(t,y):
    s=-y+np.cos(t)+np.sin(t)
    return s


N=50 #  Αριθμός προσεγγίσεων
h=10./N #Βήμα

# Δημιουργώ N+1 θέσεις που θα αποθηκεύσω τις τιμές
y=np.zeros(N+1)

# Αρχική τιμή
y[0]=1

# Επαναλήψεις για να βρω την τιμή που ζητάω
for i in range(N):
    ti=i*h # Το σημείο ti
    y[i+1]=y[i]+h*f(ti,y[i])

plt.plot(t,y)
plt.show()
```



**Παράδειγμα 3:** Για το παραπάνω πρόβλημα αρχικών τιμών, υπολογίστε το σφάλμα ανάμεσα στην ακριβή λύση και την προσεγγιστική στο σημείο $t = 10$, $|y_N - y(10)|$, όταν $N = 50, 100, 200, 400$. Αν τα σφάλματα είναι αντίστοιχα $err_1, err_2, err_3, err_4$. Διαπιστώστε αν ο λόγος $err_i/err_{i+1}$ είναι περίπου ο ίδιος για $i = 1, 2, 3$

In [4]:
```
import numpy as np
import matplotlib.pyplot as plt
def y_exact(t):
    s=np.exp(-t)+np.sin(t)
    return s

def f(t,y):
    s=-y+np.cos(t)+np.sin(t)
    return s

N=[50,100,200,400]
err=np.zeros(4)
for j in range(4):
    #Διαμερισμός
    t=np.linspace(0,10,N[j]+1)
    h=t[1]-t[0]

    #Θέσεις για να αποθηκεύσω τις προσεγγίσεις
    y=np.zeros(N[j]+1)
    y[0]=1

    #Μέθοδος Euler
    for i in range(N[j]):
      y[i+1]=y[i]+h*f(t[i],y[i])

    #Σφάλματα
    err[j]=abs(y[-1]-y_exact(10))


for j in range(3):
    print('Ο λόγος %d: %3.10f'%(j+1,err[j+1]/err[j]))
```

```
Ο λόγος 1: 0.5091306229
Ο λόγος 2: 0.5039291095
Ο λόγος 3: 0.5018269725
```

**Παράδειγμα 4:** Θεωρούμε τώρα ως

$$err_i = \max_{0 \le n \le N} |y_n - y(t_n)|, \quad N = 50, 100, 200, 400,$$

και διαπιστώστε αν ο λόγος $err_i/err_{i+1}$ είναι περίπου ο ίδιος για $i = 1, 2, 3$

In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
def y_exact(t):
    s=np.exp(-t)+np.sin(t)
    return s

def f(t,y):
    s=-y+np.cos(t)+np.sin(t)
    return s

N=[50,100,200,400]
err=np.zeros(4)
for j in range(4):
    #Διαμερισμός
    t=np.linspace(0,10,N[j]+1)
    h=t[1]-t[0]

    #Θέσεις για να αποθηκεύσω τις προσεγγίσεις
    y=np.zeros(N[j]+1)
    y[0]=1

    #Μέθοδος Euler
    for i in range(N[j]):
      y[i+1]=y[i]+h*f(t[i],y[i])

    #Σφάλματα
    err[j]=max(abs(y-y_exact(t)))


for j in range(3):
     print('Ο λόγος %d: %3.10f'%(j+1,err[j+1]/err[j]))
```

```
Ο λόγος 1: 0.4883780990
Ο λόγος 2: 0.4942511606
Ο λόγος 3: 0.4971822885
```

**Πειραματική εκτίμηση της τάξης σύγκλισης.** Γνωρίζουμε ότι για τη μέθοδο Euler το σφάλμα της μεθόδου ικανοποιεί

$$\max_{0 \le n \le N} |y_n - y(t_n)| \le Ch^p$$

με $p = 1$.

Υπολογίζοντας το σφάλμα

$$err_N = \max_{0 \le n \le N} |y_n - y(t_n)|$$

για δύο διαφορετικές διαμερίσεις με $N_1 < N_2$, η πειραματική τάξη σύγκλισης ορίζεται ως

$$p = \frac{\ln(\frac{err_{N_2}}{err_{N_1}})}{\ln(\frac{N_1}{N_2})}$$

**Παράδειγμα 5:** Θεωρείστε τις διαμερίσεις του $[0, 10]$, με $N = 50, 100, 200, 400$. Υπολογίστε τα σφάλματα $err_N$ και βρείτε τους λόγους που χρησιμοποιούμε για την πειραματική τάξη σύγκλισης της μεθόδου του Euler. Είναι $p \approx 1$;

In [6]:
```
import numpy as np
import matplotlib.pyplot as plt
def y_exact(t):
    s=np.exp(-t)+np.sin(t)
    return s

def f(t,y):
    s=-y+np.cos(t)+np.sin(t)
    return s

N=[50,100,200,400]
err=np.zeros(4)

for j in range(len(N)):
    t=np.linspace(0,10,N[j]+1)
    h=t[1]-t[0]
    y=np.zeros(N[j]+1)
    y[0]=1

    for i in range(N[j]):
      y[i+1]=y[i]+h*f(t[i],y[i])

    err[j]=max(abs(y_exact(t)-y))
    print('Μέγιστο σφάλμα για N=',N[j],':',err[j])

print('Πειραματική ταξη σύγκλισης')
for i in range(len(N)-1):
    p=np.log(err[i+1]/err[i])/np.log(N[i]/N[i+1])
    print(p)
```

```
Μέγιστο σφάλμα για N= 50 : 0.07624201608945702
Μέγιστο σφάλμα για N= 100 : 0.037234930883311645
Μέγιστο σφάλμα για N= 200 : 0.018403407803719407
Μέγιστο σφάλμα για N= 400 : 0.009149848407935313
Πειραματική ταξη σύγκλισης
1.0339295897749374
1.0166837412111878
1.0081531918145255
```

In [ ]: