Euler

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

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

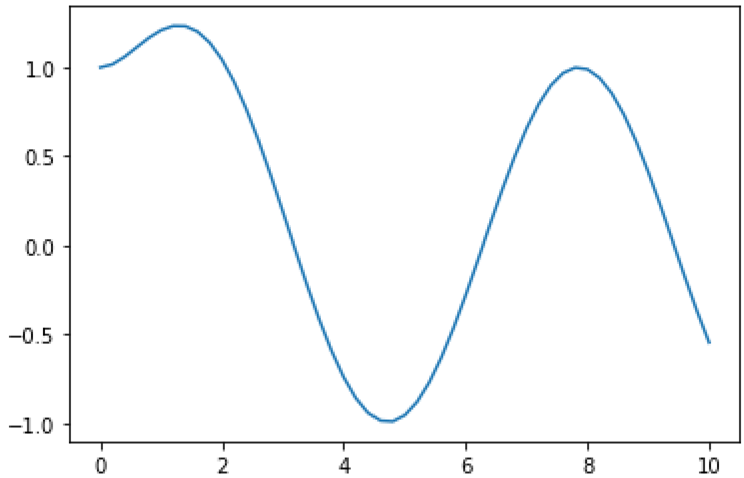
θεωρήσαμε τη μέθοδο του Euler. Έστω ένας ομοιόμορφος διαμερισμός του $[a, b]$, στα σημεία $t_n = a + nh$, $n = 0, \ldots, N$, με βήμα $h = \dfrac{b-a}{N}$, υπολογίζουμε τις τιμές y_n που αποτελούν προσεγγίσεις στις τιμές $y(t_n)$, $n = 0, \ldots, N$, όπου

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

Άσκηση 1: Έστω $y(t) = e^{-t} + \sin(t)$, στο $[0, 10]$. Δημιουργείστε μια διαμέριση του $[0, 10]$ με 51 σημεία, t_n, $n = 0, 1, \ldots, 50$, 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

t=np.linspace(0,10,51)
plt.plot(t,y(t))
plt.show()
```



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

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

Ορίστε στη Python τη συνάρτηση 2 μεταβλητών $f(t, y) = -y + cos(t) + sin(t)$ χρησιμοποιώντας τις συναρτήσεις sin και cos της Numpy

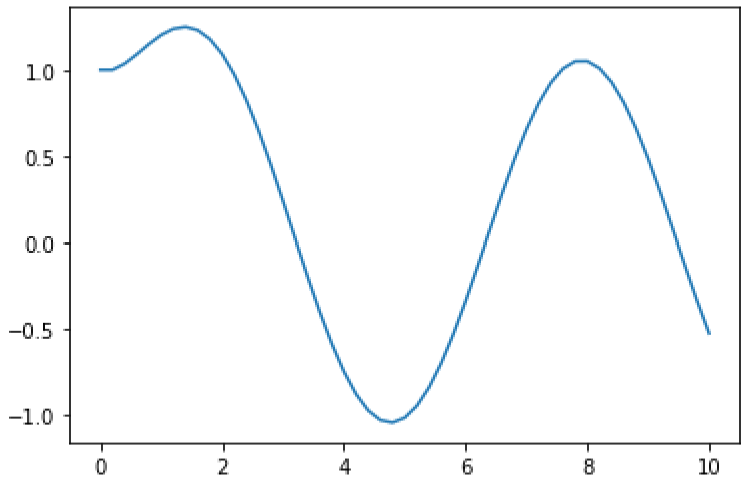
In [ ]:

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

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 [6]:

```
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 #21 βηματα, h=0.5 # 51 βηματα h
t=np.linspace(0,10,N+1)
h=t[1]-t[0]

y=np.zeros(N+1)

y[0]=1 #Αρχική τιμή

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

#print(y[10])
#print(t[10])

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



Άσκηση 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 [8]:

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

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

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

N=[50,100,200,400]
err=np.zeros(4)
for j in range(4):

    t=np.linspace(0,10,N[j]+1) #διαμερισμος
    h=t[1]-t[0]

    y=np.zeros(N[j]+1) #Μηδενισμός του διανυσματος τιμων

    y[0]=1 #Αρχική τιμή

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

    err[j]=abs(y[-1]-y_exact(10)) #σφαλματα για αντιστοιχη διαμεριση

for j in range(3):
    print(err[j+1]/err[j])
```

```
0.509130622901394
0.503929109465842
0.5018269724508366
```

Άσκηση 4: Στην προηγούμενη Άσκηση 3, θεωρούμε τώρα ως

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

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

In [10]:

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

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

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

N=[50,100,200,400]
err=np.zeros(4)
for j in range(4):

    t=np.linspace(0,10,N[j]+1) #διαμερισμος
    h=t[1]-t[0]

    y=np.zeros(N[j]+1) #Μηδενισμός του διανυσματος τιμων

    y[0]=1 #Αρχική τιμή

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

    err[j]=max(abs(y-y_exact(t))) #σφαλματα για αντιστοιχη διαμεριση. Αφαίρεση αντιστοιχ
ων arrays

for j in range(3):
    print(err[j+1]/err[j])
```

```
0.48837809902118534
0.4942511605941411
0.4971822884936609
```