**Міністерство освіти і науки України**
**Національний технічний університет України**
“Київський політехнічний інститут”
Факультет електроніки
Кафедра мікроелектроніки

# ЛАБОРАТОРНА РОБОТА № 1

з курсу
«Теорія сигналів»
"Основи програмування мовою Python"

Студента 3 курсу
групи ДП-82
Мнацаканова Антона

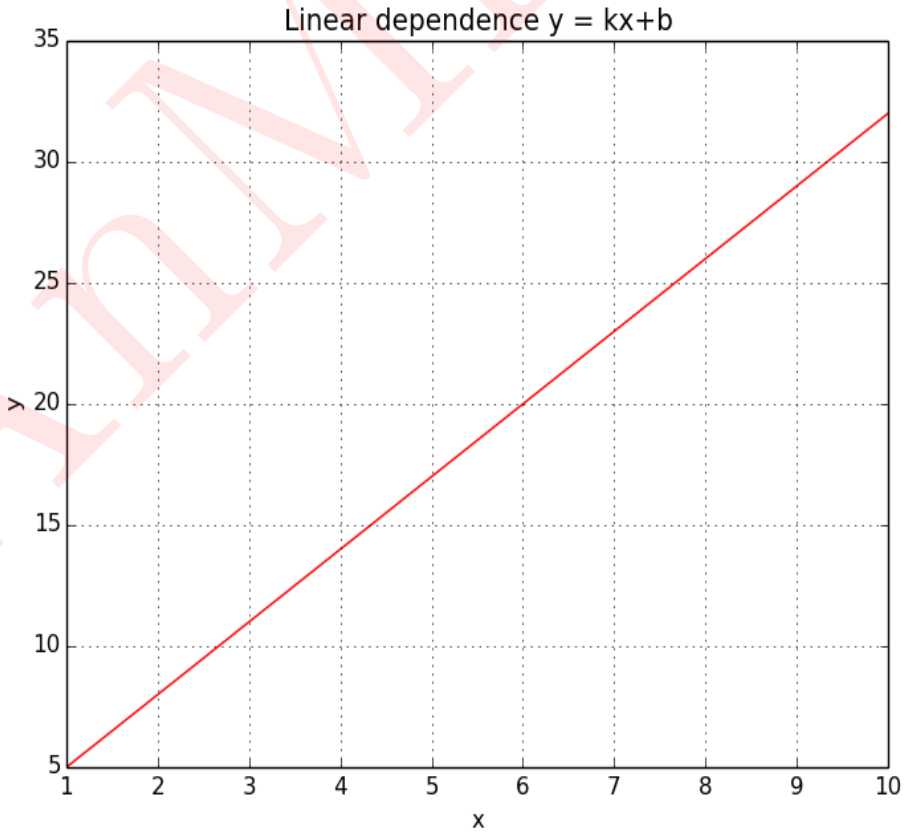


Київ — 2020

**Завдання, частина 1.**

4. Ознайомитися з написанням власних файлів-сценаріїв. У власному файлі-сценарії побудувати графік лінійної функції однієї змінної. Позначити вісі та заголовок графіку, нанести координатну сітку.

```
#!/usr/bin/env python
#coding=utf8
from numpy import array, arange, abs as np_abs
import numpy as np
from math import sin, pi
import matplotlib.pyplot as plt
import matplotlib as mpl
import random
fig, ax = plt.subplots()
mpl.rcParams['font.family'] = 'fantasy'
mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'

x = np.linspace(1, 10, 50)
k = 3
b = 2
y = k*x + b
plt.title("Linear dependence y = kx+b")
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.plot(x, y, "r")
plt.show()
```




5.1 Побудувати графіки синусоїд частот 1, 10, 50 Гц. Тривалість сигналів – 1 сек., частота

дискретизації 256 Гц. Графіки будувати в одному вікні, але в різних осях. Амплітуди кожної синусоїди повинні бути випадковими числами.

5.2 Виконати теж саме, але задавати амплітуду кожної синусоїди з клавіатури.

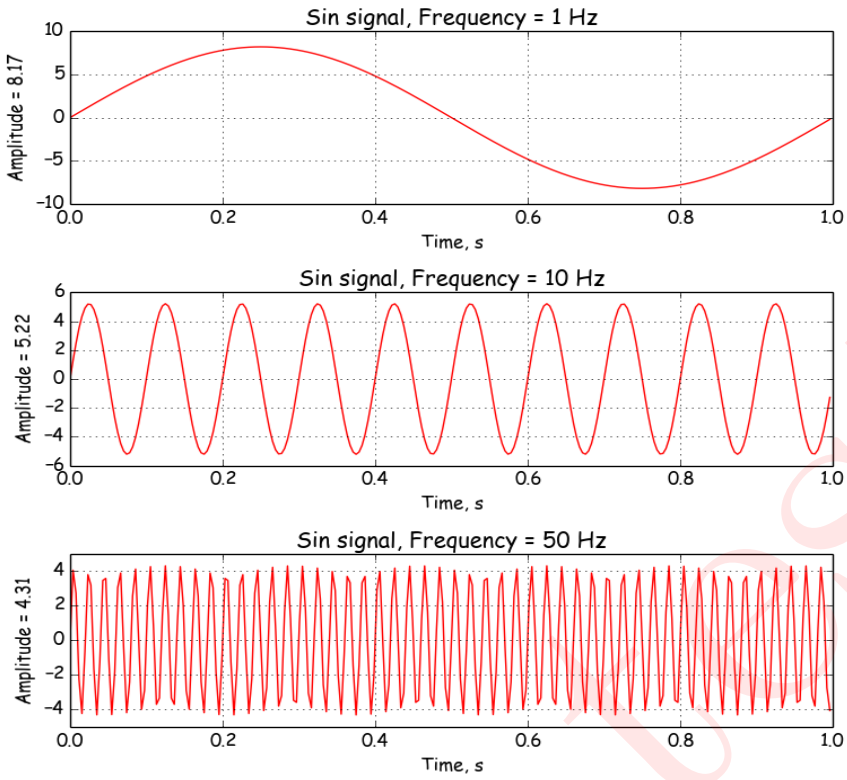
5.3 Підписати заголовок кожного графіку текстом, який буде містити значення частоти та амплітуди відповідної синусоїди.

```
from numpy import array, arange, abs as np_abs
from math import sin, pi
import matplotlib.pyplot as plt
import matplotlib as mpl
import random
mpl.rcParams['font.family'] = 'fantasy'
mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'

FD = 256
N = 256
F=1.0
w=(2.*pi*F/FD)
A=random.uniform(1,10)
```

(заміняємо всі ф-ї radom() на input() для вводу значень з клавіатури)

```
plt.subplot(3,1,1)
sin_sig = array([A*sin(w*t) for t in range(N)])
plt.plot(arange(N)/float(FD), sin_sig, 'r')
plt.xlabel('Time, s')
plt.ylabel('Applitude = '+ str("{0:.2f}".format(A) ))
plt.title('Sin signal, Frequency = ' + str("{0:.2f}".format(F) ))
plt.grid(True)
plt.hold(True)
FD = 256
N = 256
F=10.0
w=(2.*pi*F/FD)
A=random.uniform(1,10)
plt.subplot(3,1,2)
sin_sig = array([A*sin(w*t) for t in range(N)])
plt.plot(arange(N)/float(FD), sin_sig, 'r')
plt.xlabel('Time, s')
plt.ylabel('Applitude = '+ str("{0:.2f}".format(A) ))
plt.title('Sin signal, Frequency =' + str("{0:.2f}".format(F) ))
plt.grid(True)
plt.hold(True)
FD = 256
N = 256
F=50.0
w=(2.*pi*F/FD)
A=random.uniform(1,10)
plt.subplot(3,1,3)
sin_sig = array([A*sin(w*t) for t in range(N)])
plt.plot(arange(N)/float(FD), sin_sig, 'r')
plt.xlabel('Time, s')
plt.ylabel('Applitude = '+ str("{0:.2f}".format(A) ))
plt.title('Sin signal, Applitude = ' + str("{0:.2f}".format(F) ))
plt.grid(True)
plt.hold(True)
```

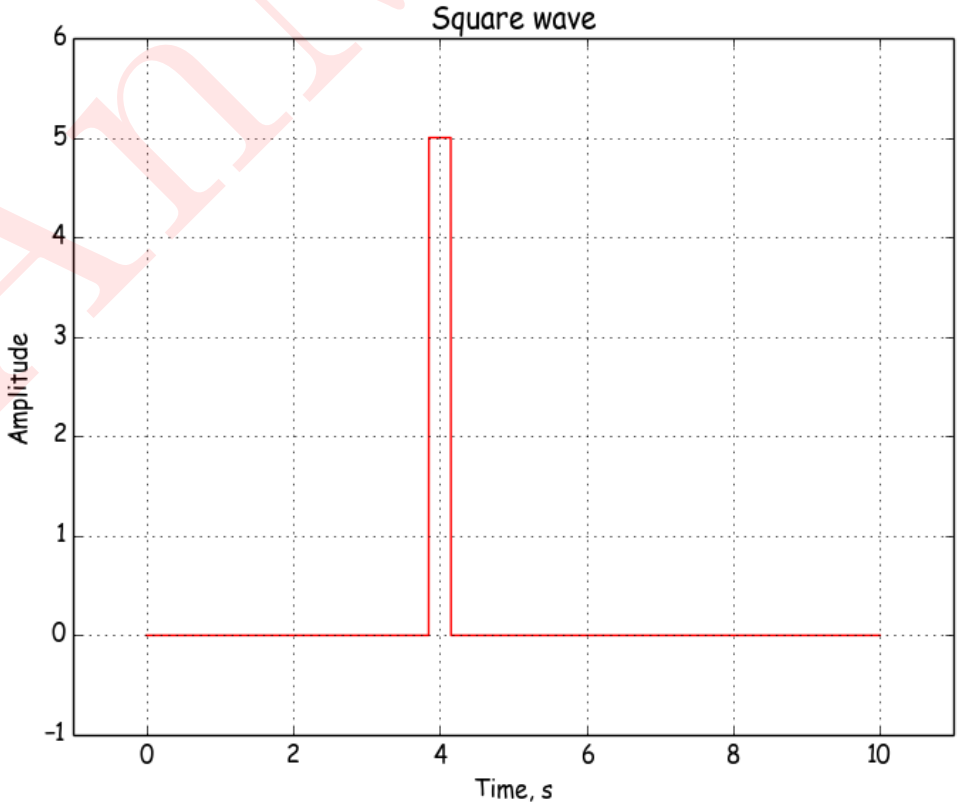
```
plt.show()
```




6. Ознайомитися з роботою функцій, що генерують прямокутні імпульси, гаусівські імпульси, трикутні імпульси та послідовності імпульсів заданої форми.

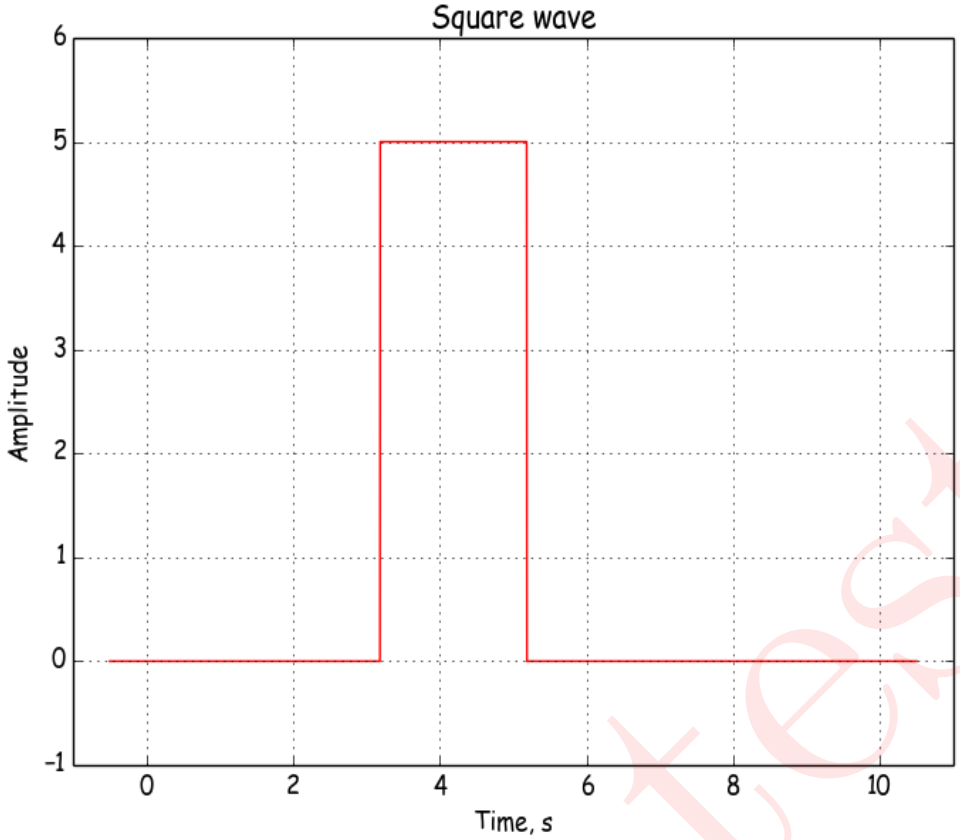
6.1 Побудувати одиночний прямокутний імпульс. Задати проміжок значень часу 10 секунд, частота дискретизації 256 Гц. Побудувати графік одиничного прямокутного імпульсу шириною 300 мс, з центром в момент часу 4 с.

lstinputlisting[language=Python]6.1.py



6.2 Написати файл-сценарій для побудови графіку прямокутного імпульсу, тривалість та амплітуда якого буде задаватися з клавіатури. Розташування імпульсу задавати випадковим числом, але передбачити перевірку, чи не виходе імпульс за межі графіка.

```
#!/usr/bin/env python
#coding=utf8
from numpy import array, arange, abs as np_abs

from math import sin, pi
import matplotlib.pyplot as plt
import matplotlib as mpl
import random
import numpy as np
mpl.rcParams['font.family'] = 'fantasy'
mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'

fd = 256.0
t_min = -0.5
t_max = 10.5
delta_t = 1.0/fd;
print ('Imput Amplitude: ')
A = input()
print ('Imput Pulse Width((0;10]): ')
b = input()
a = random.uniform(0,10-b)
if b>10:
        print ("Impulse outside the schedule")
        exit()
def f(x):
    if x < a:
        return 0;
    if x < a+b:
        return A;
    return 0;

x = [];
y = [];
t = t_min;
while t<=t_max:
    x.append(t);
    y.append(f(t));
    t += delta_t

np.savez('example_2', x, y, A)
plt.plot(x, y, 'r')
plt.xlabel('Time, s')
plt.ylabel('Amplitude')
plt.title('Square wave')
plt.grid(True)
axes = plt.gca()
axes.set_xlim([-1,11])
axes.set_ylim([-1,A+1])
plt.show()
```

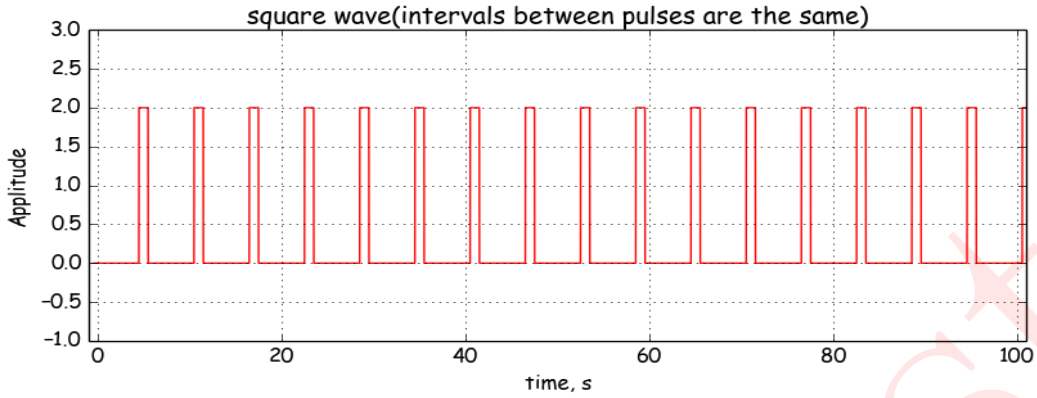


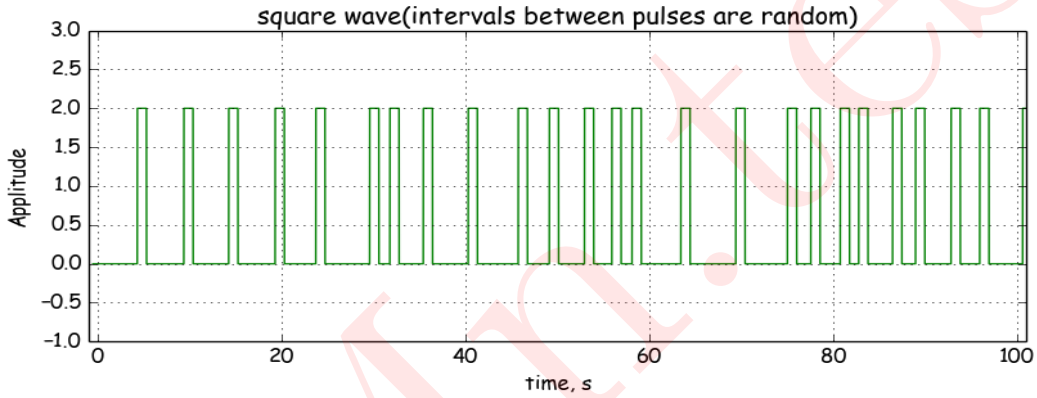

6.3 Побудувати послідовність прямокутних імпульсів для двох випадків: а) коли інтервали між імпульсами однакові, б) коли інтервали між імпульсами випадкові і задаються програмно.

```
#!/usr/bin/env python
#coding=utf8
from numpy import array, arange, abs as np_abs

from math import sin, pi
import matplotlib.pyplot as plt
import matplotlib as mpl
import random
mpl.rcParams['font.family'] = 'fantasy'
mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'

fd = 256.0
t_min = -0.5
t_max = 100.5
delta_t = 1.0/fd;
print ('Maintain the amplitude: ')
A = input()
print ('Enter the impulse width( (0;10] ): ')
b = input()
print ('Enter the intervals between impulses: ')
a = input()
#a = random.uniform(0,10-b)
if b>10:
        print ("Impulse outside the schedule")
        exit()
def f(x,imp_begin,imp_width):
    if x < imp_begin:
        return 0;
```

```
    if x < imp_begin+imp_width:
        return A;
    return 0;



x = [];
y = [];
t = t_min;
while t<=t_max:
    aa = t
    bb = t + a + b
    cc = t+a
    while aa<=bb:
        x.append(t);
        y.append(f(t,cc,b));
        aa += delta_t
        t += delta_t

x1 = [];
y1 = [];
t = t_min;
while t<=t_max:
    aa = t
    aaa = random.uniform(1,5)
    bb = t + aaa + b
    cc = t+aaa
    while aa<=bb:
        x1.append(t);
        y1.append(f(t,cc,b));
        aa += delta_t
        t += delta_t

plt.subplot(2,1,1)
plt.plot(x, y, 'r')
plt.xlabel('Time, s')
plt.ylabel('Amplitude')
plt.title('Square wave(intervals between pulses are the same)')
plt.grid(True)
axes = plt.gca()
axes.set_xlim([-1,101])
axes.set_ylim([-1,A+1])
plt.hold(True)



plt.subplot(2,1,2)
plt.plot(x1, y1, 'g')
plt.xlabel('Time, s')
plt.ylabel('Amplitude')
plt.title('Square wave(intervals between pulses are random)')
plt.grid(True)
axes = plt.gca()
axes.set_xlim([-1,101])
```

```
axes.set_ylim([-1,A+1])
plt.hold(True)
plt.show()
```




7 Зберегти дані розрахунку функції в файл. Прочитати їх із файлу в іншому сценарії, побудувати графік функції.

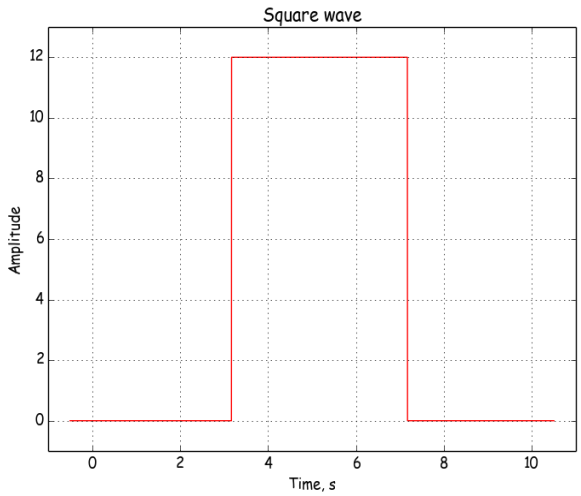
Writing

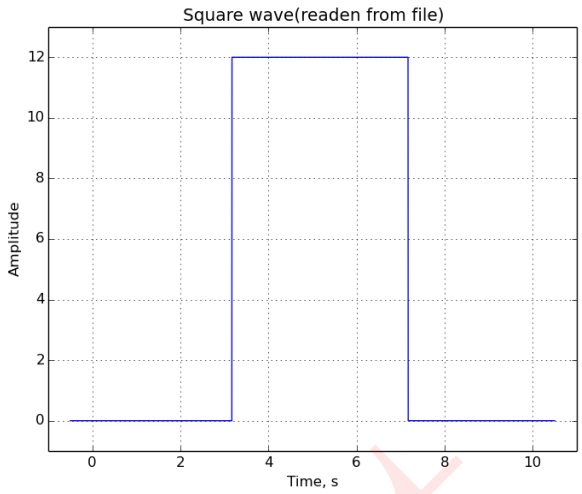
```
from numpy import array, arange, abs as np_abs
from math import sin, pi
import matplotlib.pyplot as plt
import matplotlib as mpl
import random
import numpy as np
mpl.rcParams['font.family'] = 'fantasy'
mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'

fd = 256.0
t_min = -0.5
t_max = 10.5
delta_t = 1.0/fd;

print ('Amplitude:')
A = input()
print ('Pulse Width((0;10]): ')
b = input()
a = random.uniform(0,10-b)
```

```
if b>10:
        print ("Impulse outside the schedule!")
        exit()
def f(x):
    if x < a:
        return 0;
    if x < a+b:
        return A;
    return 0;

x = [];
y = [];
t = t_min;
while t<=t_max:
    x.append(t);
    y.append(f(t));
    t += delta_t
np.savez('example_2', x, y, A) #SAVING
plt.plot(x, y, 'r')
plt.xlabel('Time, s')
plt.ylabel('Amplitude')
plt.title('Square wave')
plt.grid(True)
axes = plt.gca()
axes.set_xlim([-1,11])
axes.set_ylim([-1,A+1])
plt.show()
```

Reading

```
#!/usr/bin/env python
#coding=utf8
import numpy as np
import matplotlib.pyplot as plt
ex_2 = np.load('example_2.npz')
ex_2.files
x = ex_2['arr_0']
y = ex_2['arr_1']
A = ex_2['arr_2']

plt.plot(x, y, A, 'r')
plt.xlabel('Time, s')
plt.ylabel('Amplitude')
plt.title('Square wave(readen from file)')
plt.grid(True)
axes = plt.gca()
axes.set_xlim([-1,11])
axes.set_ylim([-1,A+1])
plt.show()
```

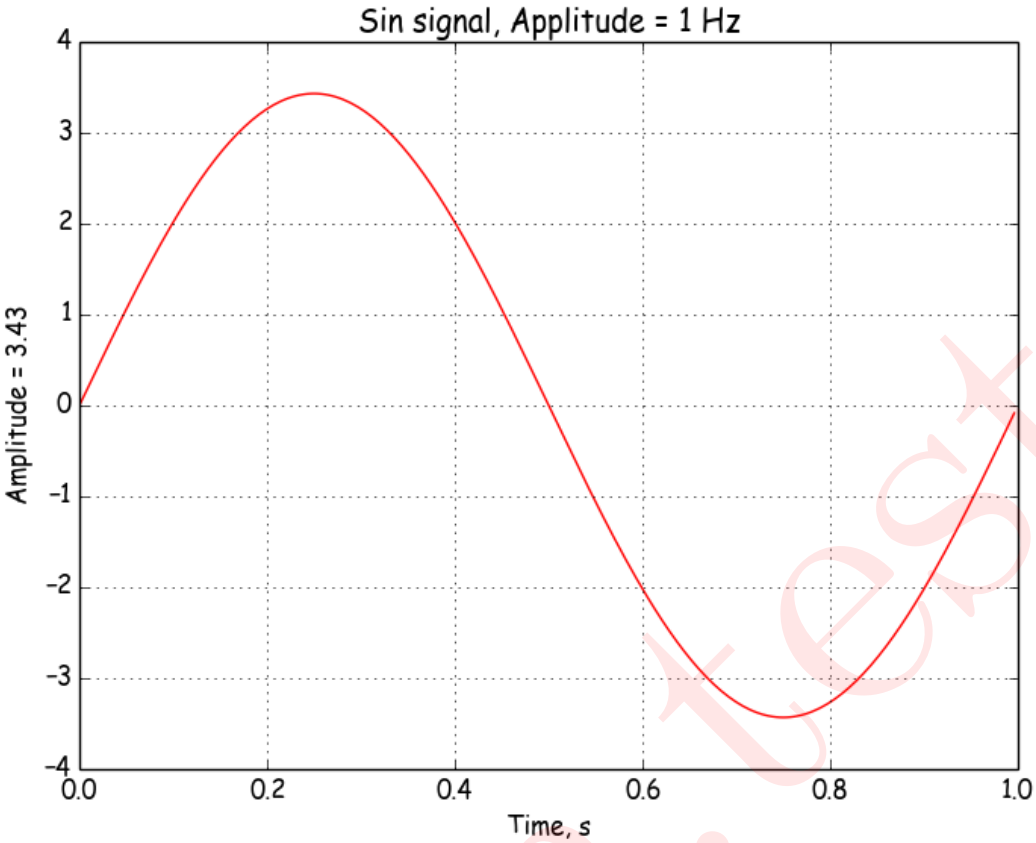



8 8. Побудувати власний файл-функцію для побудови графіка синусоїдального сигналу із заданою частотою, амплітудою та тривалістю для частоти дискретизації 256 Гц. В якості вихідного параметру функції вивести середнє значення синусоїди.

```
#!/usr/bin/env python
#coding=utf8
from numpy import array, arange, abs as np_abs
import numpy as np
from math import sin, pi
import matplotlib.pyplot as plt
import matplotlib as mpl
import random
fig, ax = plt.subplots()
mpl.rcParams['font.family'] = 'fantasy'
mpl.rcParams['font.fantasy'] = 'Comic Sans MS, Arial'

x = np.linspace(1, 10, 50)
k = 3
b = 2
y = k*x + b
plt.title("Linear dependence y = kx+b")
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.plot(x, y, "r")
plt.show()
```

Results




2.18578465652