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"

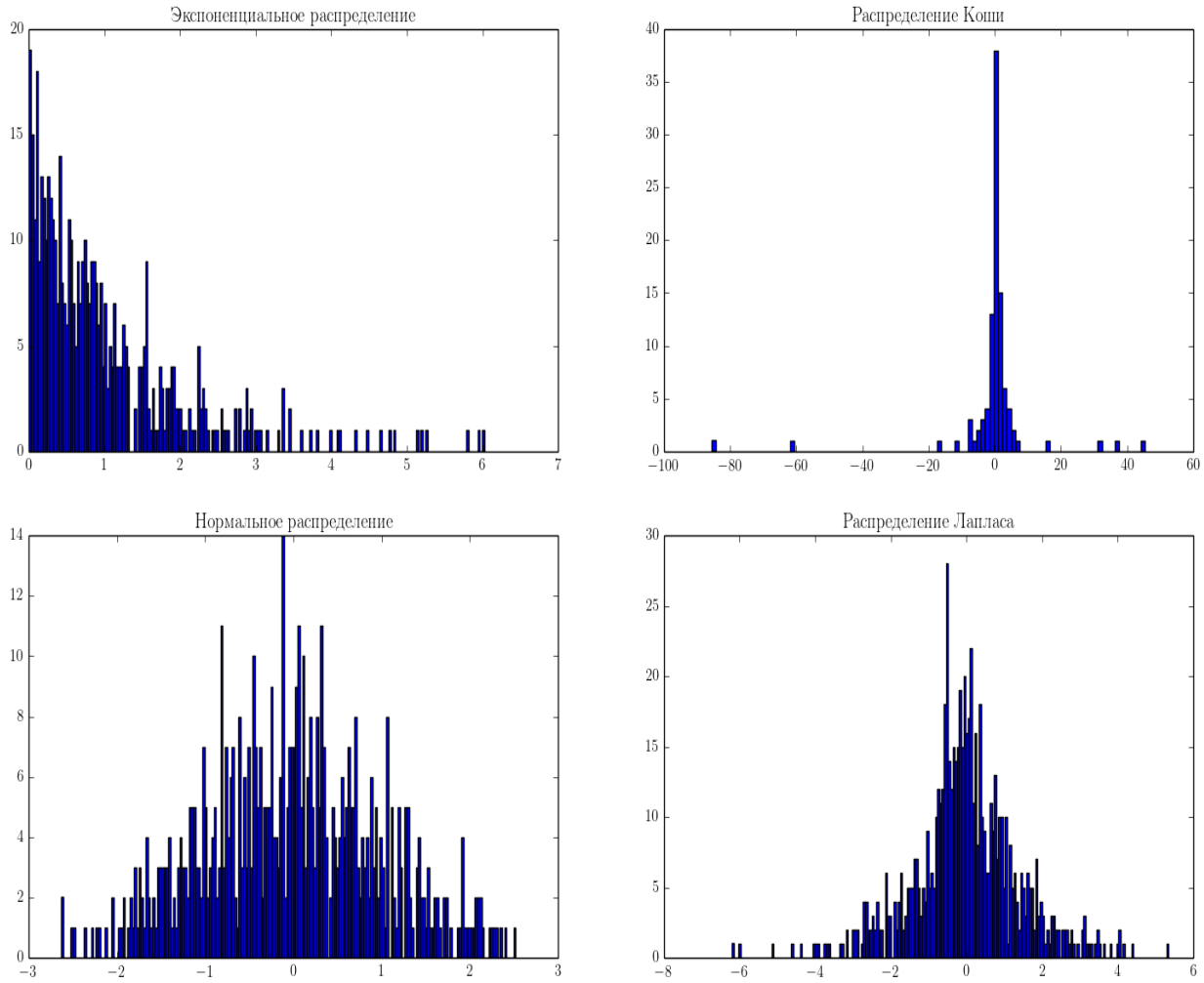
Студента 3 курсу
групи ДП-81
Фіцая Руслана

Київ — 2020

Пункт 3

Ознайомитися:
- з задаванням масиву, елементи якого є арифметичною послідовністю;
- з роботою функцій генерації випадкових чисел із заданими густинами розподілу імовірності.
Ознайомитися з функцією побудови гістограм, побудувати гістограми випадкових чисел з різними розподілами густини ймовірності;

```
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
from numpy import array, arange, abs as np_abs
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import rc

rc('text', usetex=True)
rc('text.latex',unicode=True)
rc('text.latex',preamble= '\usepackage[utf8]{inputenc}')
rc('text.latex',preamble= '\usepackage[russian]{babel}')

mas = [i for i in range(150)]#massive
print(mas)

fig ,axs = plt.subplots(2,2)
ax0, ax1, ax2, ax3 = axs.flatten()

a = np.random.standard_exponential(500)
b = np.random.standard_cauchy(100)
c = np.random.normal(size = 600)
d = np.random.laplace(size = 700)


axs[0,0].hist(a, bins=200)
ax0.set_title(u'Экспоненциальное распределение' )
axs[0,1].hist(b, bins=100)
ax1.set_title(u'Распределение Коши')
axs[1,0].hist(c, bins=200)
ax2.set_title(u'Нормальное распределение')
axs[1,1].hist(d, bins=200)
ax3.set_title(u'Распределение Лапласа')
plt.show()
```

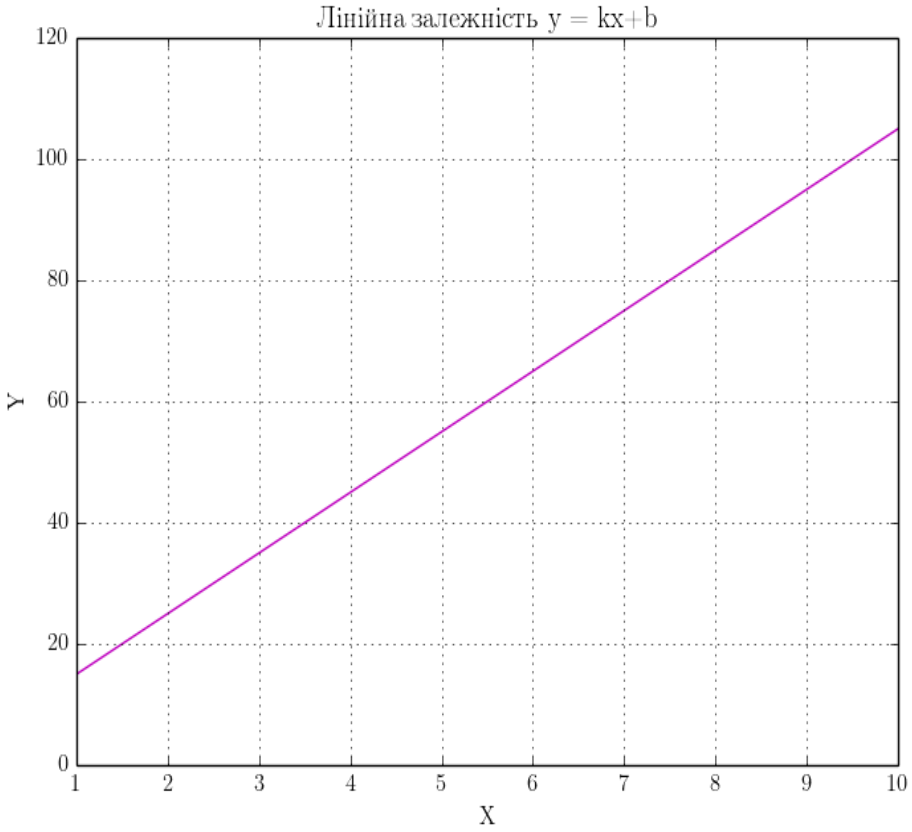


Пункт 4

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

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
fig, ax = plt.subplots()
from matplotlib import rc

rc('text', usetex=True)
rc('text.latex',unicode=True)
rc('text.latex',preamble='\usepackage[utf8]{inputenc}')
rc('text.latex',preamble='\usepackage[russian]{babel}')

x = np.linspace(1, 10, 50)
k = 10
b = 5
y = k*x + b
```

```
plt.title(u"Лінійна залежність y = kx+b")
plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True)
plt.plot(x, y, "m")
plt.show()
```



Пункт 5

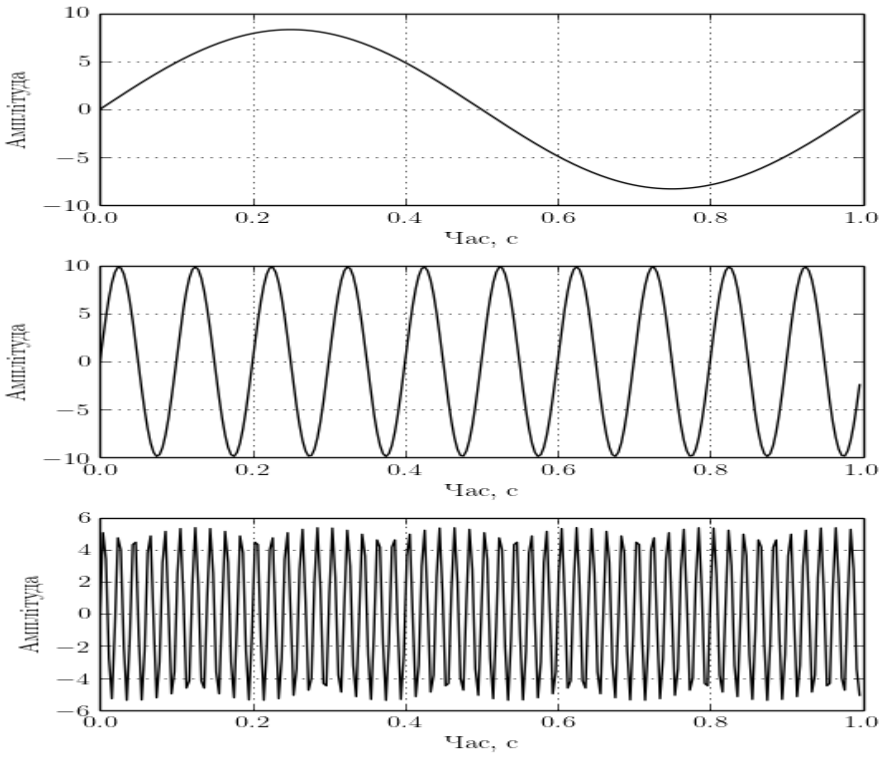
5.1 Побудувати графіки синусоїд частот 1, 10, 50 Гц. Тривалість сигналів – 1 сек., частота дискретизації 256 Гц. Графіки будувати в одному вікні, але в різних осях. Амплітуди кожної синусоїди повинні бути випадковими числами.

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
from matplotlib import rc

rc('text', usetex=True)
```

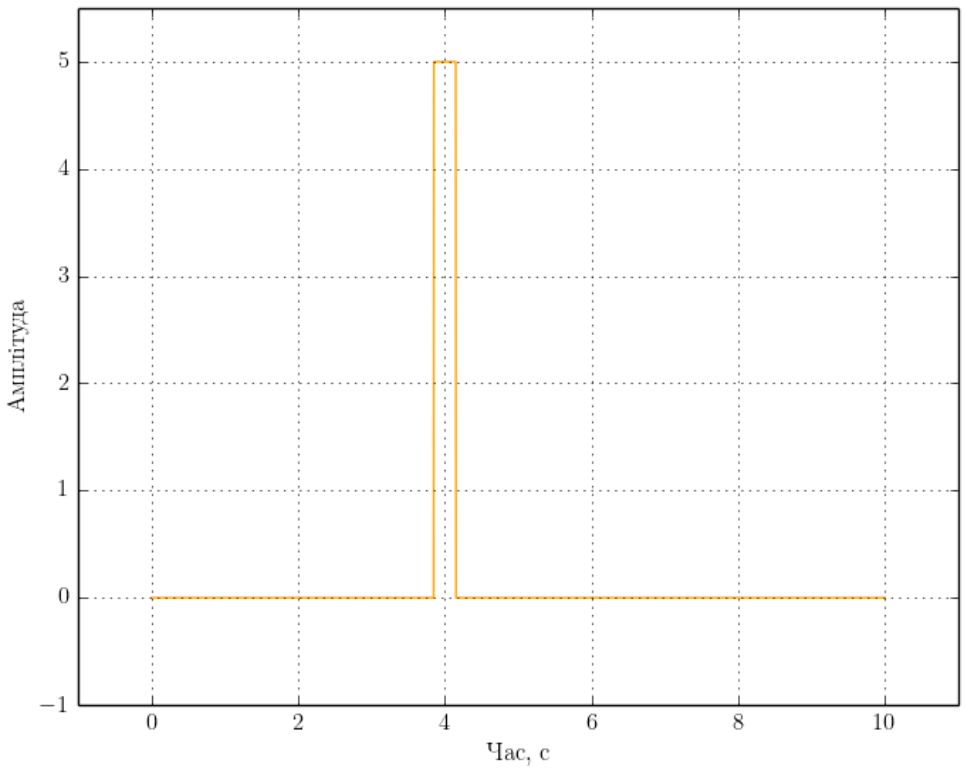
```
rc('text.latex',unicode=True)
rc('text.latex',preamble='\usepackage[utf8]{inputenc}')
rc('text.latex',preamble='\usepackage[russian]{babel}')

def one(fr, p):
    fd = 256.
    b = 256.0
    F=fr
    w=(2.*pi*F/fd)
    Amp=random.uniform(1,10)
    plt.subplot(3,1,p)
    signal = array([Amp*sin(w*t) for t in arange(b)])
    plt.plot(arange(b)/float(fd), signal, 'k')
    plt.xlabel(u'Час, с')
    plt.ylabel(u'Амплітуда')
    plt.grid(True)
    plt.hold(True)

one(1,1)
one(10,2)
one(50,3)

plt.show()
```



Пункт 6

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.

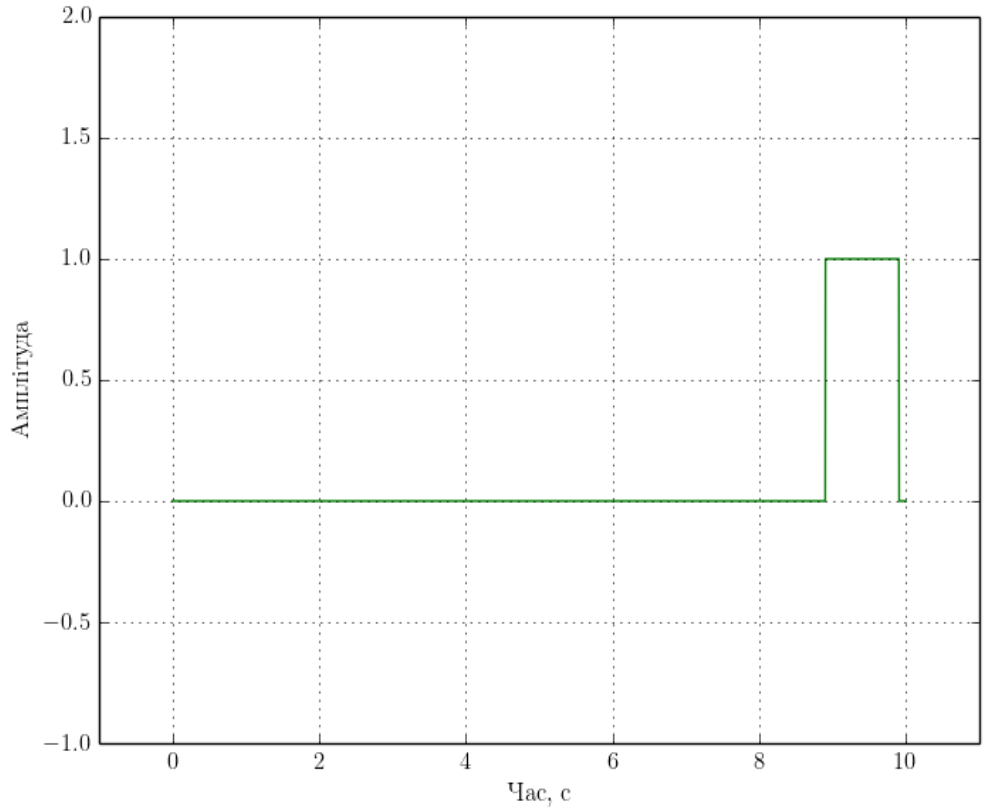
```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot
from matplotlib import rc

rc('text', usetex=True)
rc('text.latex',unicode=True)
rc('text.latex',preamble='\usepackage[utf8]{inputenc}')
rc('text.latex',preamble='\usepackage[russian]{babel}')

fd = 256.0
Amp = 5.0
b = 0.3
a = 4.0 - b/2;
tmin = 0
tmax = 10

data = [[],[]];
t = tmin;
while t<=tmax:
    data[0].append(t);
    if t < a:
        data[1].append(0);
    elif t < a+b:
        data[1].append(Amp);
    else:
        data[1].append(0);
    t += 1.0/fd;

matplotlib.pyplot.plot(data[0], data[1], 'orange')
plt.ylabel(u"Амплітуда")
plt.xlabel(u"Час, с")
matplotlib.pyplot.grid(True)
axes = matplotlib.pyplot.gca()
axes.set_xlim([-1,11])
axes.set_ylim([-1,Amp+0.5])
matplotlib.pyplot.show()
```



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.

```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib as mpl
from matplotlib import rc

rc('text', usetex=True)
rc('text.latex',unicode=True)
rc('text.latex',preamble='\usepackage[utf8]{inputenc}')
rc('text.latex',preamble='\usepackage[russian]{babel}')

def sqwave(p,end,T,aa,A,fd):
    x = []
    r = np.arange(p,end,1.0/fd)
    taue = aa+T
    x00 = r[(np.abs(r - aa)).argmin()]
    x01 = r[(np.abs(r - taue)).argmin()]
    if aa+T > end:
        return 1
    for i in r:
        if i >= x00 and i <= x01:
            x.append(A)
        else:
            x.append(0.0)
    return [r, x]

print("Входная длительность импульса: ")
```

```
duration = float(input())
print("A: ")
Amp = float(input())
a = sqwave(0, 10, duration, np.random.rand(1) * 10.0, Amp, 256.0)
while a == 1:
    a = sqwave(0, 10, duration, np.random.rand(1) * 10.0, Amp, 256.0)
plt.plot(a[0], a[1], 'g')
plt.ylabel(u"Амплітуда")
plt.xlabel(u"Час, с")
plt.grid(True)
axes = plt.gca()
axes.set_xlim([-1,11])
axes.set_ylim([-1,Amp+1])
plt.show()
```

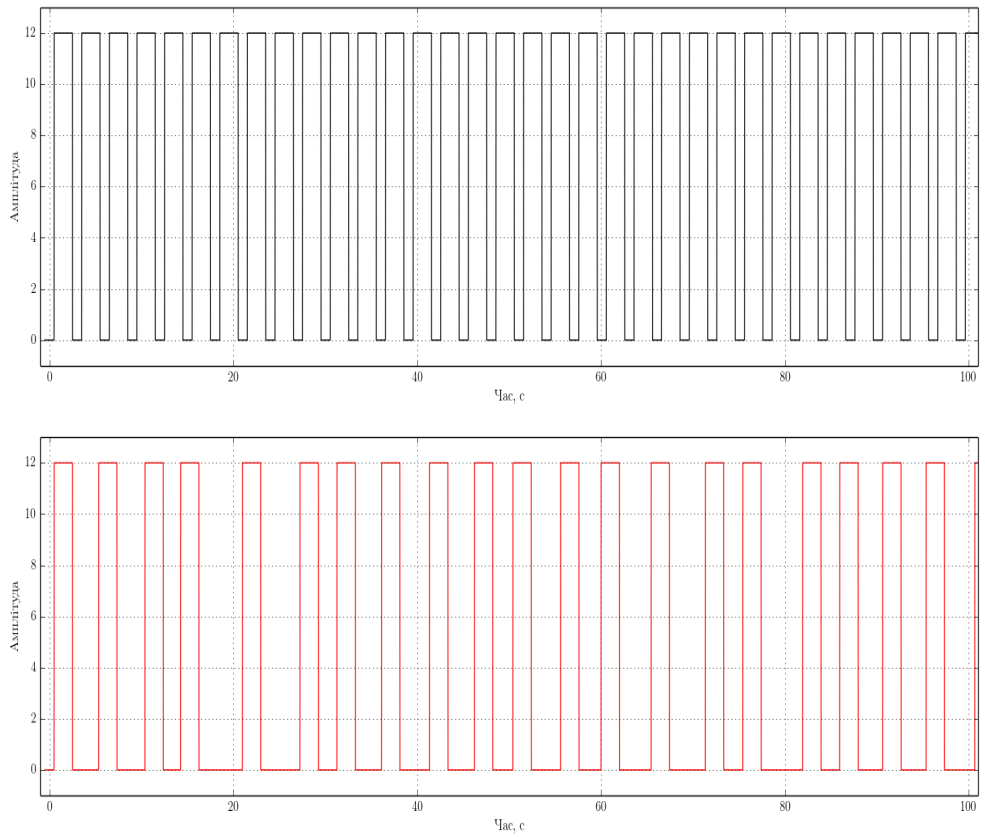


6.3 Побудувати послідовність прямокутних імпульсів для двох випадків: а) коли інтервали між імпульсами однакові, б) коли інтервали між імпульсами випадкові і задаються програмно.

```
from numpy import array, arange, abs as np_abs
import json
from math import sin, pi
import matplotlib.pyplot as plt
import matplotlib as mpl
import random
from matplotlib import rc

rc('text', usetex=True)
rc('text.latex',unicode=True)
```

```
rc('text.latex',preamble='\usepackage[utf8]{inputenc}')
rc('text.latex',preamble='\usepackage[russian]{babel}')


fd = 256.0
tmin = -0.5
tmax = 100.5
delta_t = 1.0/fd;
print ('A: ')
A = input()
print ('Ширина от 1 до 10: ')
b = input()
print ('Интервалы между импульсами: ')
a = input()
if b>10:
    print ("Импульс вылез за границы!!!")
    exit()

def f(x,imp_begin,imp_width):
    if x > imp_begin and x < imp_begin+imp_width:
        return A;
    return 0;

def gen_a1():
    return a;

def gen_a2():
    return random.uniform(1,5);

def one(ff, position, color):
    x = [];
    y = [];
    t = tmin;
    while t<=tmax:
        aa = t
        aaa = ff();
        bb = t + aaa + b
        cc = t+a
        while aa<=bb:
            x.append(t);
            y.append(f(t,cc,b));
            aa += delta_t
            t += delta_t
    plt.subplot(2,1,position)
    plt.plot(x, y, color)
    plt.xlabel(u'Час, c')
    plt.ylabel(u'Амплітуда')
    plt.grid(True)
    axes = plt.gca()
    axes.set_xlim([-1,101])
    axes.set_ylim([-1,A+1])
    plt.hold(True)
    return [x, y];
```

```
[x, y] = one(gen_a1, 1, 'k')
[x, y] = one(gen_a2, 2, 'r')
with open('data.txt', 'w') as outfile:
    json.dump({'x': x, 'y': y, 'A': A}, outfile)
plt.show()
```

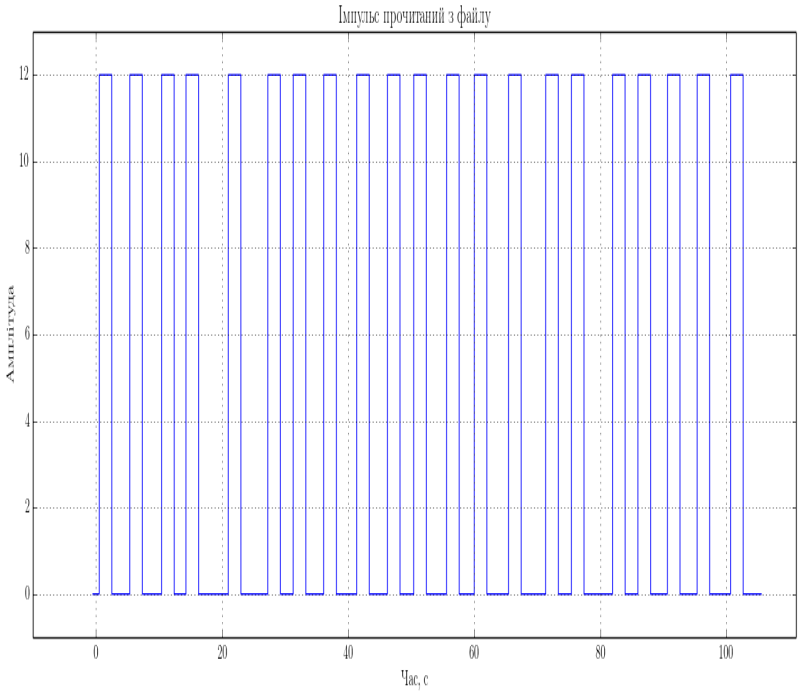


Пункт 7

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

```
import json;
import matplotlib.pyplot as plt
from matplotlib import rc

rc('text', usetex=True)
rc('text.latex',unicode=True)
rc('text.latex',preamble='\usepackage[utf8]{inputenc}')
rc('text.latex',preamble='\usepackage[russian]{babel}')

def read_f(name):
    with open(name) as json_file:
        data = json.load(json_file)
        return [data['x'], data['y'], data['A']];
```

```
[x, y, A] = read_f('data.txt')
plt.plot(x, y, A, 'm')
plt.xlabel(u'Час, с')
plt.ylabel(u'Амплітуда')
plt.title(u'мпульсI прочитаний з файлу')
plt.grid(True)
axes = plt.gca()
axes.set_xlim([-10,111])
axes.set_ylim([-1,A+1])
plt.show()
```

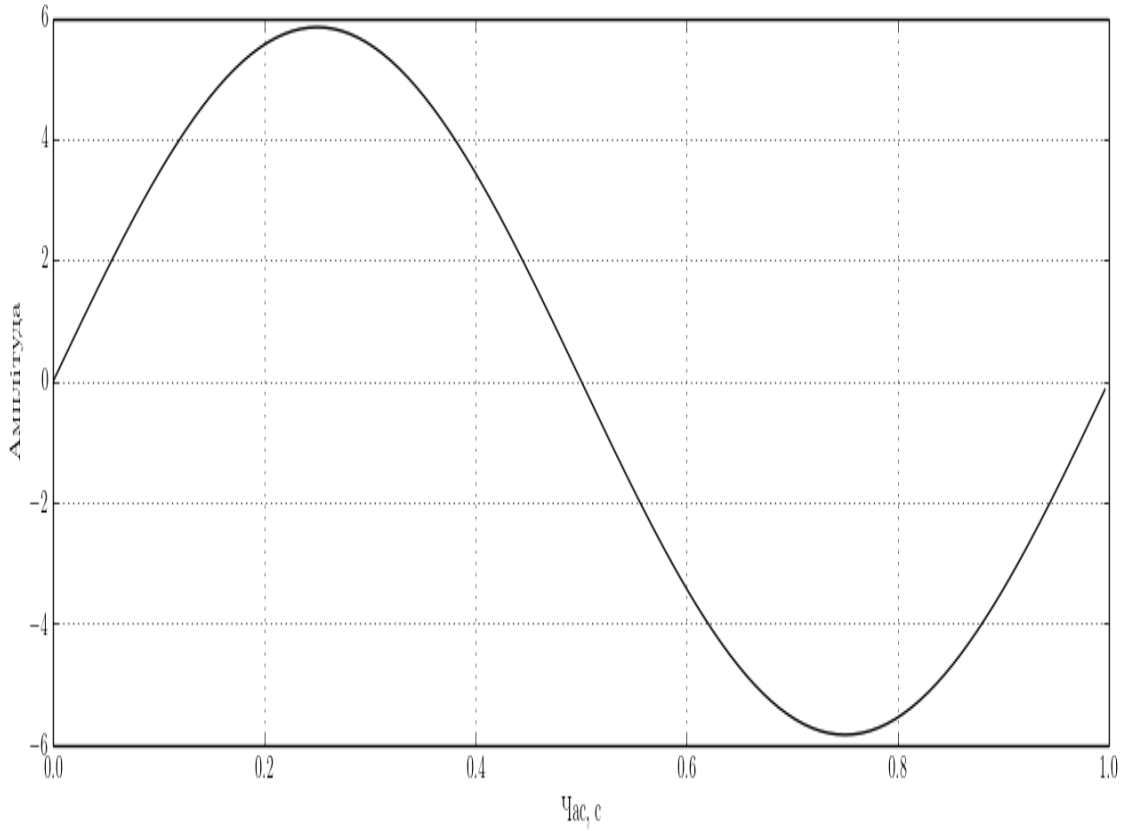


Пункт 8

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.

```
from numpy import array, arange, abs as np_abs
from math import sin, pi
import matplotlib.pyplot as plt
import random
from matplotlib import rc

rc('text', usetex=True)
rc('text.latex',unicode=True)
rc('text.latex',preamble='\usepackage[utf8]{inputenc}')
rc('text.latex',preamble='\usepackage[russian]{babel}')

def f():
        fd = 256.
        b = 256.0
```

```
        F=1.0
        w=(2.*pi*F/fd)
        Amp=random.uniform(1,10)
        signal = array([Amp*sin(w*t) for t in arange(b)])
        plt.plot(arange(b)/float(fd), signal, 'k')
        plt.xlabel(u'Час, с')
        plt.ylabel(u'Амплітуда')
        plt.grid(True)
        plt.show()
        return Amp*0.637

print (f())
```



Середнє значення синусоїди = 2.18