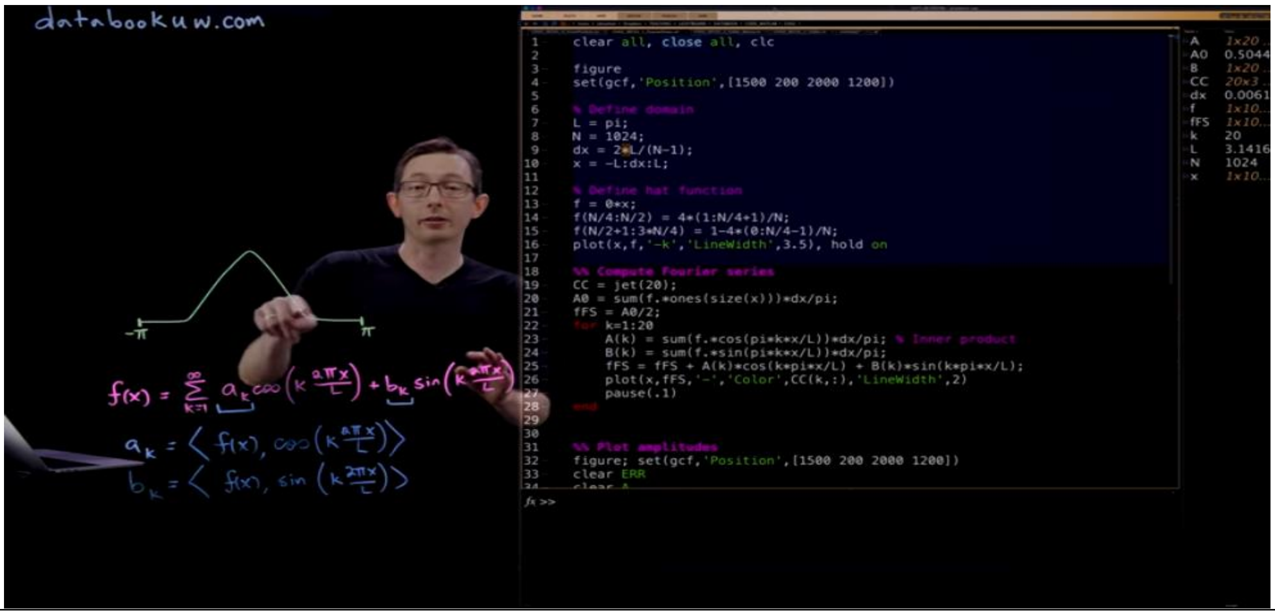
DAILY ASSESSMENT FORMAT

Date:	**25/5/2020**	**Name:**	**Shilpa S**
Course:	**Digital signal processing**	**USN:**	**4AL14EC078**
Topic:	**Introduction to fourier series and fourier transform, Hilbert transform, Fourier series using matlab and python**	**Semester & Section:**	**8th sem A sec**
Github Repository:	**Shilpa_online**		

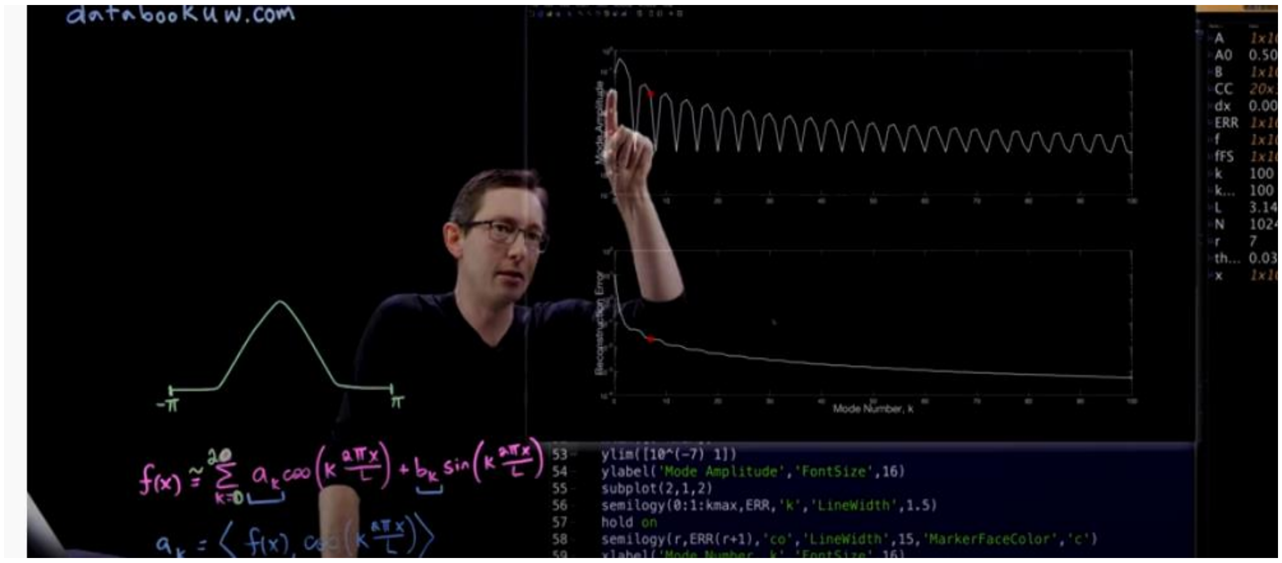
FORENOON SESSION DETAILS
Image of session


databookuw.com
$-\pi$
π
$f(x) = \sum_{k=1}^{\infty} \underbrace{a_k}\cos\left(k\frac{2\pi x}{L}\right) + \underbrace{b_k}\sin\left(k\frac{2\pi x}{L}\right)$
$a_k = \langle f(x), \cos\left(k\frac{2\pi x}{L}\right)\rangle$
$b_k = \langle f(x), \sin\left(k\frac{2\pi x}{L}\right)\rangle$

```
clear all, close all, clc

figure
set(gcf,'Position',[1500 200 2000 1200])

% Define domain
L = pi;
N = 1024;
dx = 2*L/(N-1);
x = -L:dx:L;

% Define hat function
f = 0*x;
f(N/4:N/2) = 4*(1:N/4+1)/N;
f(N/2+1:3*N/4) = 1-4*(0:N/4-1)/N;
plot(x,f,'-k','LineWidth',3.5), hold on

%% Compute Fourier series
CC = jet(20);
A0 = sum(f.*ones(size(x)))*dx/pi;
fFS = A0/2;
for k=1:20
    A(k) = sum(f.*cos(pi*k*x/L))*dx/pi; % Inner product
    B(k) = sum(f.*sin(pi*k*x/L))*dx/pi;
    fFS = fFS + A(k)*cos(k*pi*x/L) + B(k)*sin(k*pi*x/L);
    plot(x,fFS,'-','Color',CC(k,:),'LineWidth',2)
    pause(.1)
end


%% Plot amplitudes
figure; set(gcf,'Position',[1500 200 2000 1200])
clear ERR
clear A
```

```
fx >>
```

```
A    1x20 ...
A0   0.5044
B    1x20 ...
CC   20x3 ...
dx   0.0061
f    1x10...
fFS  1x10...
k    20
L    3.1416
N    1024
x    1x10...
```


databookuw.com
Mode Amplitude
Reconstruction Error
Mode Number, k
$-\pi$
π
$f(x) \approx \sum_{k=0}^{20} \underbrace{a_k} \cos\left(k\frac{2\pi x}{L}\right) + \underbrace{b_k} \sin\left(k\frac{2\pi x}{L}\right)$
$a_k = \left\langle f(x), \cos\left(k\frac{2\pi x}{L}\right)\right\rangle$

```
ylim([10^(-7) 1])
ylabel('Mode Amplitude','FontSize',16)
subplot(2,1,2)
semilogy(0:1:kmax,ERR,'k','LineWidth',1.5)
hold on
semilogy(r,ERR(r+1),'co','LineWidth',15,'MarkerFaceColor','c')
xlabel('Mode Number, k','FontSize',16)
```

Report –

A Fourier series is a periodic function composed of harmonically related sinusoids, combined by a weighted summation. With appropriate weights, one cycle (or *period*) of the summation can be made to approximate an arbitrary function in that interval (or the entire function if it too is periodic). As such, the summation is a synthesis of another function. The discrete-time Fourier transform is an example of Fourier series. The process of deriving the weights that describe a given function is a form of Fourier analysis. For functions on unbounded intervals, the analysis and synthesis analogies are Fourier transform and inverse transform.

A Fourier series is an expansion of a periodic function $f(x)$ in terms of an infinite sum of sines and cosines. Fourier series make use of the orthogonality relationships of the sine and cosine functions. The computation and study of Fourier series is known as harmonic analysis and is extremely useful as a way to break up an *arbitrary* periodic function into a set of simple terms that can be plugged in, solved individually, and then recombined to obtain the solution to the original problem or an approximation to it to whatever accuracy is desired or practical. Examples of successive approximations to common functions using Fourier series are illustrated above.

In particular, since the superposition principle holds for solutions of a linear homogeneous ordinary differential equation, if such an equation can be solved in the case of a single sinusoid, the solution for an arbitrary function is immediately available by expressing the original function as a Fourier series and then plugging in the solution for each sinusoidal component. In some special cases where the Fourier series can be summed in closed form, this technique can even yield analytic solutions.

Any set of functions that form a complete orthogonal system have a corresponding generalized Fourier series analogous to the Fourier series. For example, using orthogonality of the roots of a Bessel function of the first kind gives a so-called Fourier-Bessel series. The computation of the (usual) Fourier series is based on the integral identities

$$\int_{-\pi}^{\pi} \sin(m\,x)\sin(n\,x)\,d\,x = \pi\delta_{m\,n} \qquad (1)$$

$$\int_{-\pi}^{\pi} \cos(m\,x)\cos(n\,x)\,d\,x = \pi\delta_{m\,n} \qquad (2)$$

$$\int_{-\pi}^{\pi} \sin(m\,x)\cos(n\,x)\,d\,x = 0 \qquad (3)$$

$$\int_{-\pi}^{\pi} \sin(m\,x)\,d\,x = 0 \qquad (4)$$

$$\int_{-\pi}^{\pi} \cos(m\,x)\,d\,x = 0 \qquad (5)$$

for $m, n \neq 0$, where $\delta_{m\,n}$ is the Kronecker delta.

Using the method for a generalized Fourier series, the usual Fourier series involving sines and cosines is obtained by taking $f_1(x) = \cos x$ and $f_2(x) = \sin x$. Since these functions form a complete orthogonal system over $[-\pi, \pi]$, the Fourier series of a function $f(x)$ is given by

$$f(x) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n \cos(n\,x) + \sum_{n=1}^{\infty} b_n \sin(n\,x), \qquad (6)$$

Where

$$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\,d\,x \qquad (7)$$

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos(n\,x)\,d\,x \qquad (8)$$

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin(nx)\,dx \tag{9}$$

and $n = 1$, 2, 3, ... Note that the coefficient of the constant term a_0 has been written in a special form compared to the general form for a generalized Fourier series in order to preserve symmetry with the definitions of a_n and b_n.

The Fourier cosine coefficient a_n and sine coefficient b_n are implemented in the Wolfram Language as FourierCosCoefficient[expr, t, n] and FourierSinCoefficient[expr, t, n], respectively. A Fourier series converges to the function $\overline{f}$ (equal to the original function at points of continuity or to the average of the two limits at points of discontinuity)

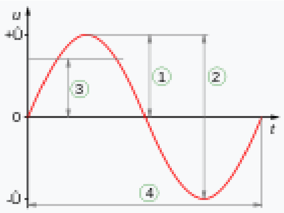
$$\overline{f} \equiv \begin{cases} \frac{1}{2}\left[\lim_{x\to x_0^-} f(x) + \lim_{x\to x_0^+} f(x)\right] & \text{for } -\pi < x_0 < \pi \\ \frac{1}{2}\left[\lim_{x\to -\pi^+} f(x) + \lim_{x\to \pi_-} f(x)\right] & \text{for } x_0 = -\pi, \pi \end{cases} \tag{10}$$

if the function satisfies so-called Dirichlet boundary conditions. Dini's test gives a condition for the convergence of Fourier series.

FOURIER TRANSFORM

A Fourier transform (FT) is a mathematical transform which decomposes a function (often a function of time, or a signal) into its constituent frequencies, such as the expression of a musical chord in terms of the volumes and frequencies of its constituent notes. The term *Fourier transform* refers to both the frequency domain representation and the mathematical operation that associates the frequency domain representation to a function of time.

The Fourier transform of a function of time is a complex-valued function of frequency, whose magnitude (absolute value) represents the amount of that frequency present in the original function, and whose argument is the phase offset of the basic sinusoid in that frequency. The Fourier transform is not limited to functions of time, but the domain of the original function is commonly referred to as the *time domain*. There is also an *inverse Fourier transform* that mathematically synthesizes the original function from its frequency domain representation, as proven by the Fourier inversion theorem.

- 

A sinusoidal curve, with peak amplitude (1), peak-to-peak (2), RMS (3), and wave period (4).

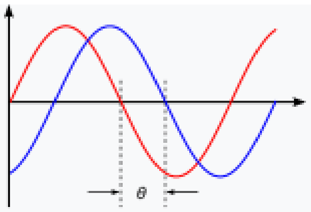
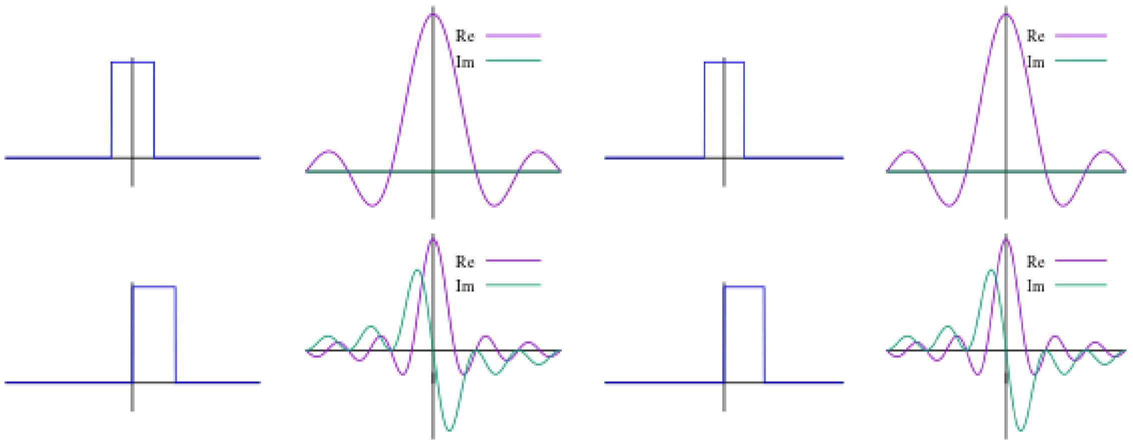
- 

Illustration of phase shift θ.

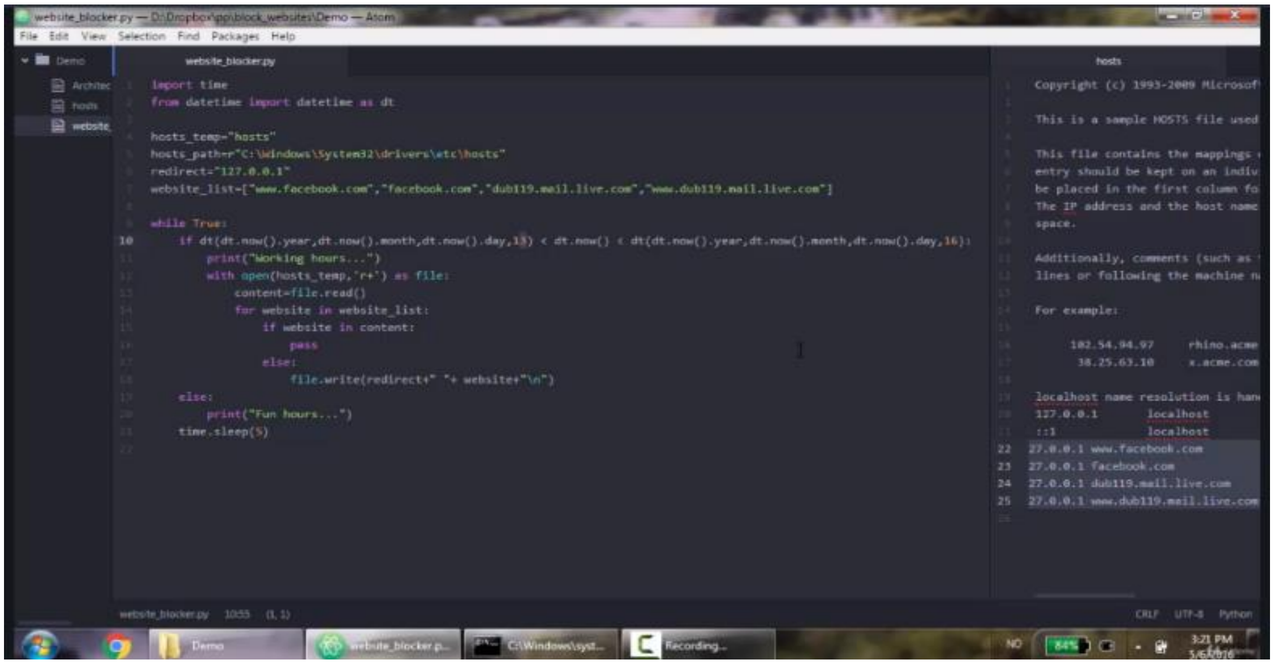


Linear operations performed in one domain (time or frequency) have corresponding operations in the other domain, which are sometimes easier to perform. The operation of differentiation in the time domain corresponds to multiplication by the frequency, so some differential equations are easier to analyze in the frequency domain. Also, convolution in the time domain corresponds to ordinary multiplication in the frequency domain (see Convolution theorem). After performing the desired operations, transformation of the result can be made back to the time domain. Harmonic analysis is the systematic study of the relationship between the frequency and time domains, including the kinds of functions or operations that are "simpler" in one or the other, and has deep connections to many areas of modern mathematics.

Date:25/5/2020
Course: Python
Topic: Fixing programming errors,
Application 3: Build a website blocker

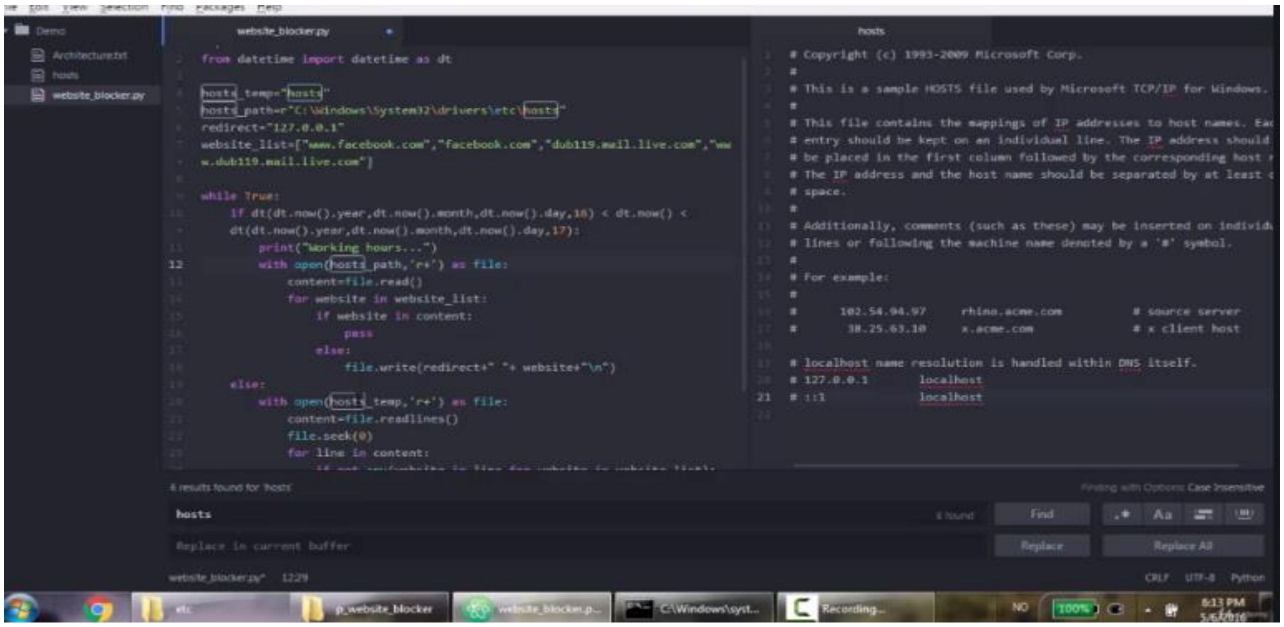
Name: Shilpa S
USN:4AL14EC078
Sem :8th sem
Section: A sec

AFTERNOON SESSION DETAILS



```
import time
from datetime import datetime as dt

hosts_temp="hosts"
hosts_path=r"C:\Windows\System32\drivers\etc\hosts"
redirect="127.0.0.1"
website_list=["www.facebook.com","facebook.com","dub119.mail.live.com","www.dub119.mail.live.com"]

while True:
    if dt(dt.now().year,dt.now().month,dt.now().day,13) < dt.now() < dt(dt.now().year,dt.now().month,dt.now().day,16):
        print("Working hours...")
        with open(hosts_temp,'r+') as file:
            content=file.read()
            for website in website_list:
                if website in content:
                    pass
                else:
                    file.write(redirect+" "+ website+"\n")
    else:
        print("Fun hours...")
    time.sleep(5)
```

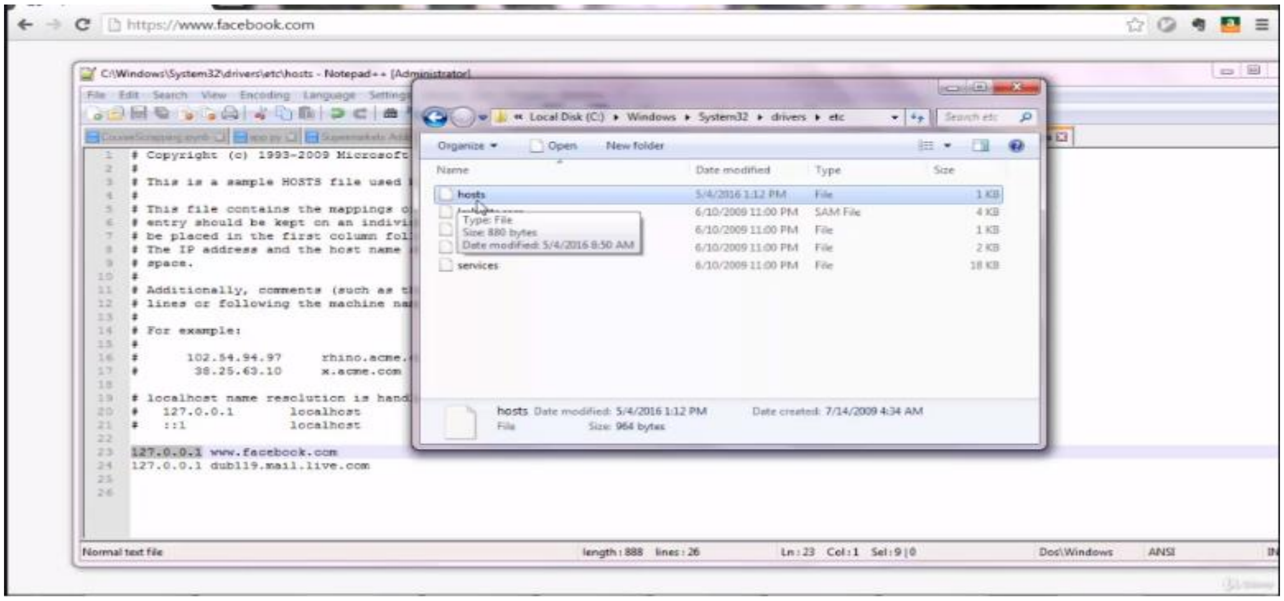
```
Copyright (c) 1993-2009 Microsof

This is a sample HOSTS file used

This file contains the mappings
entry should be kept on an indiv
be placed in the first column fo
The IP address and the host name
space.

Additionally, comments (such as
lines or following the machine n

For example:

     102.54.94.97     rhino.acme
      38.25.63.10     x.acme.com

localhost name resolution is han
127.0.0.1       localhost
::1             localhost
27.0.0.1 www.facebook.com
27.0.0.1 facebook.com
27.0.0.1 dub119.mail.live.com
27.0.0.1 www.dub119.mail.live.com
```



```
from datetime import datetime as dt

hosts_temp="hosts"
hosts_path=r"C:\Windows\System32\drivers\etc\hosts"
redirect="127.0.0.1"
website_list=["www.facebook.com","facebook.com","dub119.mail.live.com","www.dub119.mail.live.com"]

while True:
    if dt(dt.now().year,dt.now().month,dt.now().day,16) < dt.now() <
    dt(dt.now().year,dt.now().month,dt.now().day,17):
        print("Working hours...")
        with open(hosts_path,'r+') as file:
            content=file.read()
            for website in website_list:
                if website in content:
                    pass
                else:
                    file.write(redirect+" "+ website+"\n")
    else:
        with open(hosts_temp,'r+') as file:
            content=file.readlines()
            file.seek(0)
            for line in content:
```

```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Eac
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host
# The IP address and the host name should be separated by at least
# space.
#
# Additionally, comments (such as these) may be inserted on individu
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
# 127.0.0.1       localhost
# ::1             localhost
```

```
6 results found for 'hosts'
hosts
Replace in current buffer
```



```
# Copyright (c) 1993-2009 Microsoft
#
# This is a sample HOSTS file used
#
# This file contains the mappings o
# entry should be kept on an indivi
# be placed in the first column fol
# The IP address and the host name
# space.
#
# Additionally, comments (such as t
# lines or following the machine na
#
# For example:
#
#      102.54.94.97     rhino.acme.
#       38.25.63.10     x.acme.com

# localhost name resolution is hand
#   127.0.0.1       localhost
#   ::1             localhost

127.0.0.1 www.facebook.com
127.0.0.1 dub119.mail.live.com
```

REPORT:

Here is an another example for if not any(website in line for website in website_list)

```
>>> lines  = ["trees are good", "pool is fresh", "face is round"]
>>> website_list = ["face", "clock", "trend"]
>>> for line in lines:
...     any(website in line for website in website_list)
...
False
False
True
```

We start iterating over the items of website_list using a for loop. In the first iteration we would have:

any(website in "trees are good" for website in website_list)

Inside the parenthesis of any() there's another loop that iterates over website_list:

("face" in "trees are good")

("clock" in "trees are good")

("trend" in "trees are good")

If any of the above is True you get the expression evaluated to True. In this case none of them is true, so you get False. If you want to return True (if all of them are True), use all() instead of any().

So, the part any(website in line for website in website_list) will either be equal to True or False.

Scheduling a Python Program on a Server
Scheduling a Python program on a 24/7 server

Keeping your computer on 24-7 is not practical, so if you want to execute a Python script at a particular time every day, you probably need a computer that is on all the time.

PythonAnywhere gives you access to such a 24-7 computer. You can upload a Python script and schedule it to run at a certain time every day. This availability can be useful, for example, when you want to extract some values (e.g., weather data) from a website and generate a text file with the value or other reports every day.

To schedule a Python script for execution on PythonAnywhere, follow these simple steps:

1. Sign up for a free account at https://www.pythonanywhere.com.

2. Go to your Dashboard, Files, Upload a File, and upload the Python file you want to schedule for execution.

3. Go to Tasks and set the time of the day you want your script to be executed and type in the name of the Python file you uploaded (e.g., myscript.py). Note that the time you enter should be in UTC.

4. Click the Create button and you're done.

Your Python file will now be executed every day at your specified time. If you don't have a Python script and you're still confused about the benefit of this, here is a very simple Python script that you can use to try the above steps:

If you don’t have a Python script and you’re still confused about the benefits of this PythonAnywhere feature, here is a very simple Python script you can use to schedule for execution:

```
from datetime import datetime
with open(datetime.now().strftime("%Y-%m-%d-%H-%M-%S"), "w") as myfile:
myfile.write("Hi there!")
```

The above code creates a text file and writes the string “Hi there!” in that text file. The name of the text file will be the current date and time. For example one file name example would be 2018-02-16-18-20-33.txt.

That name is generated by datetime.now() indicating the date and time the script was executed. Every time the script is executed, the script generates a new text file with a different name. You will have a new text file created every day.