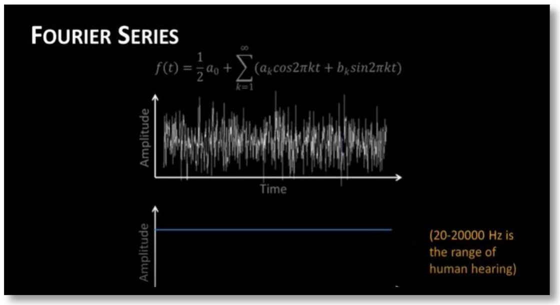
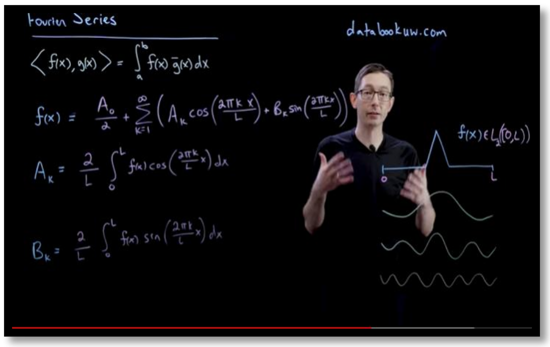
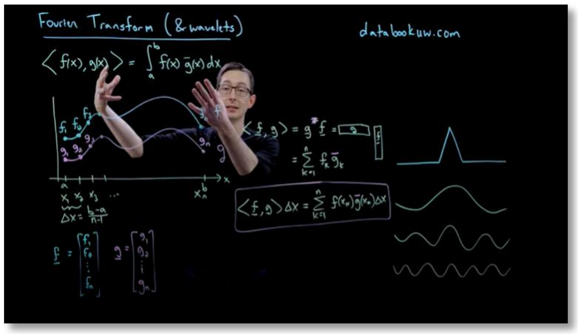
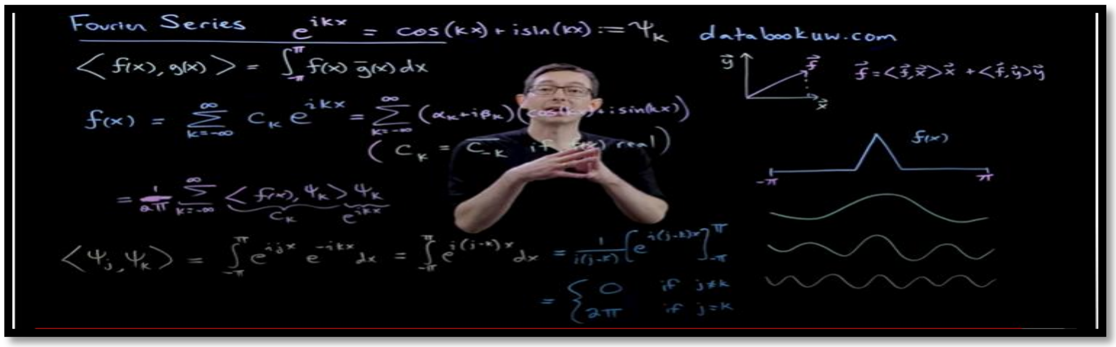
DAILY ASSESSMENT FORMAT

Date:	**25/05/2020**	**Name:**	**Davis S. Patel**
Course:	**Digital Signal Processing**	**USN:**	**4AL16EC045**
Topic:	**Introduction to Fourier Series & Fourier Transform Fourier Series – Part 1 & 2 Inner Product in Hilbert Transform Complex Fourier Series Fourier Series using Matlab (Use Octave to execute the code) Fourier Series using Python(Experience implementation using Python) Fourier Series and Gibbs Phenomena Using Matlab**	**Semester & Section:**	**8th - A**
GitHub Repository:	**Davis**		

FORENOON SESSION DETAILS
Image of session Fourier Analysis Overview STEVEN L. BRUNTON UNIVERSITY of WASHINGTON 0:04 / 7:28 FOURIER SERIES $f(t) = \frac{1}{2}a_0 + \sum_{k=1}^{\infty}(a_k cos2\pi kt + b_k sin2\pi kt)$ Amplitude Time Amplitude (20-20000 Hz is the range of human hearing) Fourier Series databookuw.com $\langle f(x), g(x) \rangle = \int_a^b f(x)\bar{g}(x)dx$ $f(x) = \frac{A_0}{2} + \sum_{k=1}^{\infty}\left(A_k \cos\left(\frac{2\pi k x}{L}\right) + B_k \sin\left(\frac{2\pi k x}{L}\right)\right)$ $A_k = \frac{2}{L}\int_0^L f(x)\cos\left(\frac{2\pi k}{L}x\right)dx$ $B_k = \frac{2}{L}\int_0^L f(x)\sin\left(\frac{2\pi k}{L}x\right)dx$ $f(x) \in L_2([0,L))$ Fourier Transform (& wavelets) databookuw.com $\langle f(x), g(x) \rangle = \int_a^b f(x)\bar{g}(x)dx$ $\langle f, g \rangle = g^* f$ $\langle f, g \rangle \Delta x = \sum_{k=1}^{n} f(x_k)\bar{g}(x_k)\Delta x$



Fourier Series $e^{ikx} = \cos(kx) + i\sin(kx) := \psi_k$ databookuw.com

$$\langle f(x), g(x) \rangle = \int_{-\pi}^{\pi} f(x)\,\bar{g}(x)\,dx$$

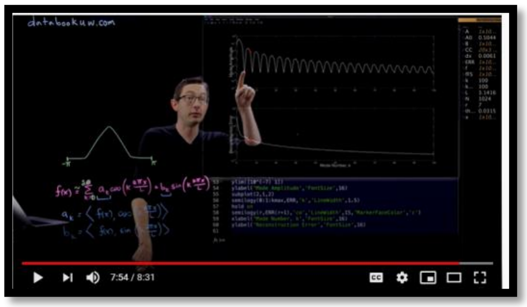
$$f(x) = \sum_{k=-\infty}^{\infty} c_k e^{ikx} = \sum_{k=-\infty}^{\infty} (\alpha_k + i\beta_k)(\cos(kx) + i\sin(kx))$$

$$(c_k = \bar{c}_{-k} \text{ if } f(x) \text{ real})$$

$$= \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \underbrace{\langle f(x), \psi_k \rangle}_{c_k} \underbrace{\psi_k}_{e^{ikx}}$$

$$\langle \psi_j, \psi_k \rangle = \int_{-\pi}^{\pi} e^{ijx} e^{-ikx} dx = \int_{-\pi}^{\pi} e^{i(j-k)x} dx = \frac{1}{i(j-k)} \left[e^{i(j-k)x} \right]_{-\pi}^{\pi} = \begin{cases} 0 & \text{if } j \neq k \\ 2\pi & \text{if } j = k \end{cases}$$

$$\vec{f} = \langle \vec{f}, \vec{x} \rangle \vec{x} + \langle \vec{f}, \vec{y} \rangle \vec{y}$$

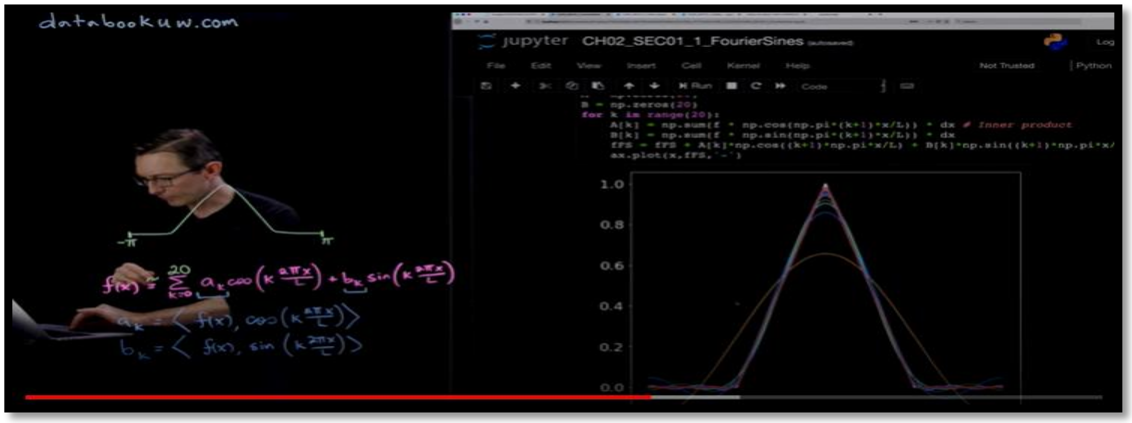


databookuw.com

$$f(x) \approx \sum_{k=0}^{10} a_k \cos\left(k\frac{2\pi x}{L}\right) + b_k \sin\left(k\frac{2\pi x}{L}\right)$$

$$a_k = \left\langle f(x), \cos\left(k\frac{2\pi x}{L}\right)\right\rangle$$

$$b_k = \left\langle f(x), \sin\left(k\frac{2\pi x}{L}\right)\right\rangle$$



databookuw.com

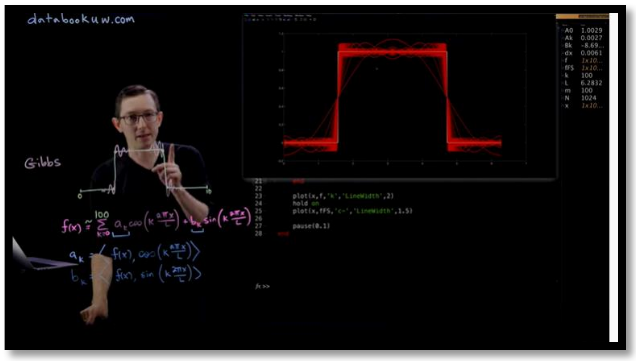
jupyter CH02_SEC01_1_FourierSines (autosaved)

```
B = np.zeros(20)
for k in range(20):
    A[k] = np.sum(f * np.cos(np.pi*(k+1)*x/L)) * dx # Inner product
    B[k] = np.sum(f * np.sin(np.pi*(k+1)*x/L)) * dx
    fFS = fFS + A[k]*np.cos((k+1)*np.pi*x/L) + B[k]*np.sin((k+1)*np.pi*x/
    ax.plot(x,fFS,'-')
```

$$f(x) \approx \sum_{k=0}^{20} a_k \cos\left(k\frac{2\pi x}{L}\right) + b_k \sin\left(k\frac{2\pi x}{L}\right)$$

$$a_k = \left\langle f(x), \cos\left(k\frac{2\pi x}{L}\right)\right\rangle$$

$$b_k = \left\langle f(x), \sin\left(k\frac{2\pi x}{L}\right)\right\rangle$$



databookuw.com

Gibbs

```
    end

    plot(x,f,'k','LineWidth',2)
    hold on
    plot(x,fFS,'c-','LineWidth',1.5)

    pause(0.1)
end
```

$$f(x) \approx \sum_{k=0}^{100} a_k \cos\left(k\frac{2\pi x}{L}\right) + b_k \sin\left(k\frac{2\pi x}{L}\right)$$

$$a_k = \left\langle f(x), \cos\left(k\frac{2\pi x}{L}\right)\right\rangle$$

$$b_k = \left\langle f(x), \sin\left(k\frac{2\pi x}{L}\right)\right\rangle$$

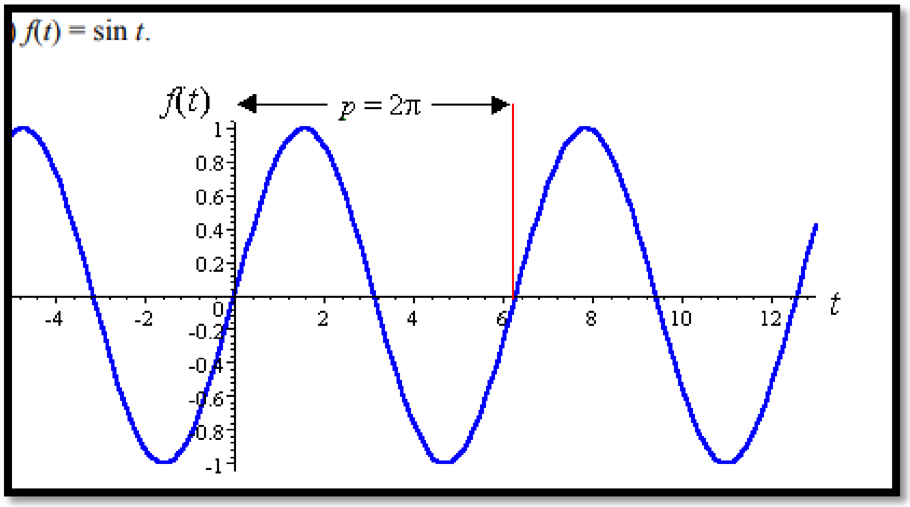
Report –

Fourier series are used in the analysis of periodic functions. In mathematics, infinite series are very important. They are used extensively in calculators and computers for evaluating values of many functions.

The Fourier series is really interesting, as it uses many of the mathematical techniques that you have learned before, like graphs, integration, differentiation, summation notation, trigonometry, etc.

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 Periodic Functions -



The **Fourier Transform** is a mathematical technique that transforms a function of tim*e, x (t)*, to a function of frequency, *X (ω)*. It is closely related to the Fourier Series.

Forward Fourier Transform: **Analysis Equation**

$$X(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-j\omega t}dt$$

Inverse Fourier Transform: **Synthesis Equation**

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} X(\omega)e^{j\omega t}d\omega$$

The **Hilbert transform** is a specific linear operator that takes a function, *u* (*t*) of a real variable and produces another function of a real variable *H* (*u*) (*t*). This linear operator is given by convolution with the function:

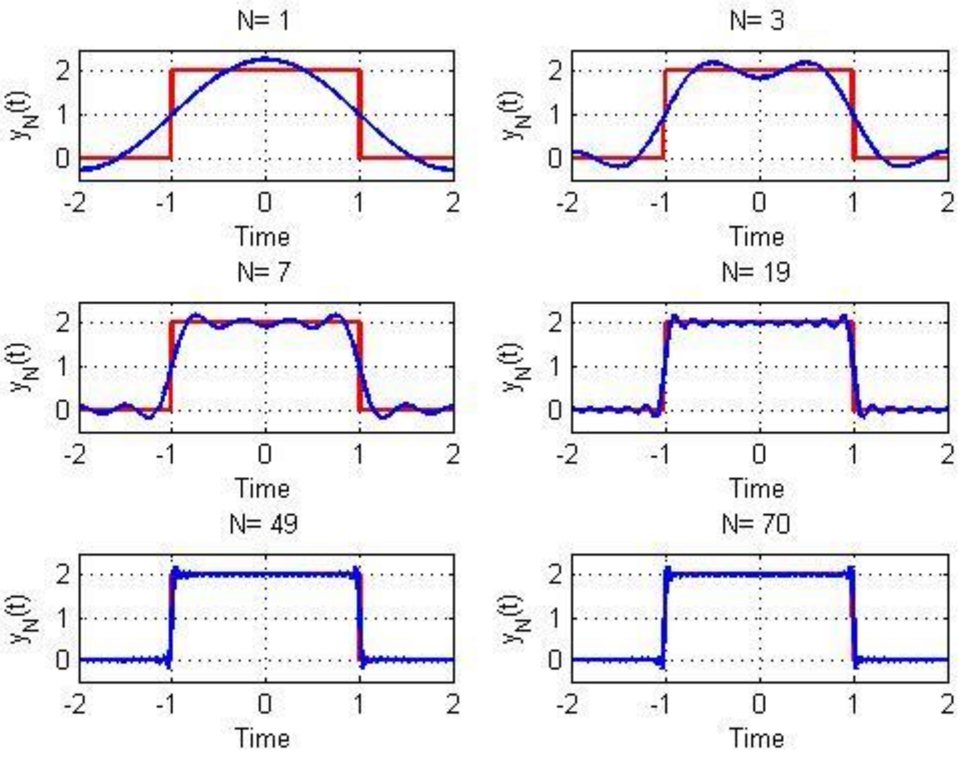
$$H(u)(t) = \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{u(\tau)}{t-\tau}\,d\tau,$$

The Hilbert transform is important in signal processing, where it derives the analytic representation of a real-valued signal *u* (*t*). Specifically, the Hilbert transform of *u* is its harmonic conjugate *v*, a function of the real variable *t* such that the complex-valued function *u* + i*v* admits an extension to the complex upper half-plane satisfying the Cauchy–Riemann equations.

The **Complex Fourier** series is presented first with period 2π, then with general period. The connection with the real-valued Fourier series is explained and formulae are given for converting between the two types of representation.

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{int}$$

Gibbs Phenomenon is used to convert the sine wave in to square wave by adding the number of harmonics to the sine wave using Fourier series.

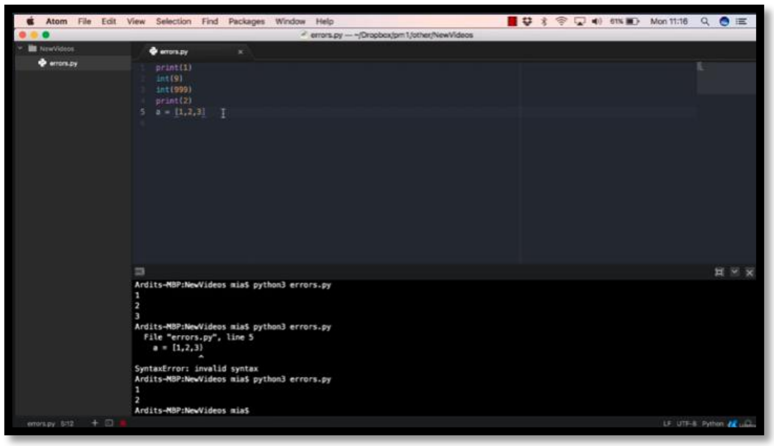


DAILY ASSESSMENT FORMAT

Date:	**25/05/2020**	**Name:**	**Davis S. Patel**
Course:	**Python Course**	**USN:**	**4AL16EC045**
Topic:	**Fixing Programming Errors Application 3: Build a Website Blocker**	**Semester & Section:**	**8th - A**
GitHub Repository:	**Davis**		

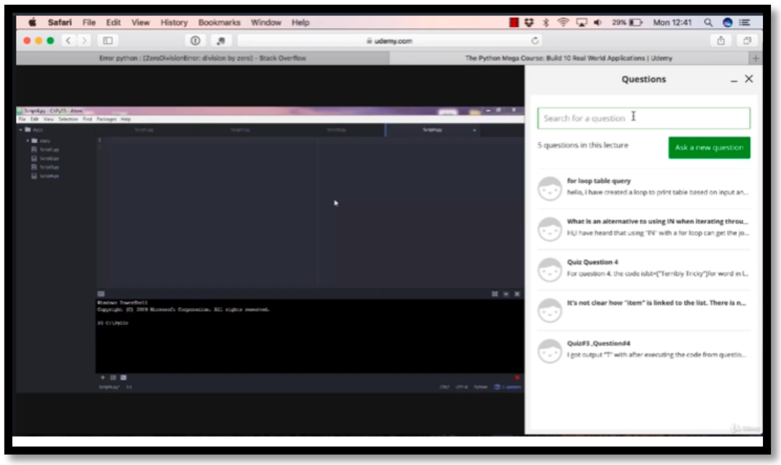
AFTERNOON SESSION DETAILS

Image of Session



```
print(1)
int(9)
int(999)
print(2)
a = [1,2,3]
```

```
Ardits-MBP:NewVideos mia$ python3 errors.py
1
2
3
Ardits-MBP:NewVideos mia$ python3 errors.py
  File "errors.py", line 5
    a = [1,2,3]
           ^
SyntaxError: invalid syntax
Ardits-MBP:NewVideos mia$ python3 errors.py
1
2
Ardits-MBP:NewVideos mia$
```



Questions

5 questions in this lecture

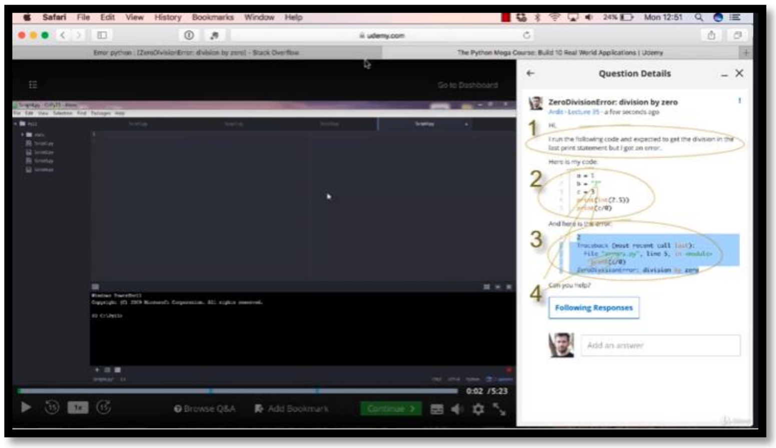
for loop table query
hello, I have created a loop to print table based on input an...

What is an alternative to using IN when iterating throu...
Hi,I have heard that using "IN" with a for loop can get the jo...

Quiz Question 4
For question 4, the code islst=["Terribly Tricky"]for word in L...

It's not clear how "item" is linked to the list. There is n...

Quiz#3 ,Question#4
I got output "T" with after executing the code from questio...


Safari File Edit View History Bookmarks Window Help
udemy.com
Error python : [ZeroDivisionError: division by zero] - Stack Overflow
The Python Mega Course: Build 10 Real World Applications | Udemy
Go to Dashboard
Question Details
Following Responses
Add an answer
0:02 / 5:23
Browse Q&A
Add Bookmark
Continue

ZeroDivisionError: division by zero

Ardit · Lecture 35 · a few seconds ago

1 Hi,

I run the following code and expected to get the division in the last print statement but I got an error.

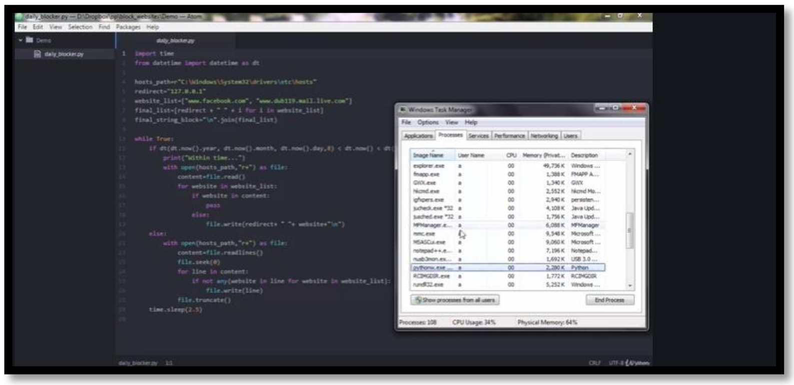
Here is my code:

2
```
a = 1
b = "2"
c = 3
print(int(7.5))
print(c/0)
```

And here is the error:

3
```
7
Traceback (most recent call last):
  File "errors.py", line 5, in <module>
    print(c/0)
ZeroDivisionError: division by zero
```

4 Can you help?


daily_blocker.py — D:\Dropbox\pp\block_websites\Demo — Atom
File Edit View Selection Find Packages Help
Demo
daily_blocker.py
Windows Task Manager
File Options View Help
Applications Processes Services Performance Networking Users
Show processes from all users
End Process
CRLF UTF-8 Python

```
import time
from datetime import datetime as dt

hosts_path=r"C:\Windows\System32\drivers\etc\hosts"
redirect="127.0.0.1"
website_list=["www.facebook.com", "www.dub119.mail.live.com"]
final_list=[redirect + " " + i for i in website_list]
final_string_block="\n".join(final_list)

while True:
    if dt(dt.now().year, dt.now().month, dt.now().day,8) < dt.now() < dt(
        print("Within time...")
        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_path,"r+") as file:
            content=file.readlines()
            file.seek(0)
            for line in content:
                if not any(website in line for website in website_list):
                    file.write(line)
            file.truncate()
    time.sleep(2.5)
```

Image Name	User Name	CPU	Memory (Privat...	Description
explorer.exe	a	00	49,736 K	Windows ...
fmapp.exe	a	00	1,388 K	FMAPP A...
GWX.exe	a	00	1,340 K	GWX
hkcmd.exe	a	00	2,552 K	hkcmd Mo...
igfxpers.exe	a	00	2,940 K	persisten...
jucheck.exe *32	a	00	4,108 K	Java Upd...
jusched.exe *32	a	00	1,756 K	Java Upd...
MPManager.e...	a	00	6,088 K	MPManager
mmc.exe	a	00	9,548 K	Microsoft ...
MSASCui.exe	a	00	9,060 K	Microsoft ...
notepad++.e...	a	00	7,196 K	Notepad...
nusb3mon.ex...	a	00	1,692 K	USB 3.0 ...
pythonw.exe ...	a	00	2,280 K	Python
RCIMGDIR.exe	a	00	1,772 K	RCIMGDIR
rundll32.exe	a	00	5,252 K	Windows ...

Processes: 108 CPU Usage: 34% Physical Memory: 64%

Report –

Fixing Programming Errors

Errors or mistakes in a program are often referred to as bugs. They are almost always the fault of the programmer. The process of finding and eliminating errors is called debugging. Errors can be classified into three major groups:

- Syntax errors
- Runtime errors
- Logical errors

Syntax errors

Python will find these kinds of errors when it tries to parse your program, and exit with an error message without running anything. Syntax errors are mistakes in the use of the Python language, and are analogous to spelling or grammar mistakes in a language like English: for example, the sentence *would you some tea?* Does not make sense – it is missing a verb.

Common Python syntax errors include:

- leaving out a keyword
- putting a keyword in the wrong place
- leaving out a symbol, such as a colon, comma or brackets
- misspelling a keyword
- incorrect indentation
- empty block

Runtime errors

If a program is syntactically correct – that is, free of syntax errors – it will be run by the Python interpreter. However, the program may exit unexpectedly during execution if it encounters a *runtime error* – a problem which was not detected when the program was parsed, but is only revealed when a particular line is executed. When a program comes to a halt because of a runtime error, we say that it has crashed.

Some examples of Python runtime errors:

- division by zero
- performing an operation on incompatible types
- using an identifier which has not been defined
- accessing a list element, dictionary value or object attribute which doesn't exist
- trying to access a file which doesn't exist

Logical errors

Logical errors are the most difficult to fix. They occur when the program runs without crashing, but produces an incorrect result. The error is caused by a mistake in the program's logic. You won't get an error message, because no syntax or runtime error has occurred.

Some examples of mistakes which lead to logical errors:

- using the wrong variable name
- indenting a block to the wrong level
- using integer division instead of floating-point division
- getting operator precedence wrong
- making a mistake in a Boolean expression
- off-by-one, and other numerical errors

In Python, exceptions can be handled using a `try` statement. The critical operation which can raise an exception is placed inside the `try` clause. The code that handles the exceptions is written in the `except` clause.

```
while True:
...     try:
...         x = int(input("Please enter a number: "))
...         break
...     except ValueError:
...         print("Oops!  That was no valid number.  Try again...")
...
```

Application 3: Build a Website Blocker

This application can be used to block the websites so that the user can not open them during the specific period. The objective of Python website blocker is to block some certain websites which can distract the user during the specified amount of time.

In this, we will block the access to the list of some particular websites during the working hours so that the user can only access those websites during the free time only.

The working time in this python application is considered from 9 AM to 5 PM. The time period except that time will be considered as free time.

Python modules to build the python website blocker.

1. **file handling:** file handling is used to do the modifications to the hosts file.
2. **time:** The time module is used to control the frequency of the modifications to the hosts file.
3. **datetime:** The datetime module is used to keep track of the free time and working time.

Code -

```
import time
from datetime import datetime as dt

hosts_temp=r"D:\Dropbox\pp\block_websites\Demo\hosts"
hosts_path="/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,8) < dt.now() <
dt(dt.now().year,dt.now().month,dt.now().day,16):
    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_path,'r+') as file:
      content=file.readlines()
      file.seek(0)
      for line in content:
        if not any(website in line for website in website_list):
          file.write(line)
      file.truncate()
    print("Fun hours...")
  time.sleep(5)
```

The python script that can run at system startup to block the access to the particular websites. Open PyCharm to edit the code or we can use any IDE you want.

Setting up the infinite loop

We need to have a while loop in the python script to make sure that our script will run at every 5 seconds.

```
import time

host_path = r"/etc/hosts"
redirect = "127.0.0.1"
websites = ["www.facebook.com", "https://www.facebook.com"]

while True:
   time.sleep(5)
```

The any () function

Python any () function accepts iterable (list, tuple, dictionary etc.) as an argument and return true if any of the element in iterable is true, else it returns false. If iterable is empty then any () method returns false.

Example:

```
>>>  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`:

1. ("face" in "trees are good")
2. ("clock" in "trees are good")
3. ("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 we 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`.