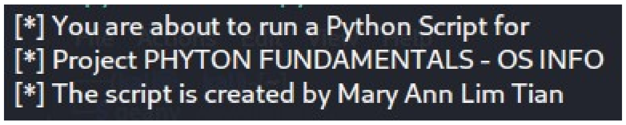
Project Python Fundamentals

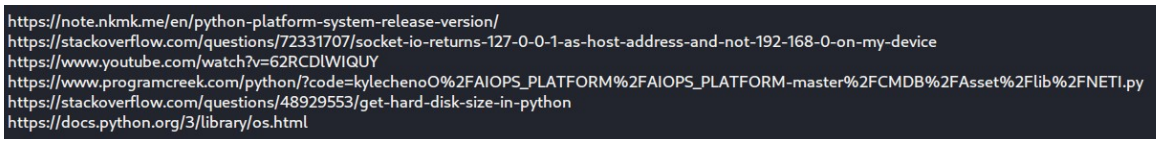
1. This project is to run using Python script to get the current operating system information.

2. Introduction message is displayed to inform the user of type of script and what it is for. The script is automated to get current system information



```
[*] You are about to run a Python Script for
[*] Project PHYTON FUNDAMENTALS - OS INFO
[*] The script is created by Mary Ann Lim Tian
```

3. Credits will be displayed - these are the reference sources to buid this script



```
https://note.nkmk.me/en/python-platform-system-release-version/
https://stackoverflow.com/questions/72331707/socket-io-returns-127-0-0-1-as-host-address-and-not-192-168-0-on-my-device
https://www.youtube.com/watch?v=62RCDlWIQUY
https://www.programcreek.com/python/?code=kylechenoO%2FAIOPS_PLATFORM%2FAIOPS_PLATFORM-master%2FCMDB%2FAsset%2Flib%2FNETI.py
https://stackoverflow.com/questions/48929553/get-hard-disk-size-in-python
https://docs.python.org/3/library/os.html
```

4. The following modules are "import" allows you to use code from other modules

a. os - is a Python built-in module that provides a way to interact with the underlying operating system

b. platform - is a Python module that provides a simple way to get information about the computer's operating system, hardware, and environment.

c. subprocess - is a Python module that allows you to spawn new processes, such as to run a system command like ifconfig and capture the output for use in your Python program

d. netifaces - is a Python module that provides a cross-platform interface to access network interfaces and addresses.

e. requests - is a module in Python that is used to make HTTP requests to URLs

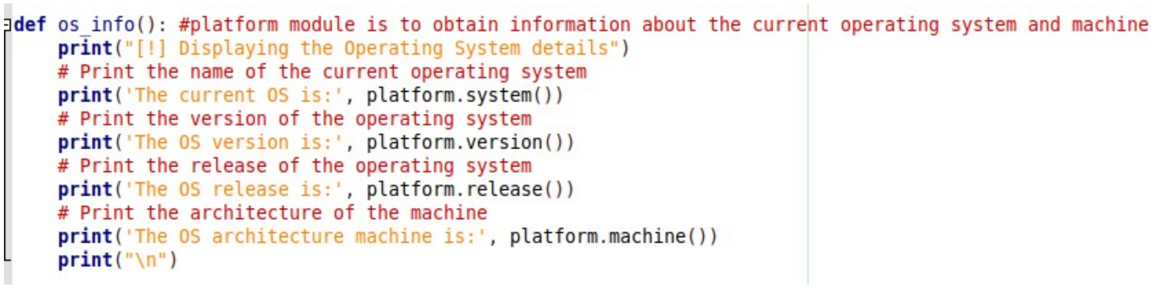
f. shutil - is a module in Python's standard library that provides a set of high-level operations for working with files and collections of files.

g. time - is a module that provides various time-related functions. It allows you to work with time and date values, measure elapsed time, and perform various other operations that require working with time.

5. There 8 functions created in the script that allow it to be reusable code that performs a specific task.

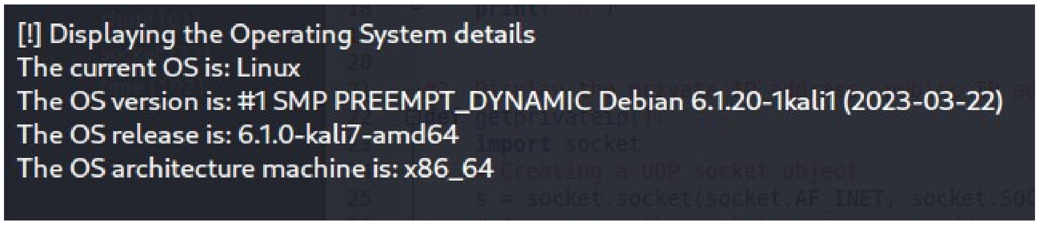
a**. credits_ref(**) - to display all the reference sources as shown in #3 b**. os_info()** - to display the current machine operating system, version, release and architecture

Using platform modules to get the OS information
platform.system() - to get the current machine OS
platform.version - to get the OS verison
platform.release - to get the OS release version
platform.archictecture - to get the bit architecture if it is 32bit or 64bit. In below sample output, the machine is 64bit.
code:



```
def os_info(): #platform module is to obtain information about the current operating system and machine
    print("[!] Displaying the Operating System details")
    # Print the name of the current operating system
    print('The current OS is:', platform.system())
    # Print the version of the operating system
    print('The OS version is:', platform.version())
    # Print the release of the operating system
    print('The OS release is:', platform.release())
    # Print the architecture of the machine
    print('The OS architecture machine is:', platform.machine())
    print("\n")
```

result:



```
[!] Displaying the Operating System details
The current OS is: Linux
The OS version is: #1 SMP PREEMPT_DYNAMIC Debian 6.1.20-1kali1 (2023-03-22)
The OS release is: 6.1.0-kali7-amd64
The OS architecture machine is: x86_64
```

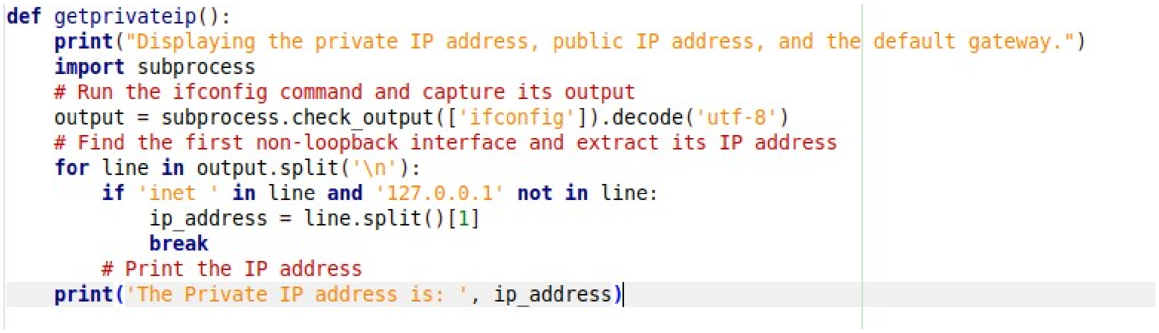
c**. getprivateip()** - to display the private IP address of the machine

Using subprocess module for this function. the for loop will split the ifconfig output split into individual lines using the newline character \n.

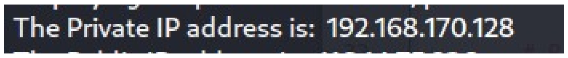
The if condition is whether the current line contains the string 'inet ' (which indicates an IP address), and also checks that the line does not contain '127.0.0.1' (which is the loopback address).

line.split() is used extracts the IP address from the line by splitting the line into words using whitespace as the separator

code:


```
def getprivateip():
    print("Displaying the private IP address, public IP address, and the default gateway.")
    import subprocess
    # Run the ifconfig command and capture its output
    output = subprocess.check_output(['ifconfig']).decode('utf-8')
    # Find the first non-loopback interface and extract its IP address
    for line in output.split('\n'):
        if 'inet ' in line and '127.0.0.1' not in line:
            ip_address = line.split()[1]
            break
        # Print the IP address
    print('The Private IP address is: ', ip_address)
```

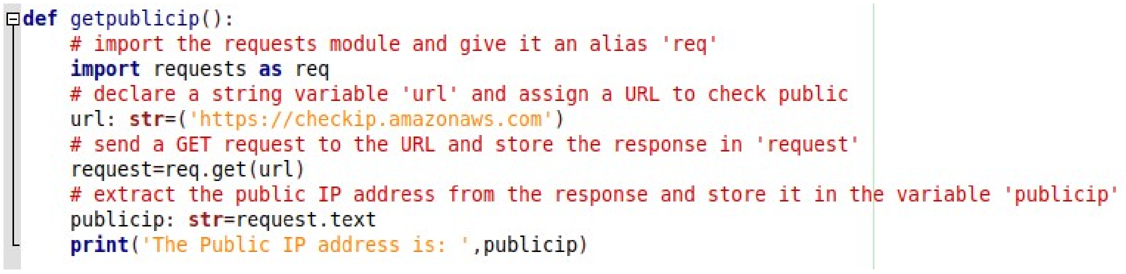
result:


```
The Private IP address is: 192.168.170.128
```

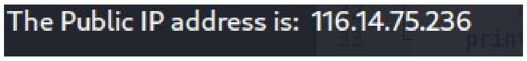
d. **getpublicip()** - to display the public IP address of the machine.

Using the requests module to get the public IP address from the website https://checkip.amazonaws.com using the GET request.

code:


```
def getpublicip():
    # import the requests module and give it an alias 'req'
    import requests as req
    # declare a string variable 'url' and assign a URL to check public
    url: str=('https://checkip.amazonaws.com')
    # send a GET request to the URL and store the response in 'request'
    request=req.get(url)
    # extract the public IP address from the response and store it in the variable 'publicip'
    publicip: str=request.text
    print('The Public IP address is: ',publicip)
```

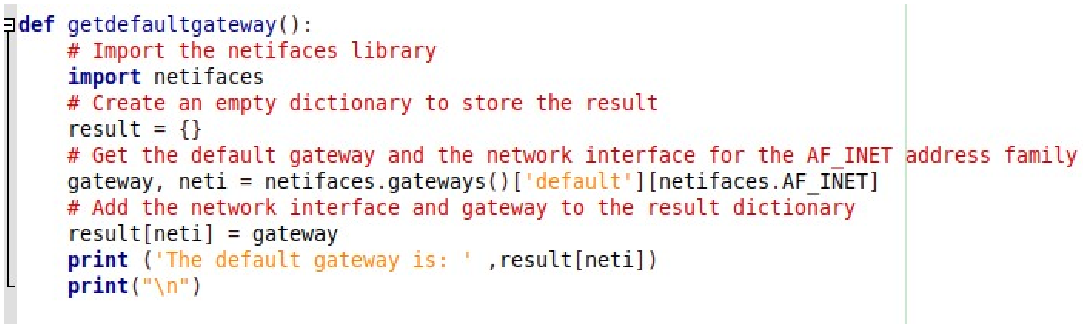
result:


```
The Public IP address is: 116.14.75.236
```

e. **getdefaultgateway()** - to display the gateway address of the mchine

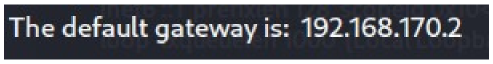
The function is using netifaces module

code:



```
def getdefaultgateway():
    # Import the netifaces library
    import netifaces
    # Create an empty dictionary to store the result
    result = {}
    # Get the default gateway and the network interface for the AF_INET address family
    gateway, neti = netifaces.gateways()['default'][netifaces.AF_INET]
    # Add the network interface and gateway to the result dictionary
    result[neti] = gateway
    print ('The default gateway is: ' ,result[neti])
    print("\n")
```

result:

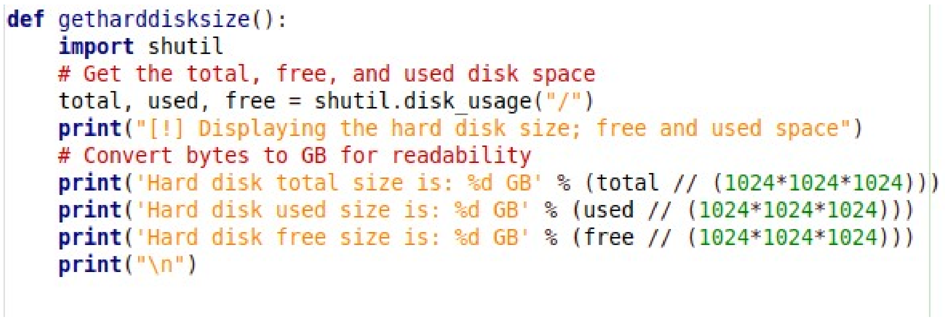


```
The default gateway is: 192.168.170.2
```

f. getharddisksize() - to display the total harddisk size of the machine including free size left and used size

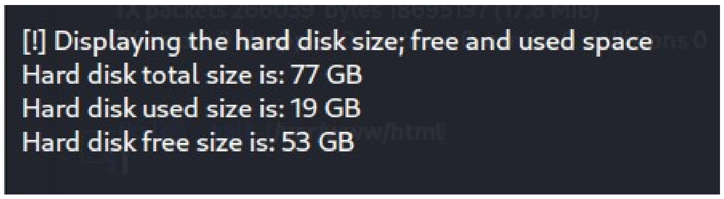
%(variable // 1024*1024*1024) The // operator performs integer division, which means that the result is rounded down to the nearest integer. The % operator then calculates the remainder of this division.

code:



```
def getharddisksize():
    import shutil
    # Get the total, free, and used disk space
    total, used, free = shutil.disk_usage("/")
    print("[!] Displaying the hard disk size; free and used space")
    # Convert bytes to GB for readability
    print('Hard disk total size is: %d GB' % (total // (1024*1024*1024)))
    print('Hard disk used size is: %d GB' % (used // (1024*1024*1024)))
    print('Hard disk free size is: %d GB' % (free // (1024*1024*1024)))
    print("\n")
```

result:



```
[!] Displaying the hard disk size; free and used space
Hard disk total size is: 77 GB
Hard disk used size is: 19 GB
Hard disk free size is: 53 GB
```

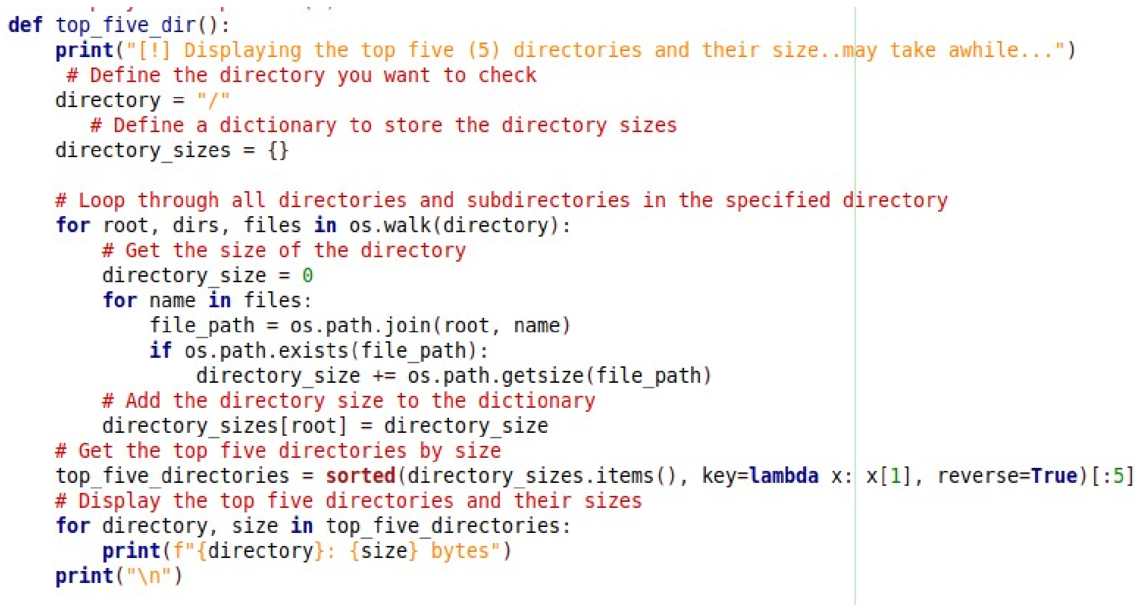
g. top_five_dir() - to check all the directories of the machine and get the top 5 directories and its size

a. for root, dirs, files in os.walk(directory): - This starts the iteration over each directory in the tree starting at directory. os.walk() returns a tuple containing the root directory, a list of

subdirectories, and a list of files in the current directory.
b. directory_size = 0 - This initializes the size of the current directory to 0.
for name in files: - This loops over each file in the current directory.
c. file_path = os.path.join(root, name) - This joins the root directory with the file name to create a complete path to the file.
d. if os.path.exists(file_path): - This checks if the file exists at the specified path.
e. directory_size += os.path.getsize(file_path) - This gets the size of the file at the specified path and adds it to the directory_size variable.
f. directory_sizes[root] = directory_size - This adds the directory_size to a dictionary called directory_sizes, with the directory name as the key.

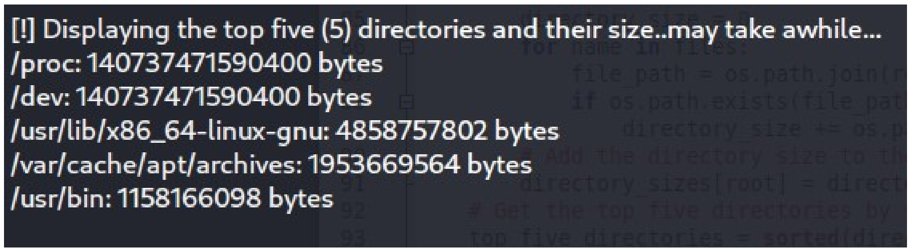
At the end of this code block, the directory_sizes dictionary will contain the size of each directory in the directory tree.

code:



```
def top_five_dir():
    print("[!] Displaying the top five (5) directories and their size..may take awhile...")
     # Define the directory you want to check
    directory = "/"
       # Define a dictionary to store the directory sizes
    directory_sizes = {}

    # Loop through all directories and subdirectories in the specified directory
    for root, dirs, files in os.walk(directory):
        # Get the size of the directory
        directory_size = 0
        for name in files:
            file_path = os.path.join(root, name)
            if os.path.exists(file_path):
                directory_size += os.path.getsize(file_path)
        # Add the directory size to the dictionary
        directory_sizes[root] = directory_size
    # Get the top five directories by size
    top_five_directories = sorted(directory_sizes.items(), key=lambda x: x[1], reverse=True)[:5]
    # Display the top five directories and their sizes
    for directory, size in top_five_directories:
        print(f"{directory}: {size} bytes")
    print("\n")
```

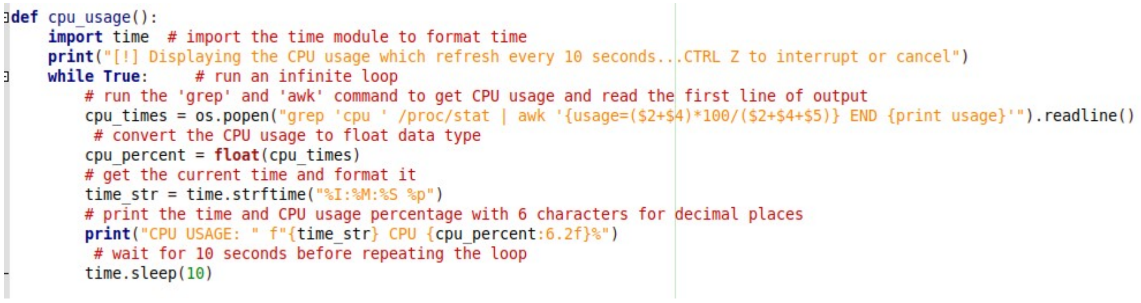
result:



```
[!] Displaying the top five (5) directories and their size..may take awhile...
/proc: 140737471590400 bytes
/dev: 140737471590400 bytes
/usr/lib/x86_64-linux-gnu: 4858757802 bytes
/var/cache/apt/archives: 1953669564 bytes
/usr/bin: 1158166098 bytes
```

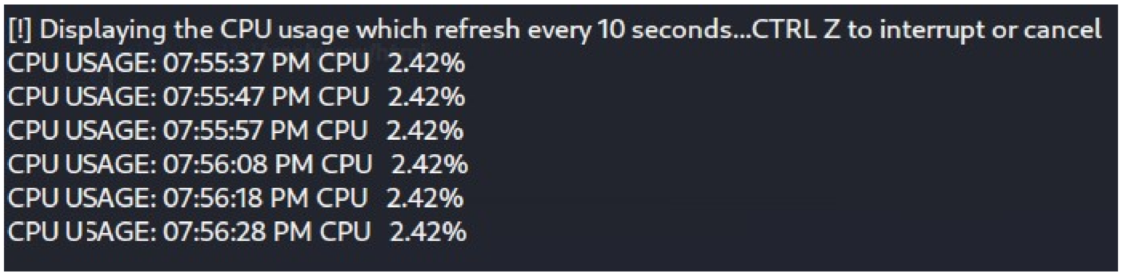
g. cpu_usage() - to display the cpu usage with interval of 10 seconds this function uses time module to get the timestamp and show the 10 seconds interval

code:



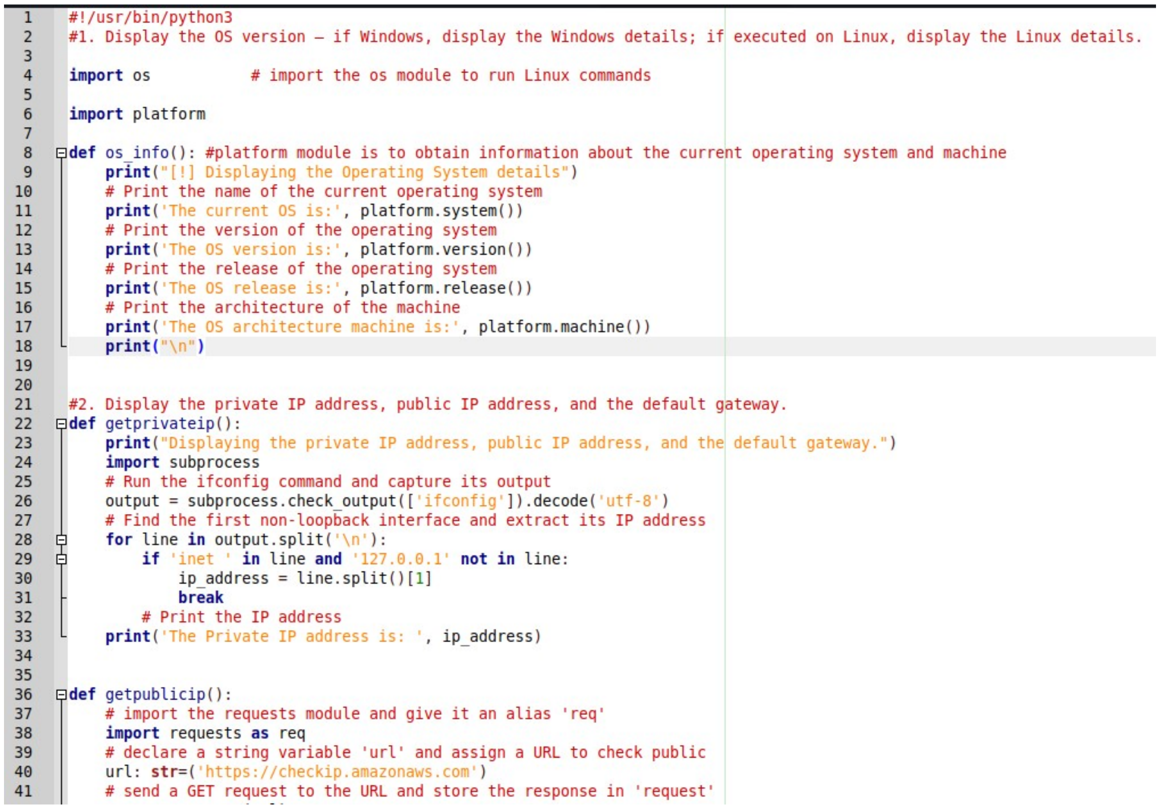
```
def cpu_usage():
    import time  # import the time module to format time
    print("[!] Displaying the CPU usage which refresh every 10 seconds...CTRL Z to interrupt or cancel")
    while True:     # run an infinite loop
        # run the 'grep' and 'awk' command to get CPU usage and read the first line of output
        cpu_times = os.popen("grep 'cpu ' /proc/stat | awk '{usage=($2+$4)*100/($2+$4+$5)} END {print usage}'").readline()
         # convert the CPU usage to float data type
        cpu_percent = float(cpu_times)
        # get the current time and format it
        time_str = time.strftime("%I:%M:%S %p")
        # print the time and CPU usage percentage with 6 characters for decimal places
        print("CPU USAGE: " f"{time_str} CPU {cpu_percent:6.2f}%")
         # wait for 10 seconds before repeating the loop
        time.sleep(10)
```

result:

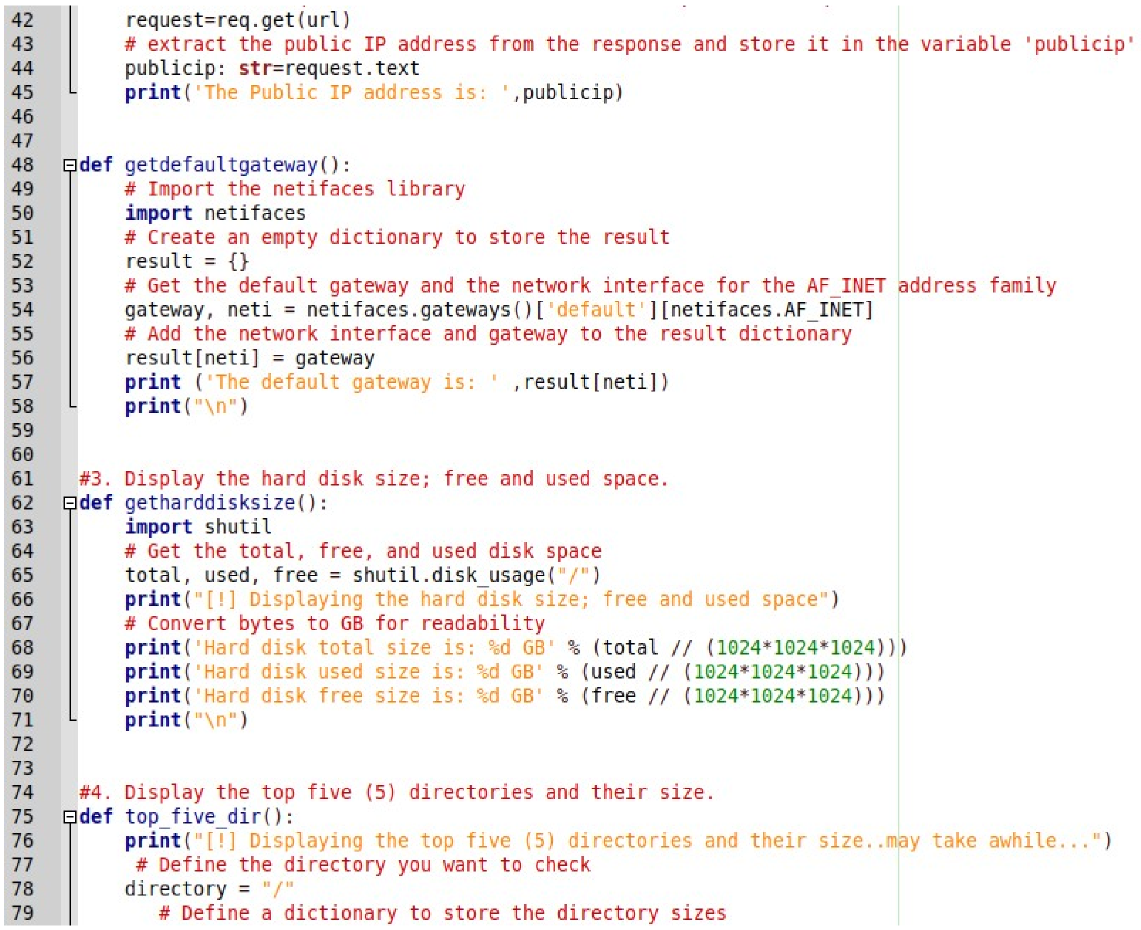


```
[!] Displaying the CPU usage which refresh every 10 seconds...CTRL Z to interrupt or cancel
CPU USAGE: 07:55:37 PM CPU   2.42%
CPU USAGE: 07:55:47 PM CPU   2.42%
CPU USAGE: 07:55:57 PM CPU   2.42%
CPU USAGE: 07:56:08 PM CPU   2.42%
CPU USAGE: 07:56:18 PM CPU   2.42%
CPU USAGE: 07:56:28 PM CPU   2.42%
```

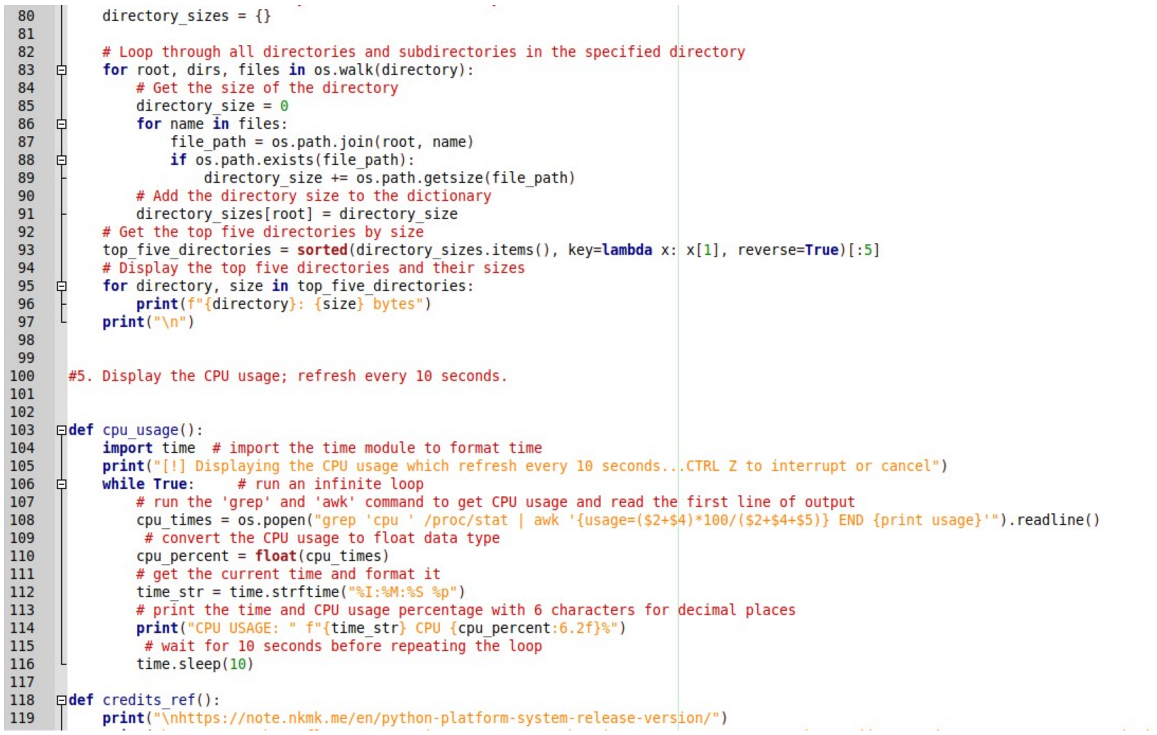
full script:



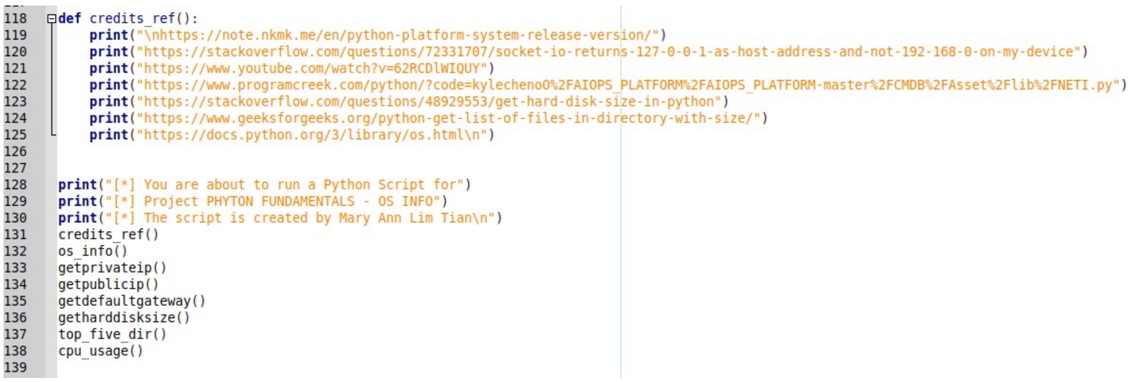
```python
#!/usr/bin/python3
#1. Display the OS version – if Windows, display the Windows details; if executed on Linux, display the Linux details.

import os           # import the os module to run Linux commands

import platform

def os_info(): #platform module is to obtain information about the current operating system and machine
    print("[!] Displaying the Operating System details")
    # Print the name of the current operating system
    print('The current OS is:', platform.system())
    # Print the version of the operating system
    print('The OS version is:', platform.version())
    # Print the release of the operating system
    print('The OS release is:', platform.release())
    # Print the architecture of the machine
    print('The OS architecture machine is:', platform.machine())
    print("\n")


#2. Display the private IP address, public IP address, and the default gateway.
def getprivateip():
    print("Displaying the private IP address, public IP address, and the default gateway.")
    import subprocess
    # Run the ifconfig command and capture its output
    output = subprocess.check_output(['ifconfig']).decode('utf-8')
    # Find the first non-loopback interface and extract its IP address
    for line in output.split('\n'):
        if 'inet ' in line and '127.0.0.1' not in line:
            ip_address = line.split()[1]
            break
        # Print the IP address
    print('The Private IP address is: ', ip_address)


def getpublicip():
    # import the requests module and give it an alias 'req'
    import requests as req
    # declare a string variable 'url' and assign a URL to check public
    url: str=('https://checkip.amazonaws.com')
    # send a GET request to the URL and store the response in 'request'
```



```python
    request=req.get(url)
    # extract the public IP address from the response and store it in the variable 'publicip'
    publicip: str=request.text
    print('The Public IP address is: ',publicip)


def getdefaultgateway():
    # Import the netifaces library
    import netifaces
    # Create an empty dictionary to store the result
    result = {}
    # Get the default gateway and the network interface for the AF_INET address family
    gateway, neti = netifaces.gateways()['default'][netifaces.AF_INET]
    # Add the network interface and gateway to the result dictionary
    result[neti] = gateway
    print ('The default gateway is: ' ,result[neti])
    print("\n")


#3. Display the hard disk size; free and used space.
def getharddisksize():
    import shutil
    # Get the total, free, and used disk space
    total, used, free = shutil.disk_usage("/")
    print("[!] Displaying the hard disk size; free and used space")
    # Convert bytes to GB for readability
    print('Hard disk total size is: %d GB' % (total // (1024*1024*1024)))
    print('Hard disk used size is: %d GB' % (used // (1024*1024*1024)))
    print('Hard disk free size is: %d GB' % (free // (1024*1024*1024)))
    print("\n")


#4. Display the top five (5) directories and their size.
def top_five_dir():
    print("[!] Displaying the top five (5) directories and their size..may take awhile...")
     # Define the directory you want to check
    directory = "/"
       # Define a dictionary to store the directory sizes
```

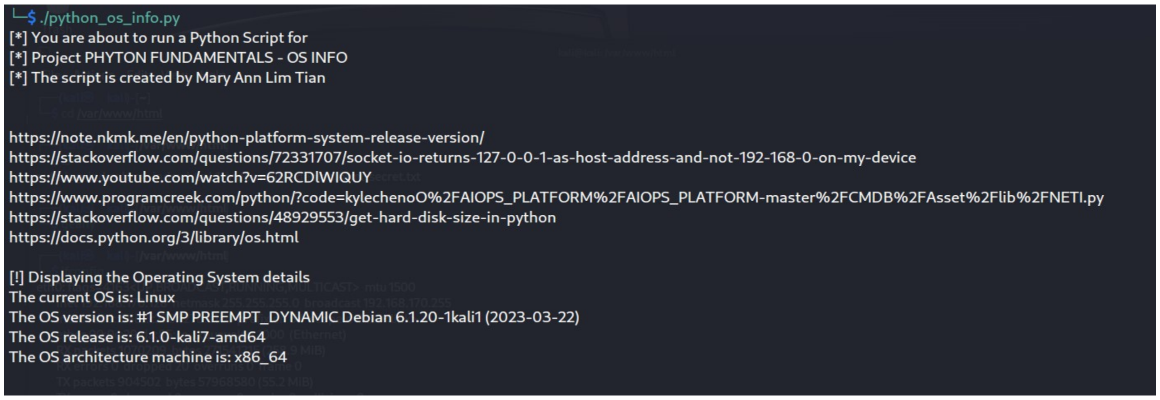


```python
    directory_sizes = {}

    # Loop through all directories and subdirectories in the specified directory
    for root, dirs, files in os.walk(directory):
        # Get the size of the directory
        directory_size = 0
        for name in files:
            file_path = os.path.join(root, name)
            if os.path.exists(file_path):
                directory_size += os.path.getsize(file_path)
        # Add the directory size to the dictionary
        directory_sizes[root] = directory_size
    # Get the top five directories by size
    top_five_directories = sorted(directory_sizes.items(), key=lambda x: x[1], reverse=True)[:5]
    # Display the top five directories and their sizes
    for directory, size in top_five_directories:
        print(f"{directory}: {size} bytes")
    print("\n")


#5. Display the CPU usage; refresh every 10 seconds.


def cpu_usage():
    import time  # import the time module to format time
    print("[!] Displaying the CPU usage which refresh every 10 seconds...CTRL Z to interrupt or cancel")
    while True:     # run an infinite loop
        # run the 'grep' and 'awk' command to get CPU usage and read the first line of output
        cpu_times = os.popen("grep 'cpu ' /proc/stat | awk '{usage=($2+$4)*100/($2+$4+$5)} END {print usage}'").readline()
         # convert the CPU usage to float data type
        cpu_percent = float(cpu_times)
        # get the current time and format it
        time_str = time.strftime("%I:%M:%S %p")
        # print the time and CPU usage percentage with 6 characters for decimal places
        print("CPU USAGE: " f"{time_str} CPU {cpu_percent:6.2f}%")
         # wait for 10 seconds before repeating the loop
        time.sleep(10)

def credits_ref():
    print("\nhttps://note.nkmk.me/en/python-platform-system-release-version/")
```

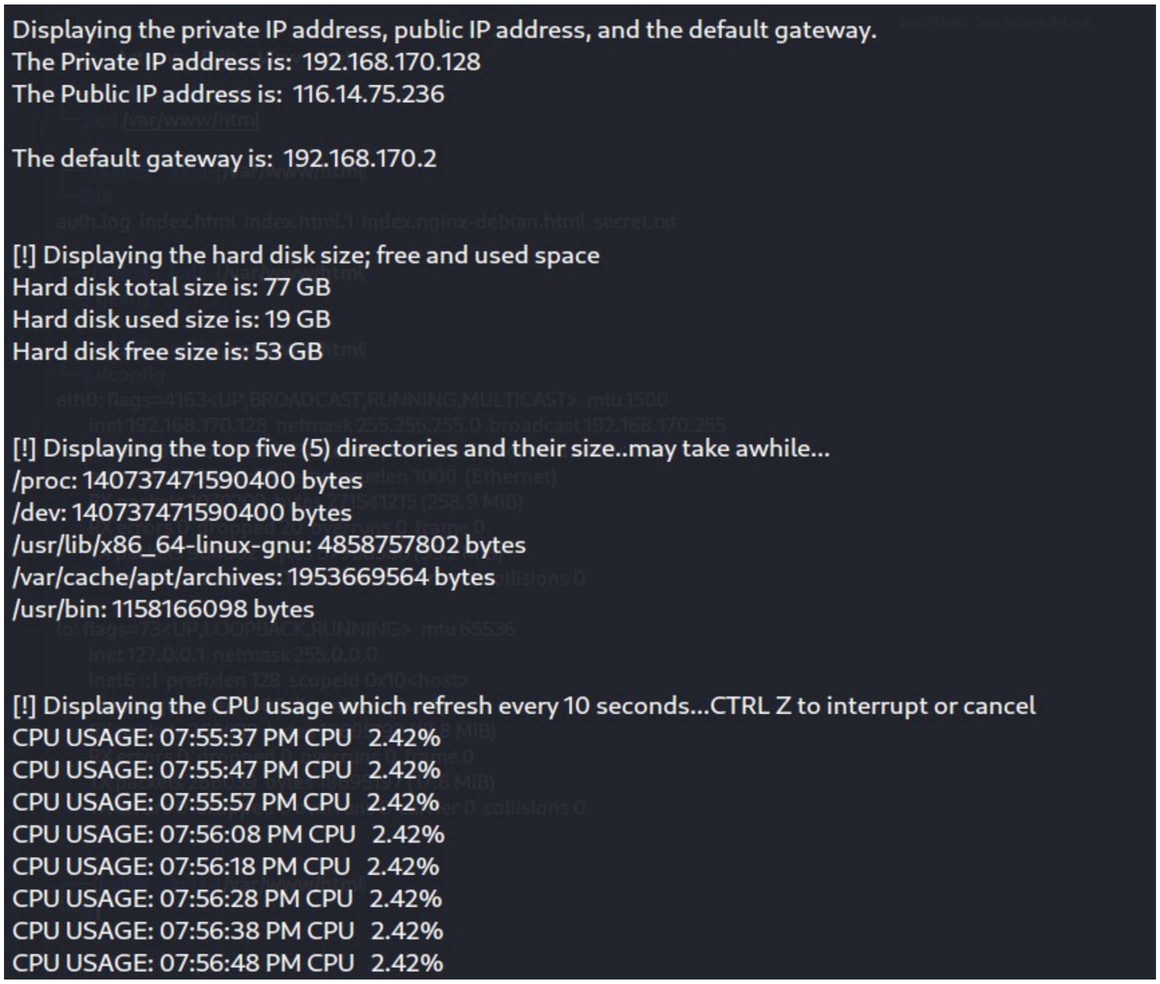


```python
def credits_ref():
    print("\nhttps://note.nkmk.me/en/python-platform-system-release-version/")
    print("https://stackoverflow.com/questions/72331707/socket-io-returns-127-0-0-1-as-host-address-and-not-192-168-0-on-my-device")
    print("https://www.youtube.com/watch?v=62RCDlWIQUY")
    print("https://www.programcreek.com/python/?code=kylecheno0%2FAIOPS_PLATFORM%2FAIOPS_PLATFORM-master%2FCMDB%2FAsset%2Flib%2FNETI.py")
    print("https://stackoverflow.com/questions/48929553/get-hard-disk-size-in-python")
    print("https://www.geeksforgeeks.org/python-get-list-of-files-in-directory-with-size/")
    print("https://docs.python.org/3/library/os.html\n")


print("[*] You are about to run a Python Script for")
print("[*] Project PHYTON FUNDAMENTALS - OS INFO")
print("[*] The script is created by Mary Ann Lim Tian\n")
credits_ref()
os_info()
getprivateip()
getpublicip()
getdefaultgateway()
getharddisksize()
top_five_dir()
cpu_usage()
```

Output:



```
└─$ ./python_os_info.py
[*] You are about to run a Python Script for
[*] Project PHYTON FUNDAMENTALS - OS INFO
[*] The script is created by Mary Ann Lim Tian


https://note.nkmk.me/en/python-platform-system-release-version/
https://stackoverflow.com/questions/72331707/socket-io-returns-127-0-0-1-as-host-address-and-not-192-168-0-on-my-device
https://www.youtube.com/watch?v=62RCDlWIQUY
https://www.programcreek.com/python/?code=kylechenoO%2FAIOPS_PLATFORM%2FAIOPS_PLATFORM-master%2FCMDB%2FAsset%2Flib%2FNETI.py
https://stackoverflow.com/questions/48929553/get-hard-disk-size-in-python
https://docs.python.org/3/library/os.html

[!] Displaying the Operating System details
The current OS is: Linux
The OS version is: #1 SMP PREEMPT_DYNAMIC Debian 6.1.20-1kali1 (2023-03-22)
The OS release is: 6.1.0-kali7-amd64
The OS architecture machine is: x86_64
```



```
Displaying the private IP address, public IP address, and the default gateway.
The Private IP address is:  192.168.170.128
The Public IP address is:  116.14.75.236

The default gateway is:  192.168.170.2


[!] Displaying the hard disk size; free and used space
Hard disk total size is: 77 GB
Hard disk used size is: 19 GB
Hard disk free size is: 53 GB


[!] Displaying the top five (5) directories and their size..may take awhile...
/proc: 140737471590400 bytes
/dev: 140737471590400 bytes
/usr/lib/x86_64-linux-gnu: 4858757802 bytes
/var/cache/apt/archives: 1953669564 bytes
/usr/bin: 1158166098 bytes


[!] Displaying the CPU usage which refresh every 10 seconds...CTRL Z to interrupt or cancel
CPU USAGE: 07:55:37 PM CPU  2.42%
CPU USAGE: 07:55:47 PM CPU  2.42%
CPU USAGE: 07:55:57 PM CPU  2.42%
CPU USAGE: 07:56:08 PM CPU  2.42%
CPU USAGE: 07:56:18 PM CPU  2.42%
CPU USAGE: 07:56:28 PM CPU  2.42%
CPU USAGE: 07:56:38 PM CPU  2.42%
CPU USAGE: 07:56:48 PM CPU  2.42%
```