VLSM DESIGNER

Technical and User Documentation

Author: Mario Pereira

**Contents**

[**Introduction** 1](#_Toc10195724)

[**Terminology, Abbreviations and Acronyms** 1](#_Toc10195725)

[**Executive Summary** 1](#_Toc10195726)

[**Technical Requirements** 1](#_Toc10195727)

[**Design and Code** 1](#_Toc10195728)

[**User Documentation** 1](#_Toc10195729)

[**Conclusions** 1](#_Toc10195730)

[**References** 1](#_Toc10195731)

[**Appendices** 1](#_Toc10195732)

## **Introduction**

This script uses the ipaddress module in Python to design a maximum of 6 subnets, the script will output the primary details for these subnets based only on script user’s hosts requirements, these subnets will follow a sequential hierarchy so long as user inputs requirements from largest to smallest.

## **Terminology, Abbreviations and Acronyms**

1. VLSM

* A logical process/technique to allow networks to divide IP addressing space into different size networks.

1. Subnet

* A logically allocated portion of a network address.

1. Pseudocode

* Not real working code, this code serves the purpose of design communication between humans, by using appropriate English for replacing coding syntax.

1. IDE

* Stands for Integrate Development Environment, this is a software application that facilitates coding by providing the coder with source code editors, build automation, debuggers and other relevant coding features.

1. Docstring

* Simply put this is a multi-line string that is not assigned to any code but is specified beside coding modules or definitions to explain the process within said modules/definitions running inside the script file.

1. Readme

* This is a txt file included within the folder that holds the script. Readme files usually provide a summary of the program and its purpose, they also include special requirements, changes between versions and links to other documentation.

## **Executive Summary**

This Document serves as a guide for understanding the purpose, design and code of the current script version 0.1 of OS Guesser as well as understanding its functionality, how to run it, operational requirements and current limitations. The technical requirements detail dependencies and best practice for running the script and the User Documentation is provided in the form of video for understanding how to use it.

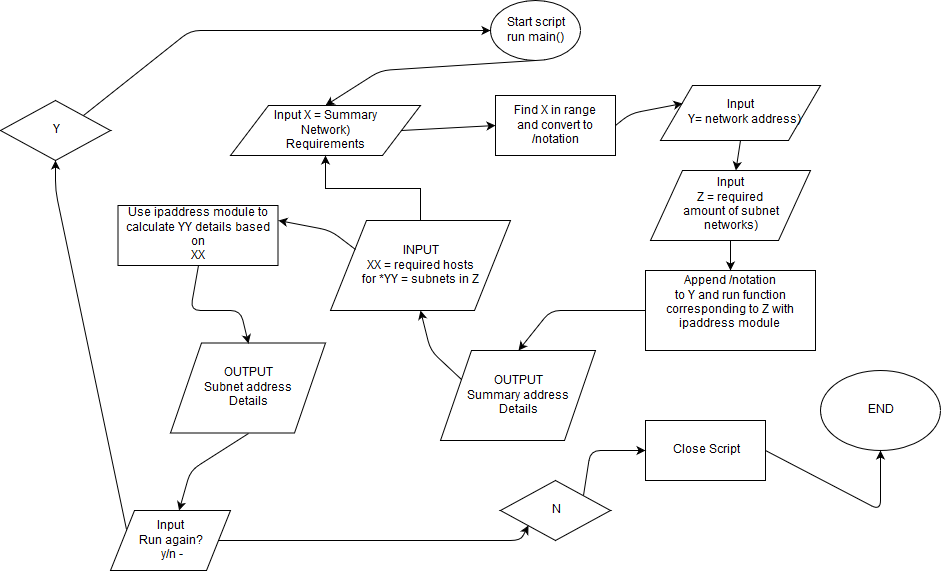
## **Technical Requirements**

1. Python 3+ Interpreter

* Available for download from <https://www.python.org/downloads/release/python-373/>
* Python is an interpreted scripting language that is object-oriented, to run a file with the .py extension python is necessary.

## **Design and Code**

1. Flow Chart



The Flowchart above lays out a simple format of the processes involved, and the path intended for the user to take.

1. Pseudocode

#The below function will show how to calculate for multiple subnets in order, one after another.

defining subnet designer(inputanIP):

print"\n\nThese are your Summary Address details"

$with IPaddress module run following functions$

ip = $ipaddress.IPv4Network(inputanIP)

vone ="The Network ID is" $ip.networkaddress and subnetmask

vtwo = "The First IP assignable for an host is" $ip.networkaddress + 1

vthree = "The Last IP assignable for an host is" $ip.broadcastaddress - 1

vfour = "The Broadcast address is" $ip.broadcastaddress

vseven = $network address

vsix = "The number of assignable IP is "$ip.number of addresses - 2

print (vone)

print (vtwo)

print (vthree)

print (vfour)

print (vsix)

newnetwork = (vseven) + 1

x = user input to question "Please input required number of hosts for your First subnet - "

ab = range between 1 4

bb = range between 1 - 8

cb = range between 1 - 16

db = range between 1 - 32

eb = range between 1 - 64

fb = range between 1 - 128

gb = range between 1 - 256

hb = range between 1 - 512

ib = range between 1 - 1024

jb = range between 1 - 2048

kb = range between 1 - 4086

lb = range between 1 - 8162

#and so on until /0

if x in ab:

bb = "/30"

else if x in bb:

bb = "/29"

else if x in cb:

bb = "/28"

else if x in db:

bb = "/27"

else if x in eb:

bb = "/26"

else if x in fb:

bb = "/25"

else if x in gb:

bb = "/24"

else if x in hb:

bb = "/23"

else if x in ib:

bb = "/22"

else if x in jb:

bb = "/21"

else if x in kb:

bb = "/20"

else if x in lb:

bb = "/19"

#and so on until /0

else:

print "I can't calculate bigger than /19 aka 8160 addresses. This is a design limitation."

theAdress = newnetwork + bb

print "Details for First subnet" theAdress

print "The Network ID is " $ip.networkaddress and subnetmask

print ("The First IP assignable for an host is " $ip.networkaddress + 1

print ("The Last IP assignable for an host is " $ip.broadcastaddress - 1

print ("The Broadcast address is " $ip.broadcastaddress

print("The number of assignable IP is "$ip.number of addresses - 2

\*\*\*\*\*\*

NewValue = ip.broadcast\_address

theAddress2 = NewValue + 2

print "Your Second subnet network will begin with an IP of " + (theAddress2)

x = user input to question "Please input required number of hosts for your First subnet - "

ab = range between 1 4

bb = range between 1 - 8

cb = range between 1 - 16

db = range between 1 - 32

eb = range between 1 - 64

fb = range between 1 - 128

gb = range between 1 - 256

hb = range between 1 - 512

ib = range between 1 - 1024

jb = range between 1 - 2048

kb = range between 1 - 4086

lb = range between 1 - 8162

#and so on until /0

if x in ab:

bbb = "/30"

else if x in bb:

bbb = "/29"

else if x in cb:

bbb = "/28"

else if x in db:

bbb = "/27"

else if x in eb:

bbb = "/26"

else if x in fb:

bbb = "/25"

else if x in gb:

bbb = "/24"

else if x in hb:

bbb = "/23"

else if x in ib:

bbb = "/22"

else if x in jb:

bbb = "/21"

else if x in kb:

bbb = "/20"

else if x in lb:

bbb = "/19"

#and so on until /0

else:

print "I can't calculate bigger than /19 aka 8160 addresses. This is a design limitation."

ThenewNetwork = theAddress2 + str(bbb)

print "Details for Second subnet"

print "The Network ID is " $ip.networkaddress and subnetmask

print ("The First IP assignable for an host is " $ip.networkaddress + 1

print ("The Last IP assignable for an host is " $ip.broadcastaddress - 1

print ("The Broadcast address is " $ip.broadcastaddress

print("The number of assignable IP is "$ip.number of addresses - 2

\*\*\*\*\*\*

Between the asterisks \*\*\*modify numerical values and references accordingly\*\*\* REPEAT THIS CODING LOGIC for each subnet required - I did it 6 times

NewNewValue = ip.broadcast\_address

theAddress3 = NewValue + 2

#ETC\_et cetera\_ETC\_et al

#ETC\_et al\_ETC\_et cetera

--------------------------------\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_--------------------------------\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-------------------------------\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-------------------------------

Defining main menu function

x = user input to question "Please input required number of hosts for your SUMMARY ADDRESS subnet - "

ab = range between 1 - 4

bb = range between 1 - 8

cb = range between 1 - 16

db = range between 1 - 32

eb = range between 1 - 64

fb = range between 1 - 128

gb = range between 1 - 256

hb = range between 1 - 512

ib = range between 1 - 1024

jb = range between 1 - 2048

kb = range between 1 - 4086

lb = range between 1 - 8162

#and so on until /0

if x in ab:

bb = "/30"

else if x in bb:

bb = "/29"

else if x in cb:

bb = "/28"

else if x in db:

bb = "/27"

else if x in eb:

bb = "/26"

else if x in fb:

bb = "/25"

else if x in gb:

bb = "/24"

else if x in hb:

bb = "/23"

else if x in ib:

bb = "/22"

else if x in jb:

bb = "/21"

else if x in kb:

bb = "/20"

else if x in lb:

bb = "/19"

#and so on until /0

else:

print "Something something something ur input is invalid dude."

ip = ask user for input("What is the network address you wish to use?)

x = (ip + bb)

maxIs6 = ask user for input("How many subnets would you like to calculate/design? (Max is 6) - ")

while True:

if maxIs6 == "6":

subnet designer function for 6(x)

run repeats function

else if maxIs6 == "5":

subnet designer function for 5(x)

run repeats function

else if maxIs6 == "4":

subnet designer function for 4(x)

repeats

else if maxIs6 == "3":

subnet designer function for 3(x)

run repeats function

else if maxIs6 == "2":

subnet designer function for 2(x)

run repeats function

else if maxIs6 == "1":

subnet designer function for 1(x)

run repeats function

else:

print "invalid input"

break

defining repeats function:

answer = ask for user input("Run Again? y/n - ").turn the output all lowercase

if answer == "":

print "you did not type anything"

run repeats function

else if answer == "y":

run main function

break

else:

close script

run main function

1. IDE

* Jetbrain’s Pycharm Community Edition 2018.3.4

1. Readme file

The following was extracted from the README.txt in the OS Guesser Script Folder.

## 

## VLSM DESIGNER

## This script uses the ipaddress module in Python to design a maximum of 6 subnets, the script will output the

## primary details for these subnets based only on script user’s hosts requirements, these subnets will follow

## a sequential hierarchy so long as user inputs requirements from largest to smallest.

## PURPOSE

## This is intended for when its necessary to spontaneously calculate Subnet Addressing Schemes.

## REQUIREMENTS

## -Some basic IPv4 knowledge

## -Basic VLSM process knowledge

## -Must input required subnet hosts from Largest to Smallest for accurate results and proper hierarchical output.

## GUIDE

## -Run the script read the instructions before input.

## -Only number keys and the space bar are necessary for script input.

## -Script has very basic input validation in it so dont press any letters or the space bar while running.

## **User Documentation**

* This is intended for those with a basic understanding in the VLSM process.
* To run this script simply follow all prompts exactly, while inputting required number of hosts ensure to begin with the largest subnet first.

1. The official documentation for this script is in the form of a video with the full utility of the OS Guesser in action <https://youtu.be/B4TzRxVQRdE> <--If this link does not work I have provided the original video in the Folder.

## **Conclusions**

The scripting program OS Guesser documented here provides the user with the ability to perform guesses from within any IPv4 network. The script does require PC configuration in the form of disabled firewalls/network security to enable accurate guessing. The Documentation including pseudo coding, flow charts, design and code are all available from the GITHUB repository <https://github.com/Monsid/OS-Guesser> with a Video User Guide available on [Youtube](https://youtu.be/B4TzRxVQRdE) .

## **References**

1. Software used to Design and Produce Script

* Creating Flowcharts - www.draw.io
* Writing Pseudocode – Notepad
* IDE for writing Code – Jetbrain’s Pycharm Community Edition 2018.3.4

1. Software used to test script.

* Python 3.7 from console.
* Jetbrain’s Pycharm in debug mode.
* Google forms – peer review testing.

1. Software used to produce Documentation.

* Microsoft Word
* Notepad
* Radeon ReLive – video capture
* Microsoft Photos – video editing

## **Appendices**

1. Testing - Peer Review

* <https://forms.gle/ncoEfUycu6knXAeJ7> currently ongoing, feel free to include a response.

1. Distribution – Github

* <https://github.com/Monsid/OS-Guesser>