

MYANMAR VERSION

PYTHON NETWORK PROGRAMMING

BY KAHNT PHYO

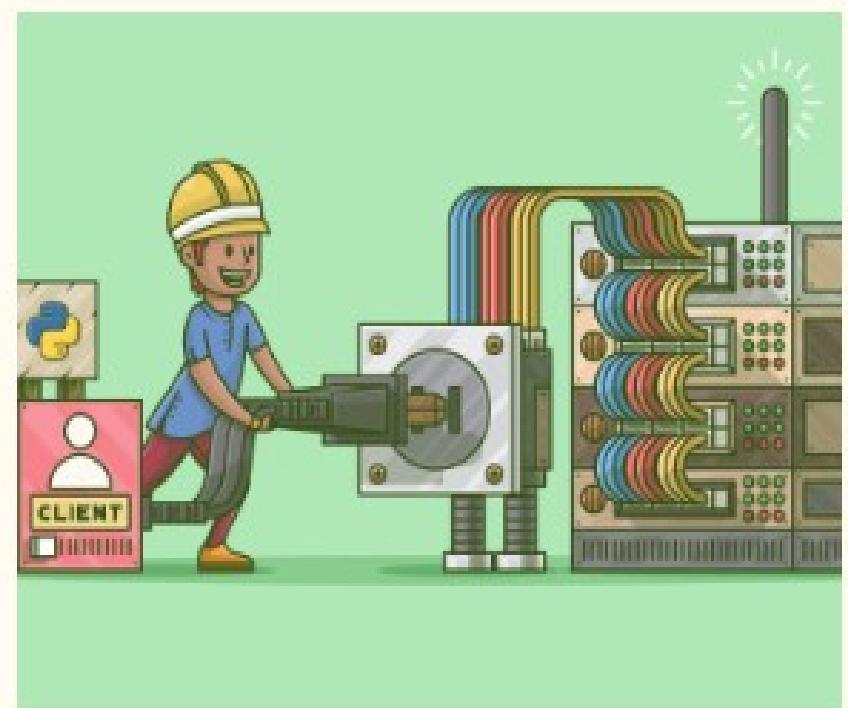


Table of contents

#Title#	.Page
Intro!!	6
1.What is Program	6
1.1. How Computer Work	7
1.2. Basic Computer Work flow	10
1.4. The Language which Can spoke to the Computer	12
1.4.1 Print out Hello, world by Assembly language	13
1.4.2 Basic Knowledge of Assembly	14
1.4.3 Assembly Register	16
1.5 Compiler vs Interpreter	18
1.5.1 Print Hello, world With C Language	18
1.6 Compared HelloWorld in Different Language	20
2. Introduction to Python	21
2.1History of Python	21
2.2 Python Installation	21
2.3 What can we do with Python	24
2.4 What python is?	24
2.4.1 Python Interactive mode & Script Mode	25
2.5 Very first Program "Hello world" and its syntax, key-words	26
2.6 Python Key-words	29
2.7 single vs double quotes	31
2.8 Variable	31
2.8.1 Numbers	34

2.8.2 String	35
2.8.2.1 int() str() float()	37
2.8.3 Python Lists	37
2.8.4 Python Tuples	40
2.8.5 Python Dictionary	41
3. Python Operator	45
3.1 Arithmetic Operators	45
3.1.1 Addition (+)	46
3.1.2 Subtraction (-)	47
3.1.3 Multiplication (*)	47
***** Python User Input *****	48
***** Python 2.7(dvision, Modulus, Exponentiation) *****	51
***** Python 3.7(dvision, Modulus, Exponentiation) *****	52
3.2 python Comparison operator	53
3.3 Python assignment Operator	56
3.4 Python Membership Operator	58
3.5 Python Identity Operator	59
4. Python Decision maker	60
4.1 if statement	60
Python Indentation	62
4.2 Python if else statement	64
4.3 Python if .. elif .. else statement	66
4.4 Nested if statement	69
5. Python Loops	70
5.1 Python for loops	70

5.2 Python While loop	75
5.2.1 Infinite loop	77
5.2.2 Else statement with while loop	77
5.3 Nested Loop	78
5.4 Loop control statement	80
5.4.1 Break Statement	81
5.4.2 Continue Statement	82
5.4.3 Python Pass Statement	83
6. Python - Date & Time	83
6.1.1 Python Datetime, Date, Time	84
6.1.2 Date object to represent a date	85
***** import vs from import *****	85
*** >> Using from module import * << is not Best practice ***	88
6.2 Format date using strftime()	91
6.3 strftime()	93
6.4 Current time of time zone	94
** pip installation **	94
6.5 Python sleep()	95
7. Python File I/O (input/output)	96
7.1 Python File Handling	99
7.1.2 read file	99
7.1.2 Python file mode	101
7.1.3 Python write file	102
7.2 Python OS module	103
7.2.1 Renaming and deleting file	103

7.2.2 Python Directory Process	104
8. Python Error Handling	104
8.1 Python try except	108
8.2 Python Try.....except.....else	116
8.3 Python try.....except.....else.....Finally	118
9. Python - Function	122
9.1 User Defined Function	122
9.2. Python Function Argument	123
9.2.1 Required arguments	123
9.2.2. Keyword Argument	125
9.2.3. Default Argument	126
9.2.4. Python Arbitrary Arguments	126
9.2.5. lambda functions in Python	127
9.3 Python recursive function	128
10. Intro OSI 7 Layer	129
10.1. Physical Layer	131
10.2. Data Link Layer	131
10.3. Network Layer	133
10.4. Transport Layer	134
10.5. Session Layer	137
10.6. Presentation Layer	138
10.7. Application Layer	140
11. Cisco Lab basic	140
11.1 Cisco CLI Level	141
11.2 Login Lab	142

11.3 Intro to VLAN	143
11.4 VLAN lab 1	145
11.5 VLAN lab 2	146
12 Network Lab Setup	150
12.1 History of GNS3	151
12.2 Why window user Need Gns3VM	152
12.3 GNS3 install (with Several ways)	153
12.3.1 linux user to get ppa support	154
12.3.2 install GNS3 with pip3	157
12.3.3 Common GNS3 error with pip	159
12.4. Get virtual machine	161
12.5 Get Gns3vm	164
12.6 Connect Gns3 with Gns3vm	165
12.7 Get cisco Device image file (IOS/IOU)	168
12.7.1 IOU license	172
12.8. Get Python Container & Ubuntu Docker image (GNS3 Appliance)	174
13. Python Telnet Script	180
14. Config vlan with for loop	185

*** Python For Network Engineer ***

Intro!!

အားလုံးပဲ မင်္ဂလာပါနှာ.... "Python For Network Engineer" ဆိုတဲ့ စာအုပ်ကို ရေးရတဲ့ ရည်ရွယ်ချက်ကတွော Network Programming နဲ့ပါဝေသက်ပြီး မြန်မာလိုပေးသားတဲ့ စာအုပ်မတွေ တာကြောင့် စပြီးက ရေးဖြစ်သွားတာပါ။ Network သမားတွေ အနေနဲ့ router တွေ switchထွေ configure ချတဲ့နေရာမှာ script/Program တွေကိုသုံးပြီ ပိုပြီးမြန်သန်စေစိုး အမှားနည်းစေစိုးနဲ့ ကိုယ့်အလုပ်အနေထား အရ ကိုယ်လုပ်ရ မှာတွေနဲ့ ကိုက်အောင် ကိုယ် ကိုယ်တိုင် ဖန်တီးနိုင်စေစိုး ရည်ရွယ်ပါတယ်။ Network Programming အဆွဲသုံးရမယ့် Language အနေနဲ့ ကနေ Python နဲ့ပဲ ရေးသားသွားမှာပါ။ Program ရေးတယ်ဆိုပေမယ့် GUI အပိုင်းတွေ Data Base အပိုင်းတွေ ပါဝင်မှာ မဟုတ်ပါဘူး။ Python basic concept တွေ Class ထွေ Operator တွေ Library တွေ ကိုသုံးပြီး Network ပိုင်းမှာ အသုံးဝင်မယ့် ကိုယ်ပိုင် Scriptတွေ Toolsတွေ ကို ဖန်တီးရေးသားပြမှာပါ။

ဘာလို့လဲ ဆိုတော့ ကျွန်တော်တို့ network engineer တွေ အနေနဲ့ programming language တစ်ခုကို လွှေလာပြီး network_config script တွေရေးမယ်ဆိုရင် Configuration နဲ့ တာတွေ analysis လုပ်တာတွေ ပြီးတော့ network တစ်ခုကို implementation လုပ်တဲ့နေရာမှာ လွှာတော်မြန်စေမှာ ဖြစ်ပါတယ်။ နမူနာ တစ်ခုပြောရရင် vlan တွေ ထပ် create လုပ်မယ် name ငဲ တွေပေးမယ် ပြီးတော့ route တွေထပ်add မယ် စတာတွေကို လူကိုယ်တိုင် လုပ်မယ်ဆို command ငဲ တွေကို တစ်ကြောင်းချင်း တစ်လုံးငယ် တစ်လုံးထွက်လိုက်ရိုက်နေရမယ်။ အဲလို့ လုပ်မယ့်အစား program script တစ်ခု ရေးထားပြီး run လိုးက်မယ်ဆို အမှားနည်းမယ် (typing error) ပြီးတော့ နာရီပိုက်တစ်နာရီလောက်ကြောမယ့်အနိမ့်က ၃မီန်းလောက်နဲ့ ပြီးမယ် ; -)

1.What is Program

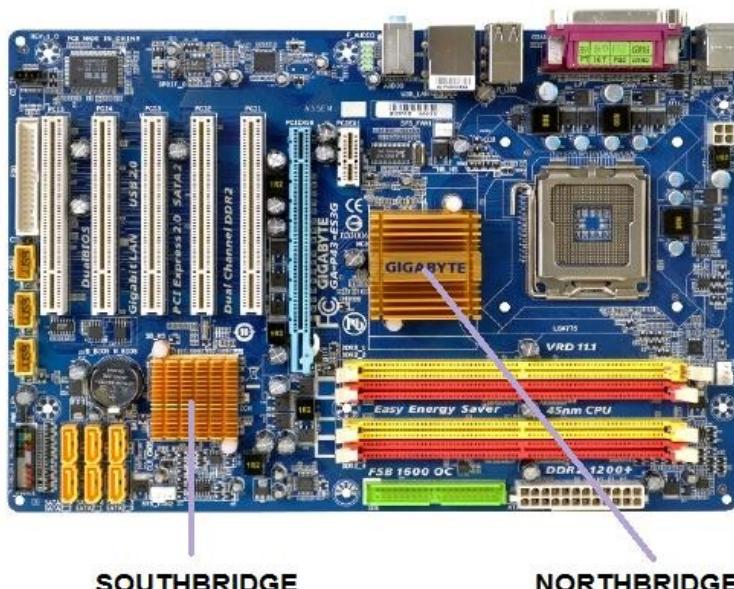
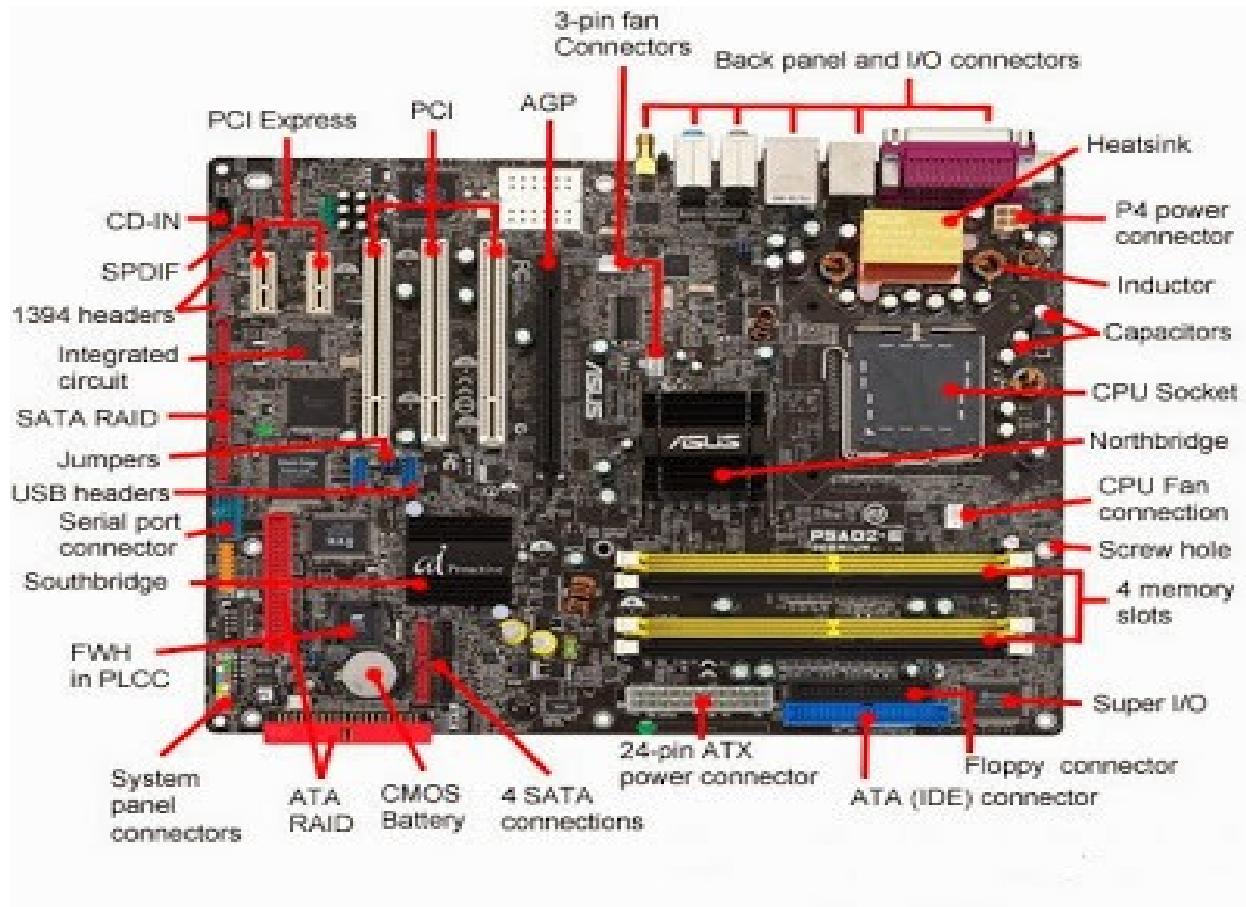
- (1) မီးလာမလာ ကြာည့်ပါ။
 - (2) မီးဖို့ချောင်မှာ ကြာက်ဥရှိမရှိ အရင်စစ်ပါ
 - (3) မရှိရင် ပိုက်ဆံယူပြီး ဆိုင်မှာ အရင်သွားပတ်ပါ
 - (4) ရေထည့်ပါ။
 - (5) Boiler ကို ပလတ်ပေါက်တပ်ပြီး မီးဖွဲ့ပါ
 - (6) ကြာက်ဥ ကို ထည့်ထားပါ
 - (7) မိနစ် ၃၁ ခန့် ကြာရင် အပြင်ထုတ်လိုက်ပါ
 - (8) အအေးခံပြီး ဘားလျှို့ပေါ်မြှုံး :-)

အပေါ်က instruction တွေအတိုင်း အစဉ်လိုက် လုပ်သွားရင် အဆင်ပြေပြီးသွားပါလိမ့်မယ် တစ်ကယ်လို့ အဖိအစဉ်မရှာပဲ တစ်ချက်လောက်ရာနဲ့ ရင် ဥပမာ No.5 အချက်ရာနဲ့ရင် ဌာက်ညပြုတ်က စားရမှာမ ဟုတ်တော့ဘူး လူအနေနဲ့ကတော့ မေ့ရာနဲ့ရင် ရာနဲ့မှန်းသိပြီး လုပ်လိုက်လို့ရပေါမယ့် Computer အနေနဲ့ရာဇ်တဲ့ လူသားလို့မစဉ်းစားနိုင်တဲ့အတွက်ကြောင့် ပေးတဲ့ Instruction အတိုင်း ဌာက်ညရှိမရှိကြည့် ရှိရင် ထည့်ပြီး ပြုတ် နာရီဝက်ကြောရင် အပြင်ကိုထုတ် အဲလိုလုပ်သွားမှာ အစဉ်လိုက် တစ်သွေးမတိမ်း လိုက်လုပ်သွားမှာ ဖြစ်ပါတယ်။ (ပြောရရင်တော့ မောင်ပုံကို ခွေးကြည့်နိုင်းသလိုပေါ့) .. ။ အဲကြောင့် computer ကို နိုင်းမယ်ဆို အဖိအ စဉ်တရာ့ တစ်ဆင့်ချင်း နိုင်းရပါမယ်။

Program ရေးတဲ့ သူတွေကို Programmer လိုပေါ်ပြီး Programming လုပ်တယ်လို့ ပြောရာပါတယ် Programming လုပ်မယ်ဆိုရင် Computer တွေကို နိုင်းဖို့ instruction တွေရေးဖို့ အတွက် computer တွေ ဘယ်လို့ အလုပ်လုပ်လဲ ပြီးတော့ ဘယ်လိုပြောပြီး နိုင်းရမလဲ ဆိုတာတွေကိုသိဖို့ နားလည်ဖို့လိုအပ်ပါတယ်။ ပြောရာပါတယ်။ ရှုရားကြီးနဲ့ အတူအလုပ်လုပ်မယ်ဆို ရှုရားလို တက်ရမယ် တရုတ်ကြီးနဲ့ အလုပ်လုပ်မယ်ဆို တရုတ်လိုက်မှ ကိုယ်က သူနားလည်အာင် ပြောနိုင်မှ အဆင်ပြေမှာပေါ့များ။ သူက ကိုယ့်ဘာဘာ စကားတက်နေရင်တော့ တစ်မျိုးပေါ့။ computer/device တွေ စက်တွေကို နိုင်းမယ်ဆိုလဲ အဲကောင်တွေ နားလည်တဲ့ ဘာဘာစကား(programming language) ကို သိမှ နိုင်းလိုရမှာပေါ့များ။

1.1. How Computer Work

Computer ကြီးကို နားလည်ဖို့ နဲ့ လောက်ကြည့်လိုက်ရာ ရအောင်များ သူမှာ ဘာတွေ ပါတယ် ဘယ်လိုလုပ်တယ်ဆိုတာ အကြောင်းများလည်နေတော့ Code ရေးရတာ ပိုအဆင်ပြေတာပေါ့များ



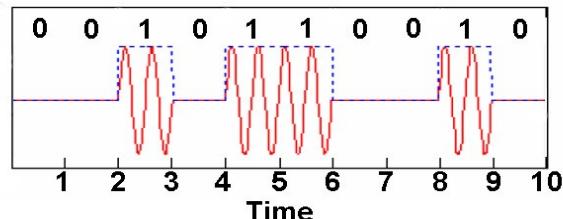
SOUTHBRIDGE **NORTHBRIDGE** ပုံမှာဆို computer motherboard တစ်လုံးမှာ ပါဝင်တဲ့ အဓိကအပိုင်းတွေကို ဖော်ပြထားပါတယ်။ ဒေါ်အဓိကအပိုင်းတွေ အားလုံးကို အစိကအားဖြင့် North Bridge နဲ့ South Bridge လို့ ခေါ်တဲ့ IC cheap နဲ့ချိတ်ဆောင်ရွက်ခဲ့တယ်။



အဲ Cheap နှစ်ခုကို North နဲ့ South ခွဲခြားလဲဆိုတော့... သူတို့၏ MotherBoard ပေါ်မှာ တည်ရှိထဲ location_Map အနေအထားငြောင်ပါ..North Bridge သည် motherboard ရဲ့ North(အပေါ်ပိုင်း) မှာရှိပါ၊ South_Bridge သည် Mother Board ရဲ့ South(အောက်ဘက်ခြေများ)မှာ တည်ရှိပါတယ် အစိတ်အပိုင်းငဲ တွေ အားလုံးကိုသေးငယ်တဲ့ wire လမ်းငြောင်းလေးတွေနဲ့ နှစ်ဆက် ထားပါတယ်။

သူတို့တွေ တစ်ခု တစ်ခု ဘယ်လို့ ဆက်သွယ်ရာလဲဆိုတော့ wires တွေကနေတစ်ဆင့် on/off signal (power ပေးမယ်မပေးဘူး) တွေအနေနဲ့ သွားပါတယ် on(voltage ပေးရင်) ဆိုတာ One ဖြစ်ပြီး off(voltage မပေးရင်) ကတော့ Zero ပါ။ computer hardware တွေသည် on/off တစ်နည်းငြောရရင် 0/1 ဖြစ်တဲ့ Binary Form ကိုပဲ နားလည်တာပါ။ data သွေး control signal တွေအားလုံးကို 0/1 binary form အနေနဲ့ ပို့ဆောင်ပြီး အချင်းချင်း ဆက်သွယ်အလုပ် လုပ်ရာပါတယ်။ 0/1 on/off signal တွေကို ဂါယာကိုပေါ်မှာ အချိန်အပိုင်းအခြား (Time Frame) တစ်ခု ဗြားပြီး ပို့ဆောင်ပါတယ်..

Binary Digit (Bit)	Electronic Charge	Electronic System
1		ON
0		OFF



ဂါယာတစ်ပင်ကို 1bit (1 byte = 8 bit)လို့ယူဆပြီး 1 bit မှာ ဖြစ်နိုင်ခြေ 2 ခုရှိပါတယ် 0ဖြစ်ရင်ဖြစ်မယ် မဖြစ်ရင် 1 ဖြစ်ပါမယ်။ ဂါယာနှစ်ပင်ဆိုရင် 2bit ဖြစ်ပြီးသူ့မှာ 00, 01, 10, 11 ဆိုတဲ့ ဖြစ်နိုင်ခြေလေးခုရှိပါတယ်။ 3bit ဆိုရင်တော့ 000, 001, 010, 011, 100, 101, 110, 111 ဆိုပြီး ဖြစ်နိုင်တဲ့ Form လေးခုရှိပါတယ် formula မနေနဲ့ ကတော့ 2^n ပဲဖြစ်ပါတယ်။ wire 1ပင်ဆို 2^1 ဖြစ်တဲ့အတွက် possibility 2 ခုရှိပါတယ် wire 3 ပင်ဆို 2^2 ဖြစ်တဲ့အတွက် ဖြစ်နိုင်ချေ 4 ခု ရှိပါတယ်။ 1byte မှာ 8bit ရှိပါတယ်။ Charater တစ်ခုကို 1byte ယူပါတယ် ဘယ်လိုမျိုးလဲဆိုတော့ keyboard ကနာ လို ရှိက်ထည့်လိုက်ပေမယ့် computer တစ်ကယ်တမ်း နားလည်တာက a ဆိုပြီးနားမလည်ပါဘူး ASCII code ဖြစ်တဲ့ 0110 0001 ဆိုပြီး Binary Form အနေနဲ့ နားလည်တာပါ ဥပမာ လူသားတွေက hope လိုရေးလိုက် ရင်လဲ computer နားလည်မှာက 0110 1000 0110 1111 0111 0000 0110 0101 ဆိုပြီးပဲ နားလည်တာပါ။

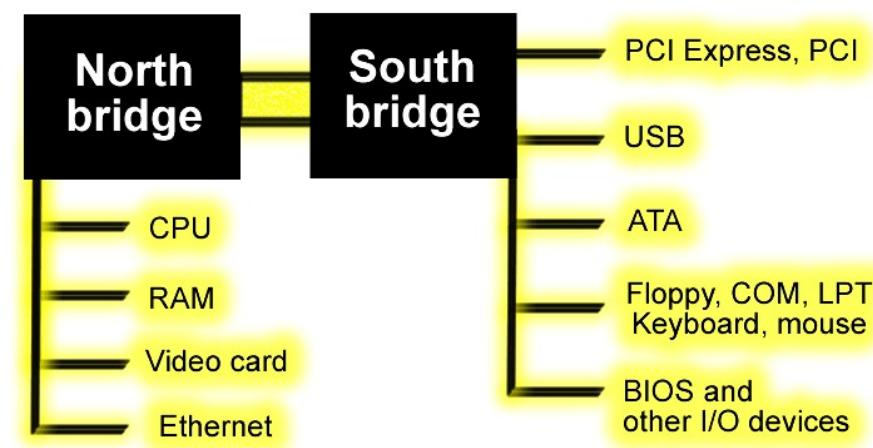
Converting the text “hope” into binary

Characters:	h	o	p	e
ASCII Values:	104	111	112	101
Binary Values:	01101000	01101111	01110000	01100101
Bits:	8	8	8	8

ComputerHope.com

North Bridge ကို memory controller hub (MCH) လွှဲလဲ ခေါ်တယ်။ North Bridge ဟာ အဓိကအဖော်နဲ့ CPU, RAM, AGP(Accelerated Graphic Port) တွေကြားမှ communication ကိုတာဝန်ယူလုပ်ဆောင် ပါတယ်။ ဥပမာအဖော်နဲ့ ပြောရရင် CPU ဟာ RAM ပေါ်မှာရှိတဲ့ Data လို လိုချင်တယ်ဆိုရာပါၟၟ CPU နဲ့ RAM က တိုက်ရှိက်ချိတ်ဆက်ထားခြင်းမရှိလေတွော ကြားမှာရှိတဲ့ North Bridge ကို CPU ကနေလှမ်းပြောရတယ်

။ သို့ Address က data ငဲ တွေလိုချင်တယ်ဆိုတာကိုပေါ့ အဲ မိန့်ကြာမှ North Bridge ကနေပါ။ CPU လိုချင်တဲ့ data ကို RAM ပေါ်ကနေလှမ်းယူပြီး CPU ကို ပို့ဆောင်ပေးပါတယ်။ South Bridge ကိုရှာ I/O controller Hub(ICH) လွှဲလဲ ခေါ်တယ်။ သူကတွော အဓိက အားဖြင့် computer တစ်လုံးရဲ့ I/O functions တွေဖြစ်တဲ့ USB, audio, serial, the



system BIOS, the ISA bus, the interrupt controller and the IDE channels တွေကို တာဝန်ယူ ထိန်းချုပ်ပါတယ်..

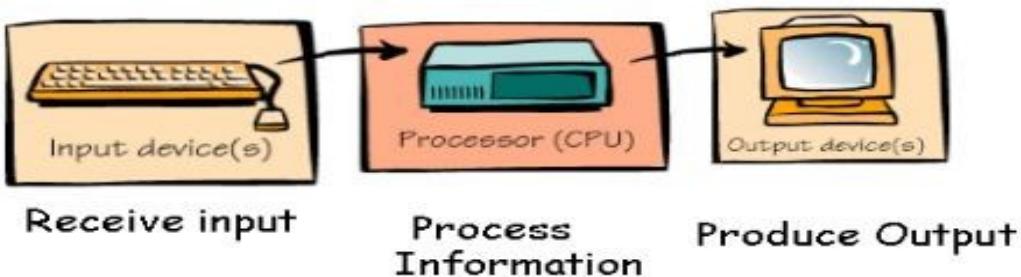
1.2. Basic Computer Work flow

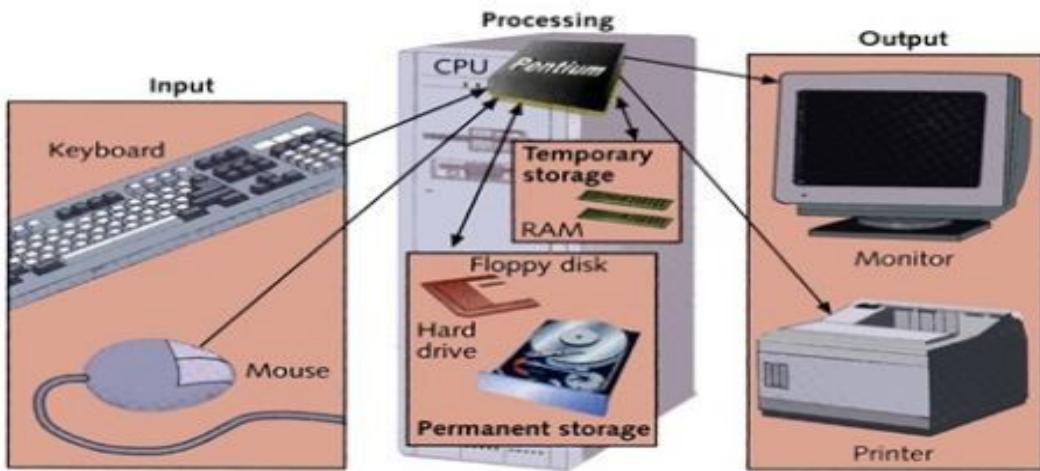


အောက်ကပုံမှာ ြာြာလိုက်မယ့်ရင် Computer တစ်လုံး၏ အခြေခံ Process Flow ကို မြင်ရမှာဖြစ်ပါတယ်။ ပထမဆုံး Input Devices တွေဖြစ်တဲ့ Keyboard, Mouse, Joy Stick, Light pen, Scanner, Microphone, Bar Code Reader တွေကနေ input data တွေ ပင်လာပါမယ်။ Computer အနေနဲ့ input device(Eg., KeyBoard) ကနေဝင်လာတဲ့ input Dataအော့ Instruction တွေကို RAM အရင်သိမ်းဆည်းပါး အဲ RAM ကနေတစ်ဆင့် KeyBoard ကရိုက်ထဲတြော့သမျှ ကိုလဲ Monitor မှာ မြင်အောင် တစ်ခါတည်း တိုက်ရိုက်ပြပေးပါတယ်။

ြီးတော့ ပင်လာတဲ့ instruction တွေကိုတွေ့ RAM ကနေ တစ်ဆင့် CPU(Central Processing Unit ကို ပို့ဆောင်ပေးပါတယ်။ တွေကရှုက်တာ စတဲ့ processorတွေ အတွက်ပါ။ CPU သည် လူသားတွေရဲ့ ဦးစွဲနာက်နဲ့ တူပါတယ် တွေ့ကိုတာရှုက်တာတွေ computer မှာ ပါဝင်တဲ့ အစိတ်အပိုင်းတွေ ရဲ့ operation များကို ထိန်းချုပ်တာတွေ စတာတွေကို (CPU ရဲ့ အဓိက တာဝန်တွေဖြစ်ပါတယ်) CPU ရဲ့ cache Memory ပေါ်မှာ ထားပါတယ်။ cache memory သည်လဲပဲ RAM အမျိုးအစား ထဲမှာပါပါတယ် processing speed ပိုမြန်ချင်စေနဲ့ အတွက် CPU နဲ့ အနီးဆုံး နေရာမှာ ရှိနေတာပါ။ သူ့သည် RAM (Main Memory) နဲ့ CPU ြာြားမှာ buffer အနေနဲ့လဲ လုပ်ဆောင်ပေးပါတယ် .. CPU သည် ပင်လာတဲ့ Data တွေကို Instruction အရ execute လုပ်လိုက်ပါတယ်.. CPU ကနေ ရလာတဲ့ result ကို RAM ကနေတစ်ဆင့် output devices တွေဖြစ်တဲ့ Monitor တို့ ကနေ ပြပေးပါတယ်။

What Computers Do





Computer activity consists of input, processing, storage, and output

RAM(Random Access Memory) ကို Main Memory လိုလဲ ခေါ်ပါတယ်။ RAM သည် ကွန်ဖူတာ run နေစဉ် အတွင်း လက်ရှိ ရှိနေစဲ အချက်အလက်ကိုမြန်မြန်ဆန်ဆန်သိမ်းဆည်း မြန်မြန်ဆန်ဆန် ထုတ်ပြန် ငြို့သောလဲ ယာယိပဲသိမ်းနိုင်ပါတယ်။ RAM ကို capacitor တွေနဲ့ လုပ်ထားတာဖြစ်လို့ computer Power on ထားချိန် electronic current ပီးနေချိန်မှာသာ ရှိနေတာပါ။ RAM မှာ Data တွေကို ဘယ်လိုမှတ်လဲဆိုတော့ RAM မှာရှိစဲ အတွက် Capacitor တွေကို charged/discharged လုပ်စဲ ပုံစံနဲ့ မှတ်ပါတယ်။ charged ဆိုရင် 1ဖြစ်ပါ ဒါး discharged ဆိုရင်တော့ 0ဆိုစဲ Binary Form နဲ့မှတ်ပါတယ်။ Computer ပိတ်လိုက်တာနဲ့ current မ ပီးတော့ အတွက် Capacitor တွေ discharge ဖြစ်ပြီးတော့ ပျောက်သွားမှာပါ။ အဲလိုမဖြစ်စေနဲ့ ပိတ်လိုက်လဲ မဖျက်ချင်ဘူးဆိုရင် Secondary Memory လို့ ခေါ်စဲ Hard Disk ပေါ်မှာ သွားပြီး SAVE ထားမှ နောင်တစ် နှင့်လိုတဲ့ အခါ ပြန်သုံးလိုရမှာပါ။ Hard Disk ရာ သတ္တုပြားပေါ်မှာ သံလိုက်ရည်ပါးပါးလေး သုတေသားတဲ့ သံပေါ်မှာ မှတ်ပါတယ်။ သံလိုက်သာဘဝ ဖြစ်စဲ north(N) နဲ့ south(S) ကို 0 နဲ့ 1 အနေနဲ့ ယူဆပြီးမှတ်ပါတယ်။ ပါဂါဝိတ်လိုက်လဲ သူဆီမှာရာ data ကတော့ ရှိနေမှာပါ။ Hard Disk မှာ data ဖတ်စဲ နှန်းက RAM လောက်မမြန်ပါဘူး။

1.4. The Language which Can spoke to the Computer

အပေါ်မှာလဲ ပြောခဲ့ပါတယ် Computer တွေဟာ Machine Code တွေ ဖြစ်စဲ 0 နဲ့ 1 ကိုသာနားလည်ရှာတာပါ။ Human Language ဖြစ်စဲ character, integer တွေ စကားတွေကို နားမလည်ပါဘူး အဲတော့ computer နားလည်အောင် ပြောပေးနိုင်မယ့် Low Level Programming Language နဲ့ ဖြစ်ဖြစ် High Level Programming Language နဲ့ ဖြစ်ဖြစ် ရေးသားပြီး နိုင်းစေရာပါတယ်။

Low Level/High Level Programming Language တွေရဲ့ Source code(Human Readable)ဆွဲ ကန့်Machine Code သို့ပြောင်းလဲပေးတဲ့ Translator 3 မျိုးရှိပါတယ်..

- (1) Assembler- Assembly language ကန့်သို့ Machine code အဖြစ်သို့ ပြောင်းလဲပေးပါတယ်
- (2) Interpreter- သူကရာ instruction တွေကို တစ်ကြောင်းပြီးမှ နောက်တစ်ကြောင်း Machine Code တွေ အနေနဲ့ translate လုပ်ပါတယ်။
- (3) Compiler- Compiler အနေနဲ့ သူက Program ပြီးတစ်ခုလုံး Source Code ပြီးတစ်ခုလုံးကို Machine code အနေနဲ့ ပြောင်းပြီးမှ RUN ပါတယ်။
- (4) Low Level Language မှာဆို Lower Level ဖြစ်တဲ့ Hardware ဘက်ခြမ်းနဲ့ အရမ်းနီးကပ်ပါတယ် Hardware တွေရဲ့ အလုပ်လုပ်ပုံနဲ့ ဆင်ပါတယ်။ Code ရေးရာတွင် CPU architecture ပေါ်မှတည်ပြီးပြောင်းလဲပါတယ်။ Program ရေးရင်လဲ Hardware တွေ လုပ်ဆောင်ပုံအတိုင်း တစ်ခုချင်း ရေးသားရပါတယ်။ ဥပမာအနေနဲ့ အောက်က code ကိုဖြော်ပါ။

1.4.1 Print out Hello, world by Assembly language

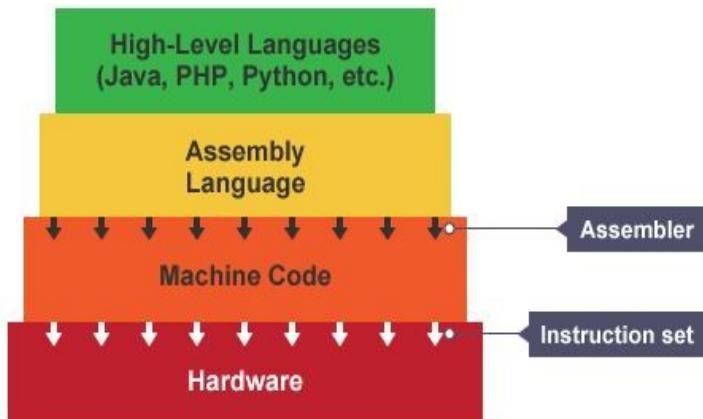
```
:Hello.asm

section .data
    text db "Hello world!",10
section .text
    global _start

_start:
    mov rax, 1
    mov rdi, 1
    mov rsi, text
    mov rdx,14
    syscall

    mov rax, 60
    mov rdi, 0
    syscall
```

Low Level Langauge ဖြစ်တဲ့ Assembly နဲ့ "Hello World!" ဆိုတဲ့ စာသားလေးတစ်ကြောင်းထဲ တိမ့် ရေးထားတာပါ။



စမ်းကြေည့်ချင်တဲ့ သူတွေကတွာ အဲ
တာဘား(Source Code)တွေကို copyကူးပြီး Text Editor တစ်ခန့် Hello.asm ဆိုပြီး save ပါ။ Source Code သည် Computer ရဲ့ architecture ပေါ်မှတည်ပြီး ကွဲပြားပါလိမယ်။ ခုကတွာ Linux OS, intel 64bit CPU အတွက်ပါ Assembly Language အောင် Netwide Assembler(NASM) လိုပေါ်တဲ့ Assembler နဲ့ "\$nasm -f elf64 -o Hello.o Hello.asm" ဆိုတဲ့ command လေးနဲ့ Compile လုပ်လိုက်ရင် dot.o(*.o) ဆိုတဲ့ Object file လေး ထွက်လာပါ လိမယ်။ ဒီနေရာမှာ တစ်ချက်မှတ်ရမှာက NASM သည် assembler ပါ Compiler မဟုတ်ဘူး assembly အောင် compile လုပ်ပေးတယ်ဆိုတာလေးပါ။ အဲဖိုင်လေးကို execute လုပ်ဖို့ အတွက် Link ချက်ပေးဖို့လိုပါတယ်။ " \$ld Hello.o -o hello " ဆိုတဲ့ command နဲ့ လင့်ချိတ်ပေးလိုက်မှ ထွက်လာတဲ့ Output ဖြစ်တဲ့ hello ဆိုတဲ့ file လေးသည် executable ဖြစ်ပါတယ်။ အောက်က ပုံကို တစ်ချက်ကြေည့်ပေးပါ။

```

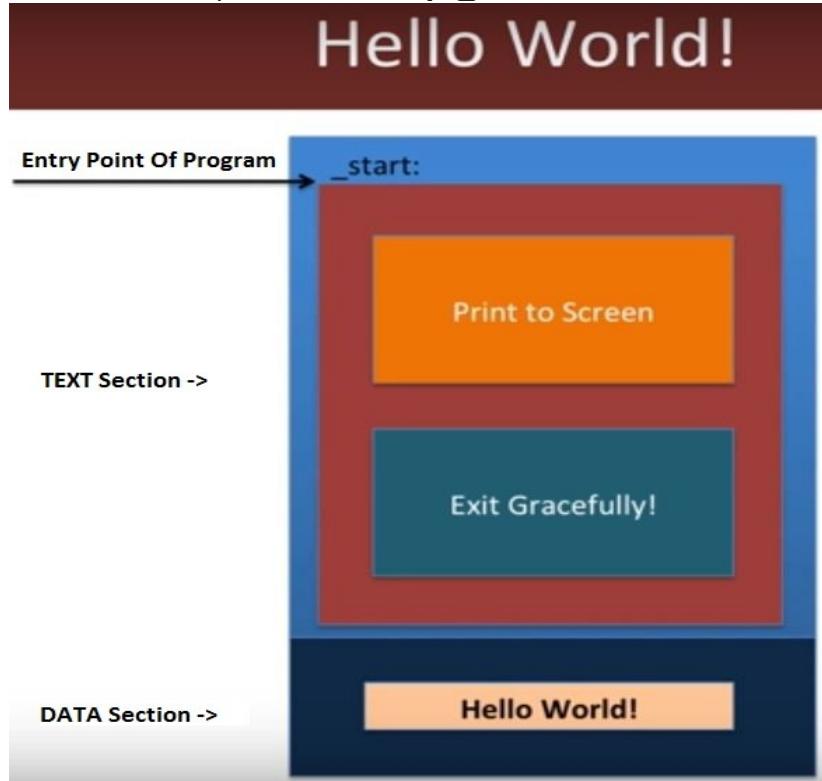
ab
[Desktop Icons]
hello
Hello.asm
Hello.o

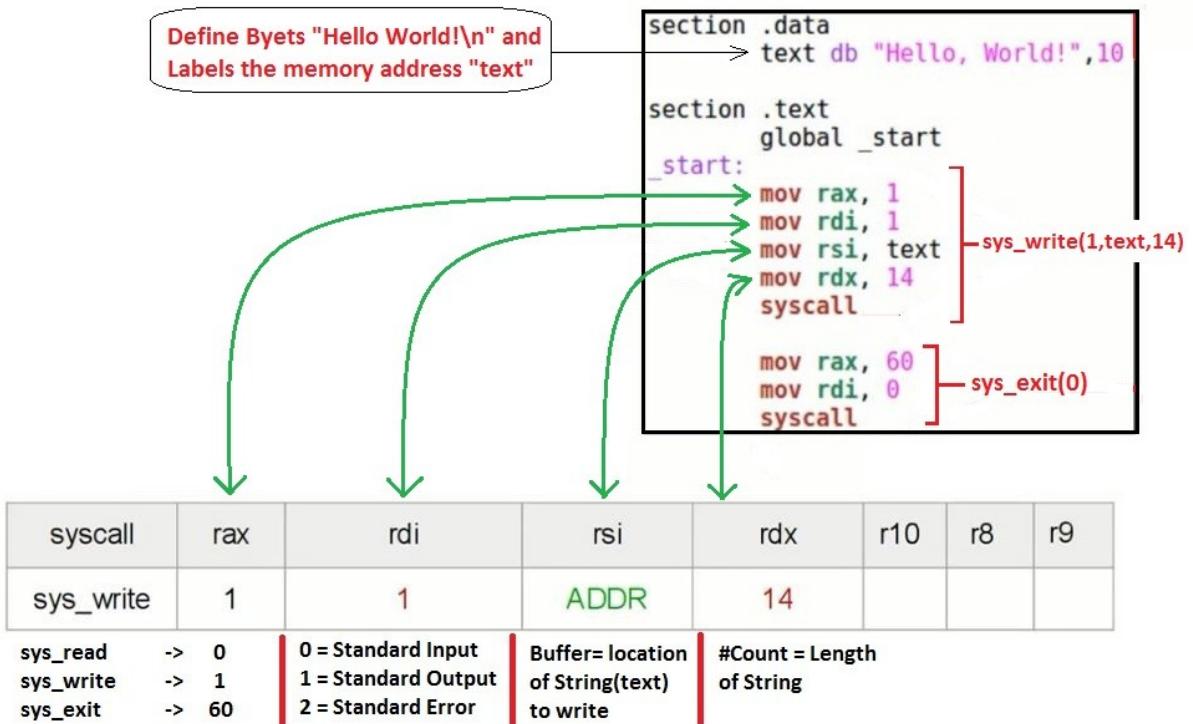
[Terminal]
spacex@spacex-Lenovo-ideapad-110-14IBR: ~/lab
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab 81x27
spacex@spacex-Lenovo-ideapad-110-14IBR:~$ cd lab/
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab$ uname -a
Linux spacex-Lenovo-ideapad-110-14IBR 4.4.0-79-generic #100-Ubuntu SMP Wed May 17
UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab$ nano Hello.asm
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab$ nasm -f elf64 -o Hello.o Hello.asm
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab$ ld Hello.o -o hello
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab$ ls
hello Hello.asm Hello.o
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab$ ./hello
Hello world!
spacex@spacex-Lenovo-ideapad-110-14IBR:~/lab$ 
  
```

အဲ hello ဆိုတဲ့ Executable file လေးကို RUN လိုက်မယ်ဆိုရင် Hello ဆိုတဲ့ output လေးထွက်လာပါပြီး..

1.4.2 Basic Knowledge of Assembly

အဲလို Hello, world ဆိတာလေး ထွက်ဖိုအတွက် High Level Language လောက်မလွယ်ကူပါဘူး။ Assembly ဆိတဲ့ Low Level Language မှာ Program တစ်ပုဒ်အတွက် အပိုင်းသုံးပိုင်းပါတယ်။ Data section, Text section နဲ့ Bss Section တို့ပဲဖြစ်ပါတယ်။





အပေါကုပုလေးကို ကျွန်တော် တစ်ခုကိုရင်းပြေပေးပို့မယ်။

.data section မှာဆိုရင် complie လုပ်တဲ့ နေရာမှာ လိုအပ်မယ့် data တွေကို အရင် define လုပ်ပါတယ်။

.text section ကတွော တစ်ကယ် run မယ့် code တွေကို ရေးထားပါတယ်။

ခုံcode မှာ မပါတဲ့ .bss section ကတွော နောင်ရာသုံးမယ့် data အသွေး memory ပေါ်မှာ ဖြို့ဖြို့ နေရာယူထားရမယ် အခြေအနေမှာ သုံးပါတယ်။

ပြီးတွော `text db "Hello, world!",10` မှာဆို db(define bytes) ဆိုတာသည် define raw byte of data ပါ။ code မှာ ထည့်ရမယ့် data ပေါ့။ ပြီးတွော "Hello, world!",10 ကတွော ရှိ မှာပြောခဲ့တဲ့ bytes of data ပါ ..Hello, world! ဟဲတို့ e တို့၊ တို့ ကောင်တွေက single byte အ နေနဲ့ တည်ရှိတာပါ။ ပြီးကတွော နောက်က 10 ဆိုတာကရာတွေက newline ကို ပြောတာပါ။ \n ပေါ့။ text ဆိုတာက hello world! ဆိုတဲ့ data ကို memory မှာ သိမ်းရင်.. သိမ်းမယ့် memory address ကို ဉာဏ်ဆိုပေးမယ့်ကောင်ပါ။ code ကို compile လုပ်ရင် text နေရာမှာ memory location ကို replace လုပ်ပြီး run သွားမှာပါ။

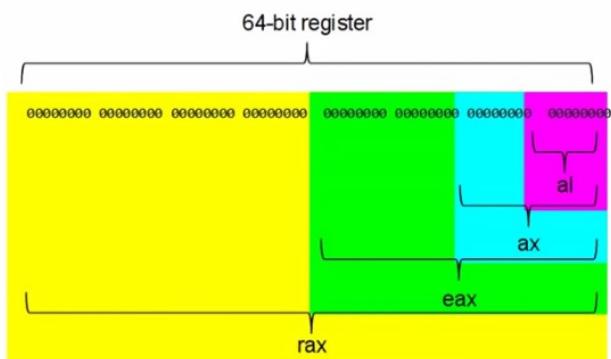
.text section မှာ ရှိနေတဲ့ global _start ဆိတာကတွေ့ ရေးကို ရေးရမှပါ။ အဲတာရှာမှ ဘယ် ကနေစပြီး execute လုပ်ရမယ်ကို သိမှာပါ။ global သည် linker ကို သုံးပြီး run လိုက်တဲ့အန္တိနှင့်မှာ memory address သိစေခဲ့တဲ့ label တွေနဲ့ တွဲပြီး ကြော်ပေးရတာပါ။

အောက်_start ဆိတာကရှာတော့ program တိုင်း အတွက် မပါမဖြစ်ပါပဲ။ program တွေကို စပြီး execute လုပ်မယ်ဆို _start ကနေ စပြီး လုပ်ပါတယ်။ အကယ်လို Linker သည် _start ကို မတွေ့ရင် error တို့လာပါလိမ့်မယ်။

1.4.3 Assembly Register

အောက်code လေးတွေနဲ့ register တွေကို ဆက်ပြီး လေ့လာကြည့်ရာရအင် များ။

Registers



8-bit	16-bit	32-bit	64-bit
al	ax	eax	rax
bl	bx	ebx	rbx
cl	cx	ecx	rcx
dl	dx	edx	rdx
sil	si	esi	rsi
di	di	edi	rdi
bpl	bp	ebp	rbp
spl	sp	esp	rsp
r8b	r8w	r8d	r8
r9b	r9w	r9d	r9
r10b	r10w	r10d	r10
r11b	r11w	r11d	r11
r12b	r12w	r12d	r12
r13b	r13w	r13d	r13
r14b	r14w	r14d	r14
r15b	r15w	r15d	r15

ခုလက် ရှိ run နေတဲ့ PC သည် linux OS, CPU သည် 64 bit processor ဖြစ်ပါတယ်။ ခု helloworld ထုတ်ဖို့လိုအပ်မယ့် register တွေကို အရင် တစ်ချက် လေ့လာပြောလိုက်ရအောင်။

Rax ဆိုတဲ့ register တဲ့ကို value 1 ထည့်လိုက်ပါတယ်။ 1 ထည့်လိုက်ချင်းအားဖြင့် sys_write လုပ်မယ်လို . ပြောလိုက်တာပါ။ rdi ကတွော file descriptor ဖြစ်ပြီးတွော 0 သည် iwrite, 1 သည် output, 2 သည် error ပါ။ ခု output ထုတ်မှာဖြစ်လို rdi ထဲ ကို MOV ကို သုံးပြီး 1 ထည့်လိုက်ပါ။ rsi (buffer) ကတွော location of string ကို ထည့်ပေးပါမယ်။ Hello, world! ရှိနေတဲ့ memory address သည် text မှရိတာမို့ mov rsi, text ဆိုပြီး rsi ထဲ ကို ထည့်လိုက်ပါ။ rdx ကတွော data ရဲ့ length ကို ထည့်ပေးရတာပါ။ ပြီးစုတဲ့ syscall လို ရေးပါတယ်။ linux ရဲ့ syscall ကို assembly မှာ ယူလို ရပါတယ်။ system call ID ကို rax ထဲ မှာ ထည့်ပါတယ်။ ပြီးတွော rdi တို့ rsi တို့မှာ arguemnt တွေ လိုက်ထည့်ပါတယ်။ အကုန်ပြီးသွားရင် syscall ဆိုပြီး ရေးလိုက်ရင် ရှုံးကထည့်ခဲ့တဲ့ register value တွေပေါ်မှတည်ပြီး result ကို retrunပြန် ပေးပါတယ်။

နောက်အောင်block ဖြစ်တဲ့ mov rax, 60 ဆိုပြီး rax register ထဲ ကို 60 ထည့်လိုက်ပါတယ်။ 60 သည် sys_exit ဖြစ်ပါတယ်။ အဲနောက် rdi ကို 0 ပေးလိုက်ပါတယ်။ syscall ခေါ်လိုက်တဲ့အနဲ့မျိန် ရာ program သည် ထွက်သွားပါလိမ့်မယ်။ High Level programming မှာ တွော compiler က တာဝန်ယူပေမယ့် ခု assembly မှာတွော programmer က နေပြီးတွော exit လုပ်ပေးရပါမယ်။ မပါခဲ့ရင် error တက်ပါတယ် စမ်းကြေည့်လို ရပါတယ်။

ပြီးတွော Low Level Language တွေမှာ compiler တွေ interpreter တွေ မလိုအပ်ပါဘူး တိုက်ရှိရှိ Run နိုင်ပါတယ်။ High Level Language ဆိုတာကို user-friendly ပိုဖြစ်ပါတယ် လူသားတွေအနေနဲ့ နားလည်လွယ် တဲ့ Syntax တွေ logic တွေနဲ့ ရေးသားထားတဲ့ source code ကိုမှ Compiler ဖြစ်ဖြစ် Interpreter ဖြစ်ဖြစ် သုံးပြီး Computer နားလည်အောင် Machine Code အနေနဲ့ ပြောင်းလဲပါတယ်။

1.5 Compiler vs Interpreter

High Level Programming Language ကို အဓိကအနေနဲ့ နှစ်မျိုးခဲ့ထားလိုရတယ်များ။ Compiled Languages ရယ် Scripting Languages ရယ် ဆိုပြီး ရှိတယ်များ။ Compiled Languages ဆိုတာ ဘာလဲဆိုတွော..

Computer တစ်လုံး တစ်ကယ်တမ်းနားလည်တာက 0/1 (on/off) တွေကိုပနားလည်တာမှာ printf ထို့
 cin cout တို့ အဲတာတွေကို နားလည်တာမဟုတ်ဘူး .. အဲတော့ ကိုယ်တွေလုပ်ချင် ခိုင်းချင်တဲ့ instruction တွေကို
 လူနားလည်တဲ့ code(Source Code) တွေကနေ computer machine နားလည်တဲ့ code(Machine Code) တွေ
 အဖြစ်ပြောင်းဖို့ လိုလာပြီ.. အဲလို လူနားလည်တဲ့ character/set of instructions တွေကနေ computer နားလ
 ည်တဲ့ 0/1 bits တွေကို translate လုပ်ပေးဖို့ compiler တွေနဲ့ compile လုပ်ရတယ်မှာ... အဲလို set of
 instructions တစ်ခုလုံးကို compile လုပ်လိုက် build လုပ်လိုက်မှ computer နားလည်တဲ့ machine code တွေ
 ထွက်လာပြီး computer တွေက ဖတ်ပြီး execute လုပ်လို့ run လို့ရမှာ.. Compiled language တွေက ဘယ်
 language တွေလဲ ဆိုတော့.. java,C, C++, C#, Visual Basic, Pascal, etc စတာတွေဖြစ်တယ် ;-)

1.5.1 Print Hello, world With C Language

Eg.,

\$vi helloworld.c

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
printf("Helloworld");
```

```
}
```

ရေးပြီး helloworld ဆိုတဲ့ နာမည့်နဲ့ ထားလိုက်ပါ ပြီးတော့

\$gcc helloworld.c -o hello ဆိုပြီး gcc(Gnu C Compiler) နဲ့ program တစ်ခုလုံး ကို compile လုပ်ပြီးထို့
 လာမည့် machine code တွေနဲ့ file ကို -o နဲ့ output ကို hello ဆိုပြီး မှတ်ထားလိုက်ပါ.. ပြီးတော့ အဲ machine
 code ဖိုင်ကို execute လုပ်လိုက်ပါ \$chmod 777 hello && ./hello

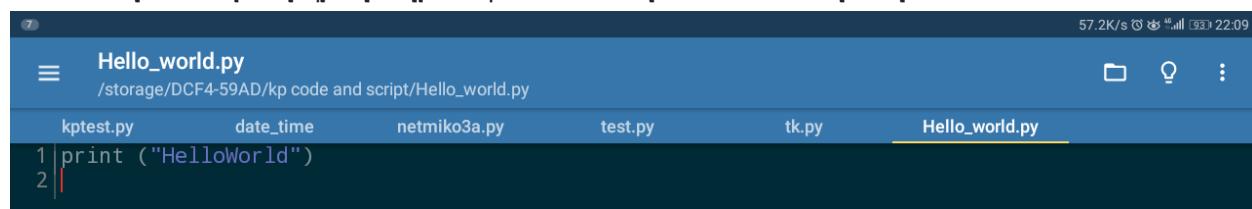
Scripting Language ဆိုတာက ရှာတွေ့ော့..

သူကရာ ပြောရရင် programကြီးတစ်ခုလုံး(set of instructions) တစ်ခုလုံးကို compile လုပ်ပေးစရာမလိုပဲ..
 instruction တစ်ခုချင်းသီးကို

one by one(Statement by statement) execute လုပ်ပေးတယ်မှာ.. ဘယ်လို့ေးလဲဆိုတော့ scripting
 language တွေက interpreter သုံးပြီး program run နေတုန်းမှာ statement တစ်ခုချင်းကို machine code အဲ
 ဖိုင်ပြောင်းပြီး တိုက်ရှိက် တစ်ခုချင်း တစ်လိုင်းချင်း execute လုပ်တယ်မှာ .. Scripting Language တွေက ဘယ်က
 ကာင်တွေလဲဆိုတော့.. Python, Ruby on Rails, Java Script, PHP,etc စတာတွေဖြစ်တယ်

Eg.,

print("Helloworld") ဆိုတဲ့ code လေးကို hello.py ဆိုတဲ့ နာမည်လေးနဲ့ save ထားလိုက်မယ်.. ပြီးတော့
 \$python hello.py ဆိုပြီးrun ပြောလိုက်ရင် Helloworld ဆိုပြီး ထွက်လာလိမ့်မယ်... သူမှာ compile language လို့
 compile လုပ်ပေးစရာမလိုဘူး တိုက်ရှိက်တန်း execute လုပ်ပေး run ပေးလိုက်ယုံပဲ..;-)



The screenshot shows a terminal window with the following content:

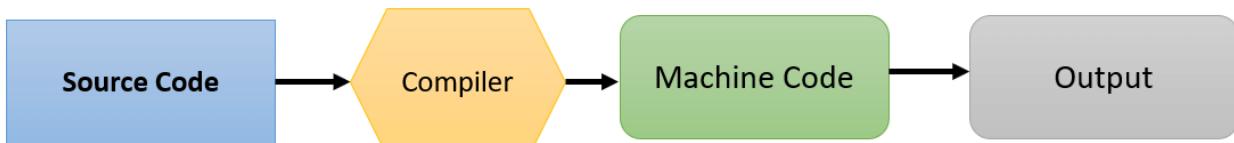
```
57.2K/s 7: 22:09
Hello_world.py
/storage/DCF4-59AD/kp code and script/Hello_world.py
kptest.py date_time netmiko3a.py test.py tk.py Hello_world.py
1|print ("HelloWorld")
2|
```

The terminal shows the command \$python Hello_world.py being run, and the output "HelloWorld" is displayed.



Scripting language ရှိထဲ compiled language မှာ ဘာက္ခာလဆိုတွေ...

How Compiler Works



© guru99.com

How Interpreter Works



compiled language သည် compile လုပ်ရတဲ့အနီးနိုင်ပါတယ်

scripting language ကတေသာ compile လုပ်စရာမလိုပါဘူး

compile လုပ်လို ထွက်လာတဲ့ machine code ကို computer တွေမှာ နောက်တစ်ရှိမ် ထပ် run စရာမလိုသလို source code ကိုလဲ ရဲပေးစရာမလိုပါဘူး ဒီတိုင်း ထွက်လာတဲ့ machine code ကို computer တွေမှာ execute လုပ်ပြီး run လိုက်ယံပါပဲ

scripting language මා තොව තර්කීකා:computer තොමා run ඔයෝ සැකිරීම් source code ගිරි පෙළපිටයා..

looping පිරිදු යුතු වන program මාධ්‍ය.. compiled language මා තම් ක්‍රියියල් compile ලබන තු අවින් ක්‍රාප්‍රී: ගොංග නැගුණු ඇත්තේ machine code ගී මුළු ප්‍රාග්‍රැම් වන්න run ලෙස දිග්‍රී යුතු පිටපත... scripting language මාදෙනා තම් ආක්‍රිතියෙන් run යුතු යුතු නොවුතු නොවුතු... අවින් ප්‍රාග්‍රැම් පිටපත...

ဟုတ်ပြီး ဒါ Low Level Language ဖြစ်တဲ့ Assembly နဲ့လဲ hellworld ဆိတဲ့ result ထုတ်ခဲ့တယ် ပုံးတွေ၊ complied language ဖြစ်တဲ့ C နဲ့ Helloworld ထုတ်ခဲ့တယ်.. Scripting Language ဖြစ်တဲ့ python နဲ့လဲ HelloWorld ထုတ်ခဲ့တယ်။ အဲတွေ့ နိုင်းယုံး ရာညွှန်အောင်။ ;)

1.6 Compared HelloWorld in Different Language



Hello World!

ဟဲ့ ဝဲ့ မဲ့ ဒဲ့

With Assembly

```
;Hello.asm

section .data
    text db "Hello world!",10
section .text
    global _start

_start:
    mov rax, 1
    mov rdi, 1
    mov rsi, text
    mov rdx,14
    syscall

    mov rax, 60
    mov rdi, 0
    syscall
```

With C

```
File Edit Search Run
[1]
#include<stdio.h>
int main()
{
    printf("Hello World!");
    return 0;
}
```

With Python

```
C:\Users\ekhaphy>python3
Python 3.7.2 (tags/v3.7.2:9a3ffc049,
Intel) on win32
Type "help", "copyright", "credits"
>>>
>>> print ("Hello world!") ←
Hello world!
>>>
>>>
```

ခုံရှာည့်လိုက်တွေ python က လွယ်တယ်ဟုတ်? အဲဒြောင့် Network programming အဆွဲ.. သုံး
မထဲ language ကို python အနေနဲ့ ဆက်ပြီး ပြောပြပေးသွားပါမယ်။ ကိုယ် စိတ်ကြိုက် language သိရင်သုံး
လိုရပါတယ်။ ရွှေ့နောကတွေ python ရေးတာ ကြိုက်တာမို့ပါ။ Programming Language တစ်ခု မှာ က
advantage disadvantage တွေ ရှိရာပါတယ်။ တစ်ခု က system ပိုင်းမှာ ပိုကောင်းတယ် တစ်ခု က web ပို
င်း တစ်ခု က app ပိုင်း မှာ ပိုကောင်းတယ်ပေါ့.. စသည်ဖြင့် ရှိရာတာပဲ.. အဲတွေ ကိုယ်ကတွေ Network
Programming မှာ Python ကြိုက်တယ်။။ က ဆက်လိုက်ရာရအောင်မှာ ။

2. Introduction to Python

2.1 History of Python

Python ကို 1980 လောက်က Guido Van Rossum ဆိုတဲ့ ပုဂ္ဂိုလ်က စတင် ဖန်တီးခဲ့ပါတယ်။
1991 ခုနှစ် ဖေဖော်ဝါရီလမှာ Python 0.9.0 ကို first version အနေနဲ့ publish ထုတ်ပေးခဲ့ပါတယ်။

Rossum က သူ create လုပ်ခဲ့တဲ့ Programming Language ကို Python လို့ အမည်ပေးဖြစ်ခဲ့တဲ့ အကြောင်းကတော့ သူကသည် Monty Python's Flying Circus(ဆပ်ကပ်) ကို အရမ်းကြိုက်လိုပါ။ ခုခံရင်တော့ Python Language သည် လူသုံးများတဲ့ language တစ်ခုဖြစ်သလို။ ဂိုင်းပြီး develop လုပ် contribute လုပ်ရှာတဲ့ python မှာ ဆိုင်ရာဆိုင်ရာ domain (eg, web, AI, machine learning, app) တွေ အသွေး library တွေလဲ ပေါ်ပါတယ်။



2.2 Python Installation

<https://www.python.org/downloads/> ကနေပြီးတော့ python ကိုဒေါင်းပြီး window installer ဖိုင်ကို install လုပ်ပါ။ Linux သုံးတဲ့ သူတွေကတော့ terminal ကနေ တစ်ခါတည်း တန်းထည့်လိုက်လို့ ရပါတယ်။

Ubuntu/Debian မှာဆို

```
#sudo apt-apt install python
#sudo apt-apt install python3
```

<http://crossnetmm.com/>

```
[root@localhost]~# apt-get install python python3
Reading package lists... Done
Building dependency tree
Reading state information... Done
python is already the newest version (2.7.15-4).
python3 is already the newest version (3.7.2-1).
0 upgraded, 0 newly installed, 0 to remove and 83 not upgraded.
[root@localhost]~#
```

Or from repository ## `sudo add-apt-repository ppa:deadsnakes/ppa`

Or use the Source file ##

```
# For apt-based systems (like Debian, Ubuntu, and Mint)
```

```
$ sudo apt-get install -y make build-essential libssl-dev zlib1g-dev libbz2-dev libreadline-dev libsqlite3-dev wget curl llvm libncurses5-dev libncursesw5-dev xz-utils tk-dev
```

```
$ wget https://www.python.org/ftp/python/3.6.5/Python-3.6.5.tgz
```

```
$ tar xvf Python-3.6.5.taz
```

```
$ cd Python-3.6.5
```

```
$ ./configure --enable-optimizations --with-ensurepip=install
```

```
$ make -j 8
```

```
$ sudo make altinstall
```

```
$ python3.6 -V
```

Fedora/CentOS ນາຄື

<http://crossnetmm.com/>

```
#yum install python
```

For python3.x

```
$ sudo yum update
$ sudo yum install yum-utils
$ sudo yum install https://centos7.iuscommunity.org/ius-release.rpm
$ sudo yum install python36u
$ sudo yum install python36u-pip
```

ဒါမှ မဟုတ် တစ်ခြား **Linux** တွေအတွက် package manager ကနေ သွင်းမရဘူးဆိုရင် .. source code ကနေ တစ်ခါတည်း တန်းသွင်းလိုလဲ ရပါတယ်။

OpenSUSE

```
$ sudo zypper install -t pattern devel_C_C++
$ wget https://www.python.org/ftp/python/3.6.5/Python-3.6.5.tgz
$ tar xvf Python-3.6.5.tgz
$ cd Python-3.6.5
$ ./configure --enable-optimizations --with-ensurepip=install
$ make -j 8
$ sudo make altinstall
$ python3.6 -V
```

For CentOS like linux system

```
$ wget https://www.python.org/ftp/python/3.6.5/Python-3.6.5.tgz
# For yum-based systems (like CentOS)
$ sudo yum -y groupinstall development
$ sudo yum -y install zlib-devel
$ tar xvf Python-3.6.5.tgz
$ cd Python-3.6.5
$ ./configure --enable-optimizations --with-ensurepip=install
$ make -j 8
```

```
$ sudo make altinstall
```

```
$ python3.6 -V
```

For **Fedora** မှာဆို

```
$ sudo dnf install python36
```

For **Arch Linux** မှာဆိုရင်

```
$ packman -S python
```

2.3 What can we do with Python

Python မှာ က version 2.7 ရပ် 3.x ရှိတယ်ဆိုတောာ.. နှစ်မျိုး လုံးကို cover ဖြစ်အောင်တောာ ရေးသွားမယ်များ။ ဒါ Python နဲ့ ဘာတွေလုပ်လို့ရတယ်.. သူ့ nature/syntax ကဘယ်လို .. စသည် ဖြင့်ပေါ့ ငလ္လာခြာဉ်းရာရအောင်များ။

Python နဲ့ ဘာတွေလုပ်လို့ ရမလဲ ဆိုတောာ ..

1. Web development ([Flask & Django](#))
2. Machine learning ([scikit-learn & TensorFlow](#))
3. Data visualization ([Matplotlib](#))
4. Scripting ([Simple Syntax, Easy to Learn & Quick to write/test](#))
5. Game developments
6. Desktop applications
7. Embedded applications

စတာတွေလုပ်လို့ ရပါတယ်။ စိတ်ဝင်စားရင် လေ့လာခြာဉ်းပါ။ ခု ဒီစာအုပ်မှာတောာ Python Basic ခြောင်းပါမယ်။ ဤတွေ့ Network programming အတွက် လိုအပ်တဲ့ telnetlib, paramiko, နဲ့ တစ်ခြား additional တွေ ဖြစ်တဲ့ socket programming, Tk interface တွေအောင်းပါဝင်မှ ဖြစ်ပါတယ်။ netmiko နဲ့ ansible တွေ့ မပါသေးပါဘူး။

2.4 What python is?

Python is interpreted အရေး တုန်းကလဲ ပြောခဲ့တယ်။ ခုထပ်ပြီး ရှင်းချင်တာက သူကည် compiled language လို့ source code ကို compile လုပ်တာမဟုတ်ပဲ.. Code ကို execute ပြီးဆိုတာနဲ့ တိပိုင်နှင်း တစ်ချိန်တည်းမှာ စပြီး အလုပ်လုပ်ပါတယ်။ ဥပမာ.. ပြောရရင်..



```
spacex@CrossNet:~$ gedit HelloWorld.py
spacex@CrossNet:~$ python HelloWorld.py
Hello world
spacex@CrossNet:~$
```

```
print ('Hello world')
```

Text editor တစ်ခုမှာ python code ဖြစ်တဲ့ `print ('Hello world')` ကို `HelloWorld.py` ဆိုတဲ့ နာမည်နဲ့ သိမ်းထားလိုက်ပါတယ်။ အဲဒေါ် code ရှိနေတဲ့ file ကို python [file name] ဆိုပြီး execute လိုက်တာနဲ့ `Hello world` ဆိုတဲ့ result ကို ချက်ချင်းရပါတယ်။ C, C++ တို့လို compile တစ်ခါလုပ်ပြီးမှာ run စရာမလိုပါဘူး။

Python is Interactive ဆိုတွော.. Interactive ဆိုတဲ့ အဓိပ္ပာယ်လေး တစ်ချက် ငြာည့် ဖြာည့်ရာ

interactive

adjective • UK /ɪn.tə'ræk.tɪv/ US /ɪn.tɪ'ræk.tɪv/

★ B2 An interactive system or computer program is designed to involve the user in the exchange of information:

ရအောင်။

ဘယ်လိုမိုးလဲဆိုတာက ဥပမာဏေးတစ်ခုနဲ့ ရှင်းပြုမယ်..

Windows မှာဆို command prompt ၊ Linux မှာဆို terminal မှာ python ဆိုပြီး ရှိတိုက်ပါပဲ။

```
0 upgraded, 0 newly installed, 0 to remove and 83 not upgraded.
[root@localhost] ~
#python
Python 2.7.16rc1 (default, Feb 18 2019, 11:05:09)
[GCC 8.2.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> print ('Hello world')
Hello world
>>>
```

အဲလို ရှိတိုက်တာနဲ့ `>>>` (Greater than Sign သုံးခု) ဆိုတဲ့ သက်တလေးကို မြင်ရပါလိမ့်မယ်။ အဲ `>>>` ကို **chevron prompt** လို့ ခေါ်တယ်။ သူက command line shell လိုပါပဲ.. user ကနေပြီးတွော input/command တစ်ခုရှိတိုက်တာနဲ့ result/feedback ချက်ချင်းပြန်ရပါတယ်။ ခုအပေါ် `>>>` လေးကိုက user

ကနေ လာပြီး တစ်ခု command/code တစ်ခု ပင်လာမှာ ကို တောင့်နေတဲ့ သဘောမျိုးပါ။ ဒါ print ('Hello world') လို့ရှိရှိလိုက်တာနဲ့။ Hello world ဆိုပြီး result တန်းရပါတယ်။ အဲလိုမျိုး အလုပ် လုပ်ပုံကို interactive လိုက်ပါတယ်။

2.4.1 Python Interactive mode & Script Mode

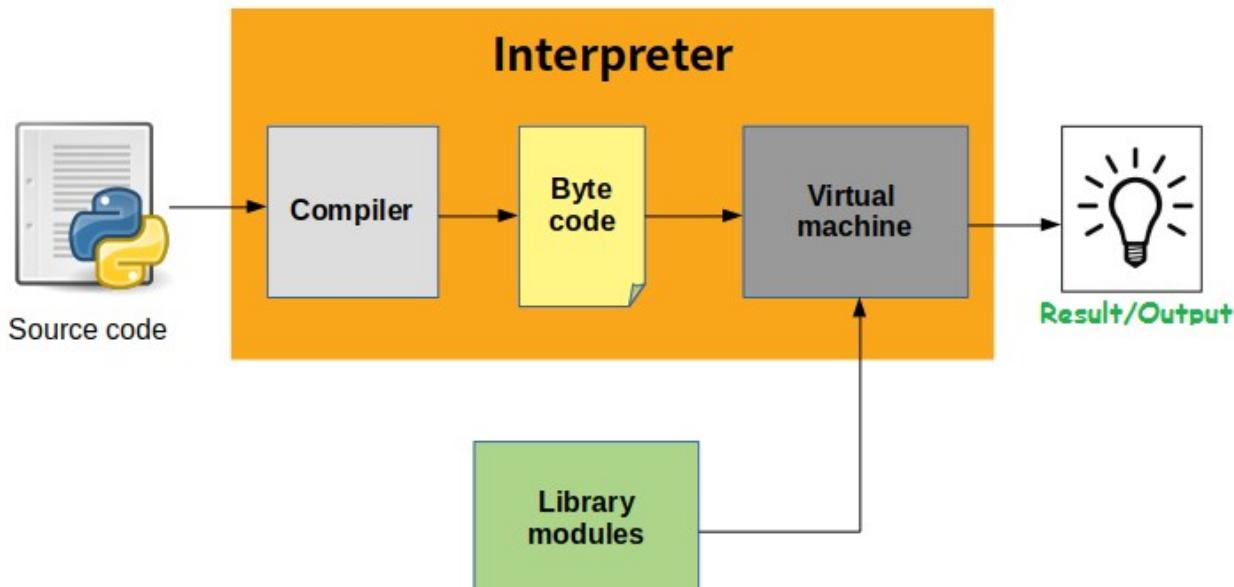
ပြောလက်စနဲ့ တစ်ခု သိမေးခွန်တာက python ကို ပုံစံနှစ်မျိုး နဲ့ ရေးရာတယ်။

1. Interactive mode

သူက `>>>` မှာ တိုက်ရှိက် ရှိရှိလိုက်တဲ့ တစ်ကြောင်းချင်းကို responds ပြန်ပါတယ်။ သူသည် device ပဲ့ RAM (memory) မှာ တင်ပြီး အလုပ်လုပ်ပါတယ်။ ပိတ်လိုက်ရင် ဘာမှမရှာန်တော့ပဲ ဖောက်သွားပါလိမ့်မယ်။ variable တွေ method တွေ စမ်းတဲ့ အချိန်မှာ အသုံးပြုတာမှားပါတယ်။

2. Script mode

သူရာကတွေ့ဘဲ text editor တစ်ခုနဲ့ code တွေ ရေးပြီးတော့ file တစ်ခု အနေနဲ့ saveပြီး .. အဲ ဖိုင်ကိုမှ python အနေနဲ့ execute လုပ်တာပါ။ save ထားတာမို့ file သည် မဖြတ်မရှင်း ရှိနေပါတယ်။



Python is Object-Oriented Programming (OOP) တစ်ခုဖြစ်ပါတယ်။ ခုမှ စလေလာမယ် Networker တွေအနေနဲ့ oop လိုပဲ သိတားပါပြီး.. နောင်ရေးရင်နဲ့ OOP ဆိုတာကို သိလာမှာပါ။ ပိတ်မပူပါနဲ့။

Python works on different platforms သူသည် OS ပေါ်ကို မမြန်ပါဘူး.. ဘယ် OS နေနေ python ရှိရင် အလုပ် လုပ်ပါတယ်။ (Windows, Mac, android, Linux, Raspberry Pi, etc)

Python has simple syntax.. Python ရဲ့ syntax တည်ဆောက်ပုံ ပုံစံ သည် English စာလိပ်များ.. စာပိုင်တွေ စာသား အနေနဲ့ အလုပ်လုပ်ပါတယ်။ ဂြိုဟ်လိုက်ရင် ရှိုးရှင်းတာကို တွေ့ရမှာပါ။ ပြီးတော့ သူသည် scripting language ဖြစ်တဲ့ အတွက် program ရေးတဲ့ သူတွေကို တစ်ခြား language တွေထက် code နဲ့ ငဲ့လုပ်ရမည်။

2.5 Very first Program “Hello world” and its syntax, key-words



`print("Hello, world!")`

ထုံးစံအတိုင်း programming Language တစ်ခု စလေ့လာရင် output ကို Hello, world! ထုတ်တာ ကတော့ အရေ့မှာ Example အနေနဲ့ အကြိမ်ကြိမ် သုံးပြုခဲ့တာဆိုတော့ ..

`>>> print ('Hello, world!')` ဆိုတဲ့ Hello, world! ဆိုတဲ့ စာသားကို **output result** အနေနဲ့ ထုတ်ပြတဲ့ python code ကိုတော့ ရင်းနှီးနေရာမှာပါ။ ;)

ဒ္ဓာတော့ ဟုတ်ပြီ .. print ဆိုတဲ့ keyword နောက်က ကိုယ်ထည့်ချင်တဲ့ စာသားထည်ရင် output ထွက်တာတော့ သိပြီ.. ဒ္ဓာတော့ အောက်က command တွေရှိက်ရင်ကော်? Outpt ထွက်မလား..

`printf("Hello, world!")`

***** C *****

```
>>> print ('Hello, world')
Hello, world
>>>
>>> printf("Hello, World!");
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'printf' is not defined
>>>
>>>
```

`system.out.println("Hello, world!");`

***** Java *****

```
>>>
>>> system.out.println("Hello, World!");
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'system' is not defined
>>>
```

```
echo "Hello World"
```

*** Linux Bash shell ***

```
>>>
>>> echo "Hello World"
      File "<stdin>", line 1
          echo "Hello World"
^
SyntaxError: invalid syntax
>>>
```

အဲလိုရှိက်လိုက်ရင် **invalid syntax, syntax error , Name [...] is not defined** သာ
ည် **ဖြင့်** error တွေ တက်နေတာကို တွေ့ရမှာပါ။ အဲဆို လူတွေ စကား ပြောသလို ခိုင်း ဗြာဉ်းရင်..
ဘယ်လို တုန်းပြန်မလဲ?..

```
>>>
>>> please print out "Hello, world!"
      File "<stdin>", line 1
          please print
^
SyntaxError: invalid syntax
>>>
```

တွေ့တဲ့အတိုင်းပါပဲ.. Python ရဲ့ syntax မဟုတ်ရင် ဘူး။ key-word မဟုတ်ရင် လက်မခံပါဘူး.. မြင်
သာတဲ့ ဥပမာ လေး တစ်ခုနဲ့.. ဗြာဉ်း ဗြာဉ်းရာရအောင်။



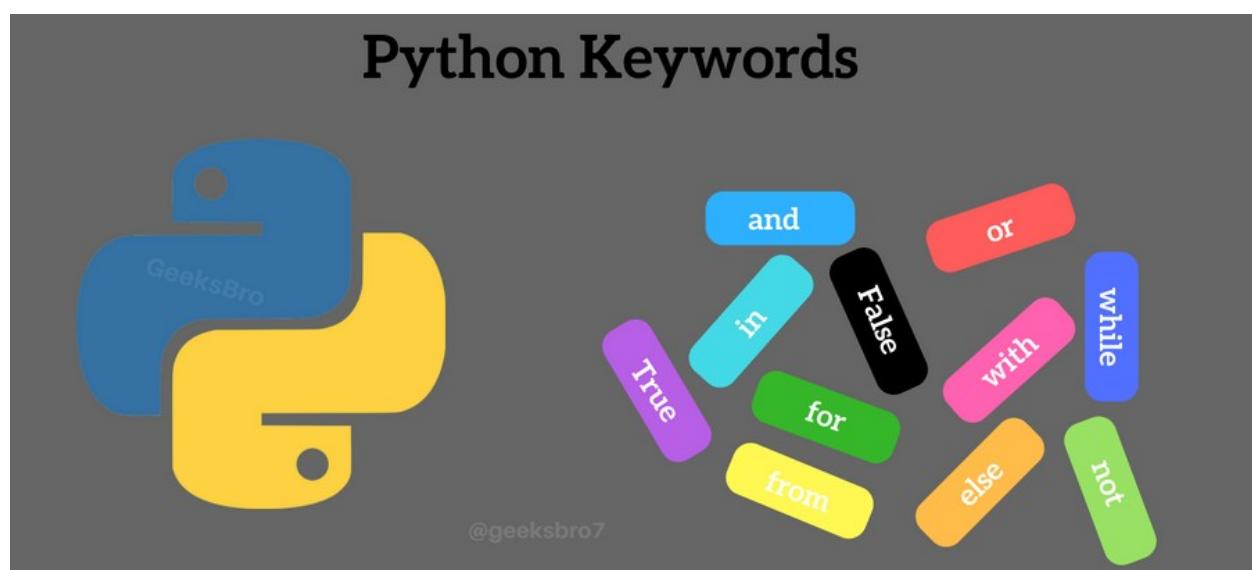
ဂွဲန်လောက်တို့တွေ အိမ်မှာ ခွေးလေးတွေ ပြောင်လေးတွေ မွေးတက်ရာတယ်ဟုတ် ? အိမ်မှာ မွေးထား
တဲ့ ခွေးလေးပဲ ဖြစ်ဖြစ်ပေါ့.. နာမည်ခေါ်ရင်လာမယ်.. ထိုင်ဆိုထိုင် ထဆိုထ.. အဲလောက်တွော နားလည်သိတက်
ရာတယ်။ အဲတက်ရုပြီး.. လူတစ်ယောက်လိမ့်း စကားတွေ ထိုင်ပြောရင်.. ဒီကောင်လေးတွေ နားလည်မှာ မဟု
တိပါဘူး.. အသည်းကွဲလို ရင်ဖွင့်လဲ.. သခင်မှုက်နာ အကောင်း ဘေးကနေ ဂိုင်ပြီးအဖော် လုပ်ပေးနေတာပဲ ရှိတ

ယ်။ ဘီယာသောက်လိုက် ဆိုပြီး ဘီယာဘူး နှီလာမှာ မဟုတ်သလို.. ခရီး သွားလိုက် ဆိုပြီးတော့လဲ အကြံ့ ပေးမှာမဟုတ်ပါဘူး.. သူမှနားမလည်တား.. :P

ထိုနည်းတူ programming language သည်လဲ.. သူမှနားလည်တဲ့ ပုံစံ syntax တွေ key-word တွေ ဆိုတာ ရှိပါတယ်။ Programming Language မတူရင် syntax အနည်းနဲ့အများ ပုံစံကဲ့ နိုင်သလို.. Key-word တွေကဲလဲ တူမှာ မဟုတ်ပါဘူး.. တရုတ်ကြီးကို japan စကားသွားပြောရင် တရုတ်ကြီးက နားလည်မှာ မဟုတ်သလို.. Python မှာ java/C code တွေ လာရေးရင်လဲ python က နားလည်မှာ မဟုတ်ပါဘူး.. ပြောရရင် ရွှေနှင့် network device တွေမှာဆိုလဲ .. cisco မှာ ရှိက်တဲ့ command ကို huawei မှာ ရှိက်ရင် ရမှာ မဟုတ်သလို .. Juniper command တွေကိုလဲ mikrotik တို့ nokia တို့မှာ သွားရှိက်ရင် ရမှာ မဟုတ်သလို ပါပဲ။

2.6 Python Key-words

Python ကို လေ့လာမယ်ဆိုတွော သူ့ Syntax တွေ key-word တွေနဲ့ ရင်းနှီးပြီး သိနေမှ ရမယ်။ အဲတွော ပထမဆုံး ဘာ key-word တွေ ရှိလဲ ဆိုတာကို လေ့လာဖြည့်ရာ ရအောင်..



Python prompt `>>> help()` ၏လိုက်ပါ။ အဲနောက် keywords လို့ ရှိက်လိုက်ရင် python မှာ သူ့အတွက် ရှိနေတဲ့ keywords တွေကို ပြ ပေးပါလိမ့်မယ်။ print ဆိုတဲ့ keyword ကိုရင်းနှီး ပြီးသားပါ။ if ဆို condition စစ်ဘာ while ဆို while loop, for ဆို for looping အဆွဲ့.. စသည်ဖြင့်ပေါ့.. နောင် လာမထဲ program တွေက keyword တွေ သုံးရင်းနဲ့ သိလာမှာပါ။ ခု keyword ဘယ်လို ဖြေည့်ရတယ်။။။ ဘယ်ကောင် အွော keyword တွေ ဖြစ်တယ်ဆိုတာ သိရင် လုံလောက်ပြီးလေ။။။



```
help> "keywords"

Here is a list of the Python keywords. Enter any keyword to get more help.

and          elif          if           print
as           else          import       raise
assert       except        in            return
break        exec          is             try
class        finally      lambda      while
continue    for           not          with
def          from          or            yield
del          global        pass

help> █
```

ကြံးတုန်း တစ်လက်စတည်းပြောချင်တာက.. အဲ help နောက်ကနေ.. Keywords ဆိတ်တင် မဟုတ်ဘူး.. ကို ယိသုံးချင်တဲ့ library တွေရှိရင်.. အဲကောင်ရဲ့ info တွေ description တွေသုံးချင်ရင် help> [library name] ထည့်ပေးယုံပါပဲ။

keywords တွေသည် reverse လုပ်ထားတာပါ။ အဲတာကြောင့် keyword တွေကို သူတို့နဲ့ မသက် ဆိုင်တဲ့ နေရာ .. ပြောရင် variable လိုကောင်တွေ နေရာမှာ သုံးလို့ မရပါဘူး.. print keyword သည် Output ထုတ်ဖော် အတွက်.. if ဆိုတဲ့ keyword သည် condition ကို စစ်ဖို့အတွက် စသည်ဖြင့်ပေါ့နော့ ဖြော်ပြီး သတ်မှတ်ထားပြီးသားပါ။

```
>>>
>>> print = 10
      File "<stdin>", line 1
          print = 10
              ^
SyntaxError: invalid syntax
>>>
```

```
>>>
>>> if = "Silent Dimension"
      File "<stdin>", line 1
          if = "Silent Dimension"
              ^
SyntaxError: invalid syntax
>>>
```

Print ထုတ်ပဲ့ နေရာမှာ ခုမှာ စလေ့လာတဲ့ သူတွေဆို python2.7 နဲ့ Python3.x မှာဆို.. နည်းနည်း သတိထားမှုလို့ တာရှိပါတယ်။ အဲတာကတော့..

Python2.7 မှာ က Print ထုတ်ချင်တဲ့ ကောင်ကို **print** ဆိုတဲ့ keyword နောက်ကနေ '**(single quote)** OR "**(double quote)**" နစ်ခြားမှာပဲ ထားထား '**[output]**' .. လက်သည်းကွင်းရပ်() quotes '' ရယ်နဲ့ အဖွဲ့



င့်အပိုတဲးထားထား (' [output] ') .. အလုပ်လုပ်ပါတယ် .. အောက်ကပုံကို ပြောလိုက်ရင် ရှင်းသွားပါလိမ့်မယ်။ နှစ်နည်းလုံးနဲ့ လုပ် လုပ်နေတာကို တွေ့ရပါလိမ့်မယ်။

```
spacex@ubuntu:~$ python
Python 2.7.12 (default, Nov 12 2018, 14:36:49)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> print ('Hello, world!')
Hello, world!
>>>
>>> print 'Hello, world!'
Hello, world!
>>>
>>> exit()
spacex@ubuntu:~$
```

Python3.x မှာတော့ Print ထုတ်ချင်တဲ့ ကောင်ကို **print** ဆိုတဲ့ keyword နောက်ကနေ လက်သည်းကွင်းရယ်() quotes '' ရယ်နဲ့ အဖွင့် အပိုတဲ့ တိုကို ထားရပါမယ်။ (' [output] ') ဆိုတဲ့ ပုံစံနဲ့ပဲ .. အလုပ်လုပ်ပါတယ် single/double quotes တွေ့နည်းပဲ ဆိုအလုပ်မလုပ်ပါဘူး။ .. အောက်ကပုံကို ပြောလိုက်ရင် ရှင်းသွားပါလိမ့်မယ်။

' [output] '	(' [output] ')	⇒	Python2.7 မှာ အကုန်လုပ် လုပ်ပါတယ်။
(' [output] ')	⇒	ဒီတစ်မိုးပဲ Python 3.x မှာ အလုပ်လုပ်ပါတယ်။	

```
spacex@ubuntu:~$ python3
Python 3.5.2 (default, Nov 12 2018, 13:43:14)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> print ('Hello, world!')
Hello, world!
>>>
>>> print 'Hello, world!'
  File "<stdin>", line 1
    print 'Hello, world!'
               ^
SyntaxError: Missing parentheses in call to 'print'
>>>
```

2.7 single vs double quotes

<https://docs.python.org/2.0/ref/strings.html>

In plain English: String literals can be enclosed in matching single quotes (') or double quotes (""). They can also be enclosed in matching groups of three single or double quotes

အပေါ်က လင့်မှာ ရှိတဲ့ document ကိုဖတ် ပြေဆုံးလိုက်ပါ။ Python မှာက single quote နဲ့ Double quote သည် အတူတူပါပဲ။ ကွားခြင်း မရှိပါဘူး။

ဟုတ်ပြီ Hello, world! လဲ ထုတ်တက်ပြီး။ Keyword အကြောင်းလဲ သိပြုဆိုတော့ variable တွေ အကြောင်း ဆက်ရှုစုံ။

2.8 Variable

Variable ဆိုတာက စွဲနှင့်တို့တွေရဲ့ program မှာလိုအပ်တဲ့ data တွေထည့်ဖို့ အတွက်ပါ။ programming မှာဆို variable တစ်ခု ပြောညာလိုက်တာနဲ့ memory မှာ နေရာ တစ်နေရာယူလိုက်ပါတယ်။ အဲ memory address ရှိတဲ့ နေရာမှာ.. စွဲနှင့်တို့ ထည့်ခွင့်တဲ့ .. variable အဆွဲ့ data တွေကို သွားပြီ သိမ်းဆည်းပါတယ်။

Python မှာက တစ်ခြား Programming Language တွေနဲ့ မတူတာက သူသည် variable data type ကိုပြောညာပေး စရာမလိုတာပါပဲ။ Example အနေနဲ့ ပြောရရင် int Num ဆိုရင် Num ဆိုတဲ့ variable သည် integer Type ဖြစ်ပါတယ် ဆိုပြီး ပြောညာပေးထားတာပါ။ python မှာဆိုရင်တော့ type ကို ပြောပြီ ပေးစရာ မလိုပါဘူး။ သူသည် variable ထဲကို ထည့်တဲ့ data ပေါ် မှတည်ပြီး type ကို ခွဲဗြားသိပါတယ်။ `Num = "MgMg"` လို့တွေ့ရင် Python သည် Num variable သည် String value တစ်ခု ဖြစ်ကြောင်း တန်းသိပါတယ်။ အဲလိုပဲ `Num = 5` လို့တွေ့လိုက်တာတဲ့ Num သည် Integer Type ဖြစ်တယ်လို့ တန်းပြီး သတ်မှတ်လိုက်ပါတယ်။ အောက်က ပုံလေးတွေကို တစ်ခုကိုပြောလိုက်ရင် ပိုနားလည်သွားပါလိမ့်မယ်။

```
>>> Name = 'Mg Mg'
>>> type(Name)
<type 'str'>
>>>
>>> print Name
Mg Mg
>>>
```

```
>>> Age = 25
>>> type(Age)
<type 'int'>
>>>
>>> print Age
25
>>>
```

ခုပုံမှာဆို `Name = 'Mg Mg' / Age = 25` ဆိုပြီး `Name` နဲ့ `Age` ဆိုတဲ့ variable ထဲကို '`Mg Mg'` ရပ် `25` ရပ် ဆိုတဲ့ data တွေ ထည့်လိုက်ပါတယ်။ အပေါ်မှာတုန်းက python သည် variable ထဲကို ထည့်လိုက်တဲ့ data ပေါ် မှတည်ပြီး ခွဲဗြားသိတယ်လို့ ပြောခဲ့တယ်နော်။ တစ်ကယ် သိမသိ သိချင်ရင် `type()` ဆိုတဲ့ Function ကို အသုံးပြုလို ရပါတယ်။ `type(Name)` ဆိုပြီးရှိက်လိုက်ရင် Name ဆိုတဲ့ variable ရဲ့ `type` ကို output ထုတ်ပါလို့ Python ကို ပြောလိုက်တာပါ။ အဲတာရာ အဲName variable သည် ဘာ Data Type အ



မျိုးအစား ဖြစ်တယ်ဆိုတာကို result ထွက်လာပါတယ်။ အဲလိုပဲ Age သည့် ဘယ်လို type မျိုးလဲ သိချင်ရင် Type(Age) ဆိုပြီး ဗြာည့်လိုက်တာနဲ့ integer ဖြစ်တယ်ဆိုတာကို တွေ့ရမှာပါ။ Variable တွေထဲက data/value ထုတ်ချင်ရင်တွေ့ print ဆိုတဲ့ keyword နောက်ကာနဲ့ variable ကို ထည့်ပေးလိုက်ယူ ပါပဲ။ ဒါ အပေါ်က ပုံမှာက interactive mode နဲ့ ရေးသွားတာ .. အဲလိုမဟုတ်ပဲနဲ့ script တစ်ခု အနေနဲ့ ရေးဗြာည့်ရာရအောင်။

```
Name = "Mg Mg"      # String
Age = 25             # Integer
Grade = 88.9          # Float
Result = True         # Boolean

print ("The data type of ")
print (Name)
print ('within Name variable is ')
print (type(Name))

print ("\nThe data type of ")
print (Age)
print ('within Age variable is ')
print (type(Age))

print ("\nThe data type of ")
print (Grade)
print ('within Grade variable is ')
print (type(Grade))

print ("\nThe data type of ")
print (Result)
print ('within Result variable is ')
print (type(Result))
```

နောက်တဲ့ လိုက်တဲ့ ကောင်စွာတို့ Python က
ignore လုပ်ပါတယ်၊ ဗြာချင်တာက run ထဲ အခါ
ထည့်ပြုတော့ မရပါ ပါဘူး

/n ဆိုတာတဲ့ ကျမေတ္တာ new line ပါ။
နောက်တစ်ဦး ဆင်းဆင်းမယ်လို့ ဆိုလိုတာပါ။

type() တော့ သိတဲ့ အတိုင်း ဘယ်လို data type လဲ
ဆိုတာတဲ့ ခြား ထုတ်ပေးပါတယ်။

အဲ order သည့် ဉာဏ်နဲ့ ဘယ်လို ဘားတာပါ။
ထည့်ချင်တဲ့ data တို့ ဉာဏ်မှာထား.. ပြီး ရင် = စို
ခရာ.. ဘယ်ဘက်မှာတော်တော် assign လုပ်မယ် variable တို့
ဘားပါ။

အပေါ်က code တွေကို text editor တစ်ခု နဲ့ ရေးပြီး နာမည်တစ်ခု(variable.py) ပေးပြီး save လို
က်ပါ။ အဲနောက် terminal မှာ python variable.py ဆိုပြီး run လိုက်ပါ။



```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\ekhaphy\Documents\atom\Python> python variable.py
The data type of
Mg Mg
within Name variable is
<type 'str'>

The data type of
25
```

အဲလို run လိုရသလို atom, Sublime text2/3, Pycharm တိုလို IDE တွေနဲ့ run လို ရပါတယ်။

The screenshot shows a Python development environment with two code files and their execution results.

serial.py:

```
1 Name = "Mg Mg" # String
2 Age = 25 # Integer
3 Grade = 88.9 # Float
4 Result = True # Boolean
5
6 print ("The data type of ")
7 print (Name)
8 print ('within Name variable is ')
9 print (type(Name))
10
11 print ("\nThe data type of ")
12 print (Age)
13 print ('within Age variable is ')
14 print (type(Age))
```

variable.py:

```
C:\Python27\python.exe
The data type of
Mg Mg
within Name variable is
<class 'str'>
The data type of
25
within Age variable is
<class 'int'>
The data type of
88.9
within Grade variable is
<class 'float'>
The data type of
True
within Result variable is
<class 'bool'>
```

Python မှာ standard variable အမျိုးအစား ၅မျိုး ရှိပါတယ်။

- Numbers
- String
- List
- Tuple
- Dictionary

2.8.1 Numbers

Numbers data type သည် Numeric တန်ဖိုးတွေကို သိမ်းဆည်းဖို့အတွက်ပါ။ Number Type ကို အမျိုးအစား 4 ခု ခွဲခြားသံမှတ်လို့ ရပါတယ်။

1. int (ကိန်းပြည့်)
2. float (ဒေသမကိန်း)
3. long
4. complex

int	long	float	complex
10	51924361L	0.0	3.14j
100	-0x19323L	15.20	45.j
-786	0122L	-21.9	9.322e-36j
080	0xDEFAFBCECBDAEBCFBAE1	32.3+e18	.876j
-0490	535633629843L	-90.	-.6545+0J
-0x260	-052318172735L	-32.54e100	3e+26J
0x69	-4721885298529L	70.2-E12	4.53e-7j

2.8.2 String

String မှာရာတွေ၊ alphabet တွေ သိမ်းထားပေးတဲ့ variable ပဲ ဖြစ်ပါတယ်။ string variable မှာ သုံးလို့ရတဲ့ ဥပမာလေးတွေကို ပြေည့် ပြောည့်ရရအင်ဖြာ ..

```
spacex@ubuntu:~$ python
Python 2.7.12 (default, Nov 12 2018, 14:36:49)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> string = 'Hello, world!'
      string ഓഫ് Hello, world! ഫോഡ് data ഓഫ് അട്ടണം
>>> print string
      print string ഫീഡ്:നോട് string ഓട്ടു data ഓഫ് അടിസ്ഥാനം
      string[0:5] ഫീഡ്:നോട് string ഓട്ടു alphabet character എന്നോ
      0,1,2,... ഫീഡ്:നോട്: 0 ഒരു 4 വരെത്തു കൊടുവേംതോട് അടിസ്ഥാനം
      string[6:] ഫീഡ്:നോട് character എന്നോ അനുക്രമാവലിക്കൽ അടിസ്ഥാനം
      string * 3 ഫീഡ്:നോട് string ഓട്ടു data ഓഫ് ഓഫ്:സൈസ് അടിസ്ഥാനം
>>> print string * 3
      Hello, world!Hello, world!Hello, world!
      string ഓഫ് അപ്പേജ്: സൈസ്:നോട്
>>> print string + " => We can Concatenated the string"
      Hello, world! => We can Concatenated the string
      string concatenated ഫീഡ്:നോട്
      സൈസ്:നോട്
```

String Concatenate

```
Hello, world! => We can Concatenated the string
>>>
>>> x = "Hello "
>>> y = 'world'
>>> print x + y
Hello world
>>> z = x + y
>>> print z
Hello world
>>>
```

print x + y ➜ x နဲ့ y အတေသာက်စွမ်းပါ၏ ပုံစံ

z = x + y ➜ ပုံစံ Hello နဲ့ world ဆိတ် string တွင်ဖြစ်ပါ၏ z အတေသာက်စွမ်းပါ၏ ပုံစံ

အဲဒေသ print z ပါ၏ ပုံစံ z အတေသာက်စွမ်းပါ၏ ပုံစံ

အကယ်လို့ variable type မတူညီစဲ့ ကောင်တွေပေါင်းမယ်ဆို + တစ်ခု တည်းသုံးပြီး ပေါင်းလို့ မရပါဘူး error တက်ပါလိမ့်မယ်။

```
>>> value = 'Price = '
>>> x = 10
>>> print (value + x)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate str (not "int") to str
>>>
>>> print (value + str(x))
Price = 10
>>>
```

Price = එහිදී ගොන්යෝජිත තුළ value එහිදී variable වනු string type ප්‍රතිඵිතයි। 10 යනුයා තුළ x වනු integer variable ප්‍රතිඵිතයි। ඇදී type මත්‍යාගේ පිරිස්ථිතයා නැත්තු නොමැතියි නැත්තු නොමැතියි නැත්තු නොමැතියි නැත්තු නොමැතියි

- int() # Change to Integer
 - str() # Change to String
 - float() # Change to float ടീംഗ്രാഫിക്ക്

အခု အပေါ်က ပုံမှာဆို x သည် integer ဖြစ်တဲ့အတွက် value ဆိုတဲ့ string type variable နဲ့ပေါင်းမရင်တာ မျိုးတူအောင် လုပ်မှ ရမှာဖြစ်ပါတယ်။ ဆိုတော့ကာ str() ဆိုတဲ့ function ကို သုံးပြီးတော့ x variable ကို string အေနနဲ့ ရမှာ print ထုတ်လို့ရမှာပါ။

```
>>> x = '5'
>>> y = 10
>>>
>>> type(x)
<type 'str'>
>>> type(y)
<type 'int'>
>>>
>>> z = int(x)
>>> type(x)
<type 'str'>
>>> type(z)
<type 'int'>
>>>
```

ဒါပေမယ့် နောက်တစ်ခါခု ထပ်သို့စေခဲ့တာက str() int() float() တို့သည် variable ထဲက တန်ဖိုး ကို ပြောင်းလဲလိုက်တာမဟုတ်ပဲ.. အဲထဲမှာ ရှိတဲ့ တန်ဖိုးကို return ပြန်ပေးတာပါ။ programming နဲ့လေ့လာဖူးတဲ့ သူတွေကတော့ သိရှာတယ်။ Return ပြန်တယ်ဆိုတာကို.. သူက ခု အပ်က ဥပမာ မှာဆိုရင် $x = '5'$ ဆိုပြီးတော့ x ထဲ မှာ 5 ကို '(single quotes)' သုံးပြီးတော့ string အေနနဲ့။ ထည့်ထားတာပါ။ အဲတာကို int(x) ဆိုပြီး လုပ်လိုက်ရင် result အေနနဲ့ သာ integer 5 ထွက်လာပြီးတော့ x ထဲမှာတော့ string '5' အေနနဲ့ပဲ ရှိနေတာပါ။ အဲတာကြောင့် မြင်သာအောင်.. int(x) လို့ လုပ်လိုက်ရင် return ပြန်လာမယ့် result ကို = (assign) ကိုသုံးပြီးတော့ z ထဲကို ထည့်လိုက်ပါတယ်။ z သည် integer value 5 ကို ထည့်ထားတဲ့ variable ပို type() နဲ့ရှာည့်ရင် int အေနနဲ့ မြင်ရမှာ ဖြစ်ပြီး x ကိုတော့ str အေနနဲ့ မြင်ရမှာ ပဲ ဖြစ်ပါတယ်။

2.8.2.1 int() str() float()

int() str() float() တို့ကို သုံးပြီးတော့ exercise လေးတွေ လုပ်ကြည့်ရှုရအောင်များ။

int() function

```
x = int(1) # x will be 1
y = int(2.8) # y will be 2
z = int("3") # z will be 3
```

float() function

```
x = float(1) # x will be 1.0
y = float(2.8) # y will be 2.8
```

```
z = float("3") # z will be 3.0
w = float("4.2") # w will be 4.2
```

str() function

```
x = str("s1") # x will be 's1'
y = str(2) # y will be '2'
z = str(3.0) # z will be '3.0'
```

2.8.3 Python Lists

ခုခြောမယ် list ရယ် ပြီးတော့ tuple တို့သည် array ရဲ့အလုပ် လုပ်ပုံနဲ့ ဆင်ပါတယ်။ list/tuple နဲ့ array ရဲ့ အမိက ကွာခြားချက်ကတော့ array တစ်ခု သည် တူညီတဲ့ data type တွေကို ပဲ သိမ်းဆည်းနိုင်တာ ပါ။ list/tuple မှာရာတော့ မတူညီတဲ့ data type (eg., Str, int) တွေကို အတူယုဉ် တွေပြီး သိမ်းဆည်းနိုင် တာပါပဲ။ ပြီးတော့ Python မှာက array သည် build-in မပါ ပါဘူး.. Array ကိုမှ သုံးချင်တယ်ဆိုရင်တော့ import array ဆိုပြီး array module ကို import လုပ်ပေးမှ သုံးလို့ရပါမယ်။

Lists ဆိုတာက python မှာ ပြောရရင် item တွေကို sequence အဖောက်၊ သိမ်းဆည်းပေးတဲ့ ကောင် ပါ။ ဥပမာ ပြောရရင် fruit ဆိုပြီး အသီးတွေကို စာရင်း လုပ်မယ်ဆိုရပါတော့.. နာမည် တစ်ခုကို variable တစ်ခုနဲ့ သိမ်းမယ့် အစား list ကိုသုံးလို့ရပါတယ်။ သူသည် ဒေါင်ကွင်းကိုသုံးပါတယ်။ item တစ်ခုနဲ့ တစ်ခုကို , (commas) နဲ့ပိုင်းခြားထားပေးပါတယ်။

```
Fruit = ['apple','banana','avocado','blackberry','muskmelon','orange']
```

ခုခြောမယ် fruit ဆိုတဲ့ list ထဲမှာ apple တို့ banana တို့ စတဲ့ ကောင်တွေကို item တွေလို့ ခေါ်ပြီး နံပါတ်တွေ တပ်ပြီး သိမ်းထားပါတယ်။ print Fruit ဆိုပြီး ထုတ်လိုက်ရင်တော့ Fruit ထဲမှာ ရှိတဲ့ item တွေအကုန် ထွက် လာပါလိမ့်မယ်။

```
>>> Fruit = ['apple','banana','avocado','blackberry','muskmelon','orange']
>>>
>>> print Fruit
['apple', 'banana', 'avocado', 'blackberry', 'muskmelon', 'orange']
>>>
```

Banana ဆိုတဲ့ item ကိုပဲ ယူချင်တယ်ဆိုရင် သူ့ရဲ့ location ကို [] ထဲထည့်ပေးယုံပါပဲ။ list မှာ sequence သည် 0 က စတာမို့ apple=1, banana=1 ဖြစ်ပါတယ်။

```
>>> print Fruit[1]
banana
>>>
>>> Fruit = ['apple','banana','avocado','blackberry','muskmelon','orange']
>>> [0] [1] [2] [3] [4] [5]
```

ကိုယ်က apple,banana,avocado ပဲ လိုချင်တာဆိုရင် 0,1,2 နေရာကကောင်တွေကို ယူမှာဖြစ်လို print Fruit[0:3] လို့ ရေးလိုက်ရင် 0 to 2 မှာ ရှိတဲ့ Item တွေ့ေးထုတ်ပေးသွားမှာပဲ ဖြစ်ပါတယ်။ blackberry နောက်မှာ ရှိနေစဲ့ item တွေကို အကုန် ထုတ်ချင်တယ် ဆိုရင်တွော print Fruit[3:] လို့ ရေးပေးလိုက်ရင် sequence 3 ကနေစပီး နောက်က ကောင်တွေအကုန် ကို ထုတ်ပေးသွားမှာပဲ ဖြစ်ပါတယ်။

```
>>> print Fruit[0:3]
['apple', 'banana', 'avocado']
>>>
>>> print Fruit[3:]
['blackberry', 'muskmelon', 'orange']
>>>
```

အမျိုးအစားတူတာတွေပဲ သိမ်းလို့ ရတာ မဟုတ်ပါဘူး.. မတူတာတွေကိုလဲ ပေါင်းပြီး သိမ်းလို့ရပါတယ်။ Random_list ဆိုပြီး string တွေနဲ့ integer တွေကို ရောပြီး သိမ်းလိုက်ပါတယ်။

```
>>> random_list = [1,'MgMg',2,'AungAung',9,'Hello']
>>>
```

print ထုတ်တာတွေဘာတွေကတော့ အပေါ်က Fruit တုန်းကထုတ်ခဲ့ သလို ထုတ်လိုက်ယံပါဝဲ။ list တွေကို + လေးသုံးပြီး ပေါင်းလို့ ရပါတယ်။ print ထုတ်ထဲ နေရာမှာ print Fruit + Random_list ဆိုပြီး ထဲလို့ရသလို.. ပေါင်းလို့ရလာတဲ့ result ကို print combine ဆိုတဲ့ထဲ ထည့်ပြီးမှ ပေါင်းဆိုပြီးလဲ ထုတ်လို့ရပါတယ်။

```
>>> combine = Fruit + random_list
>>> print combine
['apple', 'banana', 'avocado', 'blackberry', 'muskmelon', 'orange', 1, 'MgMg', 2,
 , 'AungAung', 9, 'Hello']
>>>
>>> print Fruit + random_list
['apple', 'banana', 'avocado', 'blackberry', 'muskmelon', 'orange', 1, 'MgMg', 2,
 , 'AungAung', 9, 'Hello']
>>>
```

List အကြောင်း အကြောင်းဖြင့်တော့ သိသွားလောက်ပြီလို့ ယူဆပါတယ်။ အဲတော့ list ထဲကို data တွေထည့်တာ ထုတ်တာတွေ စမ်းကြေည့်ရာရအောင် ;-)



ပထမဆုံး `Marvel = ['Iron Man', 'Thor', 'Hulk']` ဆိုပြီး ထည့်လိုက်ပါတယ်။ list ထဲကို data စုစုပေါင်း ထည့်မယ်ဆို `append()` ဆိုတဲ့ Method ကို သုံးပါတယ်။ `Marvel.append('Spidey')` #append at END ဆိုပြီး ထည့်လိုက်တဲ့ အခါမှာ spidy သည် Marvel list ရဲ့ နောက်ဆုံးမှာ ပင်သွားပါတယ်။ `extend()` ဆိုတဲ့ ကောင်ရာတွေ၊ list တစ်ခု ထပ်ထည့်တဲ့ သဘောပါ။ `Marvel.extend(['Wolverine', 'Black Bolt'])` #add other list at END ခုခိုရင် နောက်ဆုံးမှာ `['Wolverine', 'Black Bolt']` ဆိုတဲ့ list ကို ထည့်လိုက်ပါတယ်။ `print Marvel` ဆိုပြီး ထုတ်ဖြာည့်ပါ။

```
>>> Marvel = ['Iron Man', 'Thor', 'Hulk']
>>> Marvel.append('Spidey')      #append at END
>>> Marvel.extend(['Wolverine', 'Black Bolt'])  #add other list at END
>>> print Marvel
['Iron Man', 'Thor', 'Hulk', 'Spidey', 'Wolverine', 'Black Bolt']
>>>
```

အဲလို နောက်ကနေ ပင်သွားတာမျိုး မဟုတ်ပဲ.. ကိုယ်ထည့်ချင်တဲ့ Index နေရာမှာ ထားမယ် ဆို `insert()` ကိုသုံးပါ။ `Marvel.insert(1,'Captain America')` #insert at index 1 ဆိုပြီး Cap ကို 'Iron Man' နောက်က index 1 မှာထားလိုက်ပါ။ အဲနောက် Print ထုတ်ဖြာည့်ပါ။

```
>>> Marvel.insert(1,'Captain America')  #insert at index 1
>>> print Marvel
['Iron Man', 'Captain America', 'Thor', 'Hulk', 'Spidey', 'Wolverine', 'Black Bolt']
```

ပြန်ဖယ်ထုတ်ချင်တယ် ဆိုရင်တွေ၊ `remove()` Method ကို သုံးလို ရပါတယ်။ ကိုယ်ထုတ်ချင်တဲ့ ကောင်ကို ရေးလိုက်ယူပါပဲ။ `Marvel.remove('Captain America')` ဆိုရင် captain america သည် Marvel ဆိုတဲ့ list ထဲမှာ ရှိတော့မှာ မဟုတ်ပါဘူး.. print ထုတ်ပြီး စစ်ကြေည့်ပါ။

```
>>> Marvel.remove('Captain America')
>>> print Marvel
['Iron Man', 'Thor', 'Hulk', 'Spidey', 'Wolverine', 'Black Bolt']
```

အဲလို နာမည်နဲ့ list ထဲကနေ ထုတ်ပစ်လိုက်လို ရောလို index ID နဲ့ ဖယ်ထုတ်လိုက်လို ရပါတယ်။ Index နေရာကို မသိရင် အရင်ဆုံး `index()` ကို သုံးပြီး ကြေည့်လိုက်ပါ။ `Marvel.index('Black Bolt')` လို ရှိ က်လိုက်ရင် Black Bolt ရှိနေတဲ့ `index = 5` ကို ရလာပါလိမ့်မယ်။ အဲနောက် `pop()` ကို သုံးပြီးတွေ့ `index 5` မှာ ရှိနေတဲ့ ကောင်ကို ဖုက်ပစ်လိုက်ပါ။ `Marvel.pop(5)` လို ရှိကြပြီး ရင် black bote ဖုက်လား သိအောင် print ထုတ်ကြေည့်ပါ။



```
>>> Marvel.index('Black Bolt')
5
>>> Marvel.pop(5)
'Black Bolt'
>>>
>>> print Marvel
['Iron Man', 'Thor', 'Hulk', 'Spidey', 'Wolverine']
>>>
```

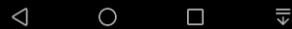
2.8.4 Python Tuples

Tuple သည် list နဲ့ အတူတူပါပဲ။ အမိက ကွဲပြားသွားတာက list သည် ထောင့်ကွင်း [] နဲ့ ဖော်ပြု ။ ရေးသားပြီး tuple ကိုတွော ()လက်သည်းကွင်းနဲ့ ရေးပါတယ်။ ပြီးတွော list ထဲမှာရှိတဲ့ elements/items တွေ ကို ပြုပြင်ပြောင်းလဲလို့ ရပြီးတွော Tuple ကတွော မရပါဘူး.. ဒဲတောငြာနဲ့ သူ့ကို read-only လို့ ပြောစုပါ တယ်။

```
>>> exit()
[root@localhost]~#
[root@localhost]~#
#python
Python 2.7.16rc1 (default, Feb 18 2019, 11:05:09)
[GCC 8.2.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> tuple = ('hello',123,'hi',2.2,'0')
>>> list = ['hello',123,'hi',2.2,'0']
>>>
>>> list[1] = 111
>>> tuple[1] = 111
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
>>> print list
['hello', 111, 'hi', 2.2, '0']
>>>
>>> █
```

ခုအပေါ်ပုံမှာဆို tuple ဆိုပြီး () နဲ့ တစ်ခု create လုပ်လိုက်ပါ။ list အတွက်ကိုတွော [] နဲ့ create လုပ်ပါ။ tuple ရယ် list ရယ် ဆိုတာက ဖွန်တော်တို့တွေပဲ နားလည်တာပါ။ Python ကိုယ်တိုင်နားလည်တာ ကတွော .. [] နဲ့ဆို ပြုပြင်ပြောင်းလဲ လို့ရတဲ့ list ဖြစ်ပြီးတွော () ဆိုရင်တွော ပြင်လို့ ပြုလို့ မရတဲ့ tuple ဆိုပြီး သိတာပါ။

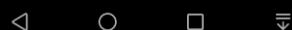
```
>>>
>>> print tuple
('hello', 123, 'hi', 2.2, '0')
>>> print tuple[0]
hello
>>> print tuple[0:2]
('hello', 123)
>>> print tuple * 2
('hello', 123, 'hi', 2.2, '0', 'hello', 123, 'hi', 2.2, '0')
>>> █
```



ရန်တဲ့ print ထုတ်တဲ့ နည်းတွေကတော့ အပေါ်က အတိုင်းပါ အတူတူပါပဲ။

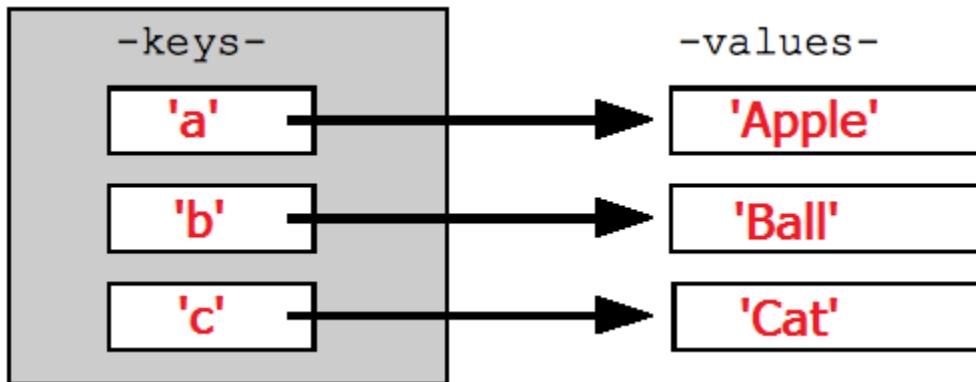
2.8.5 Python Dictionary

```
>>> thedict1 = {"name":"MgMg"
...
KeyboardInterrupt
>>> thedict1 = {
... "Name":"MgMg",
... "Age":26
... , "Position":"Engr"
...
}
>>>
>>> thedict2 = {}
>>> thedict2['a'] = 'Apple'
>>> thedict2['b'] = 'Ball'
>>> thedict2['c'] = 'Cat'
>>> █
```



Dictionary မှာကတော့ အရေးကပြောခဲ့တဲ့ lists တို့ tuples တို့ နဲ့မတူပါဘူး.. သူသည် data တွေ ကို သိမ်းဆည်းတဲ့ နေရာမှာ key နဲ့ value ကို တစ်တွဲ တွဲ ပြီး သိမ်းဆည်းပါတယ်။ key နဲ့ value ကို : နဲ့ အားထားပါတယ်။ key ကို ' or " နဲ့ ထည့်ရေးပါမယ်။ value ကတော့ string/int/float ပေါ်မှုတည်ပြီး ရေးယုံပါပဲ။ ပြီးတော့ {} တွန်းကွင်း ကို အသုံးပြုပါတယ်။

ဒဲ အပေါ်က ဥပမာ လိမ့်း thedict1 = { ဆိုပြီး တစ်ခါတည်း ရေးသွားလို့ ရသလို .. thedict2 = {} ဆိုပြီး အရင် create လုပ် နောက်မှ thedict ['a'] = 'Apple' ဆိုပြီး key = a အဆွဲ့ apple ဆိုတဲ့ value ထည့်ကပ်လို့ ရပါတယ်။



ဟုတ်ပြီ.. တစ်ခြား dictionary ကို create လုပ်တဲ့ နည်းလေး တွေကိုလဲ ဖြာည့် ဖြာည့်ရာရအောင်။ Dictionary ထဲက value သည် list အနေနဲ့လဲ ထည့်လိုက်ပါတယ်။ key 1 ထဲကို [2,4,6] ဆိုတဲ့ value ထည့်ထားပါတယ်။ python2.7 မှာဆိုရင် `print dictionary1[1]` ဆိုပြီး `print` ထုတ်ပါ။

```
>>> dictionary1 = {'Name': 'KP', 1:[2,4,6]}
>>>
>>> print dictionary1[1]
[2, 4, 6]
>>>
```

python 3.x မှာဆို `print(dictionary1[1])` ဆိုပြီးတော့ `print` ထုတ်ပါ။

```
>>> dictionary1 = {'Name': 'KP', 1:[2,4,6]}
>>>
>>> print(dictionary1[1])
[2, 4, 6]
>>>
```

ခုအောက်ကောင်ကတွေ့ python3.x မှာ ပဲ အဆင်ပြေတာပါ။

`dict()` ကို သုံးပြီး `dictionary2` ထဲ ကို key+value တွေကို ထည့်တာပါ။

```
>>> dictionary2 = dict({1: "one", 2: "two"})
```

Dictionary3 မှာရာတွေ့ ('a','apple') ဆိုပြီး key+value ကို တစ်ချမ်း သီးခြားစီ ထည့်ပါတယ်။

```
>>> dictionary3 = dict([('a', 'apple'), ('b', 'ball')])
```

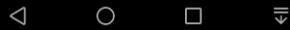
သွင်းထားတဲ့ data တွေကို ပြန်ထုတ်ဖြာည့် ရာရအောင်။ `print thedict1 / print thedict2` ဆိုပြီး ထုတ်ပဲ ကဗျာည့်ပါ။ dictionary ကို unorder, changable ဖြစ်တယ်လို့ ပြောရာတယ်မှာ့.. ဘာကို ပြောချင်တာလဲ ဆိုင်တဲ့ ခု ထည့်ထားတဲ့ data တွေသည် အဆင်လိုက်ပြန်ထွက်မလာတာကို unorder ဖြစ်တယ်လို့ ပြောပါတယ်။ ပြီးတော့ `changable` ဖြစ်တယ်ဆိုတဲ့ အတိုင်း tuple လို့ `read-only` မဟုတ်ပဲ.. ပြုပြင်ပြောင်းလဲ လို့ ရပါတယ်။



```
>>> print thedict1
{'Position': 'Engr', 'Age': 26, 'Name': 'MgMg'}
>>>
>>> print thedict2
{'a': 'Apple', 'c': 'Cat', 'b': 'Ball'}
```

Key ကို သုံးပြီးတော့ သူနဲ့ Match ဖြစ်နေတဲ့ data value ကို ထုတ်ပြန်ရာရအောင်.. အရင်ကောင်တွေလို့.. `print pyDict['apple']` ဆိုပြီး ထုတ်လိုက်ပါ။ ဒီမှာ နဲ့ ကွာခြားသွားတာက index နဲ့ မဟုတ်ပဲ ကို ထုတ်ချင်တာကို key နဲ့ ညွှန်းပေးရတာပါပဲ။ data value ကို ပြင်မယ် modify လုပ်မယ်ဆိုရင် ကိုယ် ပြင်ချင်တဲ့ ကောင်ရဲ့ key ကို ညွှန်းပေးရမှာပါ။ `pyDict['car'] = 'ကား'` ဆိုရင် car ဆိုတဲ့ Key အသွေး value သည် မော်တာကား မဟုတ်တော့ ပဲ ကားဖြစ်သွားပါတယ်။ ပြောင်းသွားတာ သေချာအောင် `print` ထုတ် ပြန်ပါ။

```
>>> pyDict = {
... 'apple':'ပန္နာသီး',
... 'car':'မော်တော်ကား',
... 'cloud':'ထိမ်'
... }
>>>
>>> print pyDict['apple']
ပန္နာသီး
>>> pyDict['car'] = 'ကား'
>>> print pyDict['car']
ကား
>>>
```



`print pyDict['car']` ဆိုပြီး ထုတ်လိုက်တဲ့ result နဲ့ `print pyDict.get('car')` ဆိုပြီး `get()` ကိုသုံးလိုက်တဲ့ result သည် pyDict မှာ တည်ရှိနေတဲ့ key အတွက်တော့ အတူတူပါပဲ.. အကယ်လို့ မရှိတဲ့ key ရဲ့ value ကိုထုတ်မယ် ဆိုရင်တွေ့ ရလာတဲ့ result သည် မတူညီတွောပါဘူး။ `get()` နဲ့ ဆိုရင် key မရှိတာ အတွက် None ဆိုပြီး result return ပြန်လာပြီးတော့ pyDict['Hello'] မှာရှာတော့ key error တက်ပါတယ်။

```
>>> print pyDict['car']
None
>>> print pyDict.get('car')
None
>>>
>>> print pyDict.get('Hello')
None
>>>
>>> print pyDict['Hello']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'Hello'
>>>
>>> █
```

ခုစာကြိမ်း ထပ်မံ့ဖြတ်ရှာမှာက ဖုန်းပစ်တာ.. ထပ်ထည့်တာ တွေကို **pop()** **update()** **clear()** **del** တို့ကို သုံးပြီး စမ်းခြားပြုရှာမှာပါ။

```
>>> dict123 = {
...: 'hi':'Hello',
...: 'ok':'Good',
...: 'run':'Play',
...}
>>> print dict123
{'run': 'Play', 'hi': 'Hello', 'ok': 'Good'}
>>> dict123.pop('hi')
'Hello'
>>>
>>> del dict123['ok']
>>> print dict123
{'run': 'Play'}
```

အပေါ်ကပုံမှာဆို **dict123** ဆိုပြီး dictionary တစ်ခု တည်ဆောက်ထားပါတယ်။ **print dict123** ဆိုပြီး ထုတ်မြှောင်းပါ။ ခုက **python 2.7** နဲ့ မြို့နော.. **Python3.x** ဆိုရင် **print(dict123)** လို့ ရေးရမှာ မမူနေနောက်။ **pop()** ကိုသုံးပြီးတော့ **dict123.pop('hi')** ဆိုပြီး 'hi' ဆိုတဲ့ key နဲ့ သူ့ရဲ့ value Hello ကို ဖယ်ထုတ် လိုက်ပါတယ်။ **del** ကလဲ အလုပ် လုပ်ပုံဆင်ပါတယ်။ **del dict123['ok']** ဆိုပြီး 'ok' နဲ့ 'Good' ကို ဖုန်းပစ်လိုက်ပါတယ်။ ကျားသွားတာက **pop()** သည် ဘယ် value ဖယ်လိုက်တယ် ဆိုတာကို ပြန် ပြုပြီး **del** ကတော့ ဘာမှ ပြန်မပြပါဘူး။ **print dict123** ဆိုပြီးထုတ်မြှောင်းပါတယ်။

```
>>> dict123.update({'black':'white'})
>>> print dict123
{'run': 'Play', 'black': 'white'}
>>>
>>> dict123.update( [ ('Name','AungAung') , ('who','why') ] )
>>> print dict123
{'run': 'Play', 'who': 'why', 'black': 'white', 'Name': 'AungAung'}
```

`update()` ကို သုံးပြီးတော့ `key+value` pair ကို ထည့်ကြာည့်ရနာရအောင်။ အပေါ်က ပုံမှာဆို `dict123.update({'black':'white'})` ဆိုပြီးတော့ **black (key) + white (value)** အတွဲ ကို `dict123` ထဲ ကို ထည့်လိုက်ပါတယ်။ `print dict123` ဆိုပြီး ပုံနှိပ် ထုတ်ကြေည့်ပါ။ pair တွေကို တစ်ခုထက် မက ထည့်လို့ ရပါတယ်။ `dict123.update([('Name','AungAung') , ('who','why')])` မှာဆိုရင် လက်သည်းကွင်းထဲ မှာ (`key , value`) တွေ pair အလိုက် များပြီး ထည့်လို့ ရပါတယ်။ ငင်သွားတာ သိချင်ရင် `print dict123` နဲ့ ထုတ်ကြေည့်ပါ။

```
>>> dict123.clear()
>>> print dict123
{}
>>> del dict123
>>> print dict123
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'dict123' is not defined
>>
```

`clear()` ကို dictionary ထဲက `key+value` pair တွေ အကုန်လုံးကို ဖုက်ပစ်လိုက်တာပဲ ဖြစ်ပါတယ်။ `print` ထုတ်ကြေည့်ရင် data မရှိတဲ့ { } ကိုပဲ တွေ့ရမှာ ဖြစ်ပါတယ်။ `del dict123` ဆိုတာကရှာတော့ `dict123` ဆိုတဲ့ ကောင်ကြီးကိုပါ ဖုက်ပစ်လိုက်တာပဲ ဖြစ်ပါတယ်။ ;-)

မှတ်ထားရမှာကာ။ **Tuple သည် ()** **List သည် []** ခုလဲလာနေတဲ့ **Dictionary သည် {}**

3. Python Operator

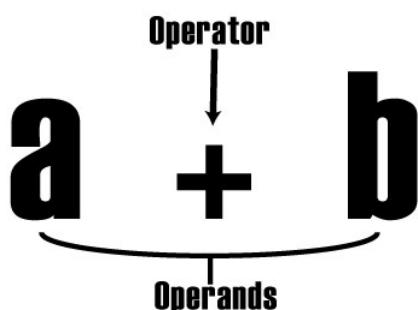
Python တွေ နိုင်မယ့် operator တွေကတော့ အောက်ပါ အတိုင်းပဲ ဖြစ်ပါတယ်။

- Arithmetic Operators
- Comparison (Relational) Operators
- Assignment Operators
- Logical Operators
- Bitwise Operators
- Membership Operators
- Identity Operator

3.1 Arithmetic Operators

Arithmetic ကတွေ့ သူ့နာမည်အတိုင်းပဲ ပေါင်း နှစ် ပြောက် စား စတဲ့ .. သရုပ်တွက်ရှုက် မှုတွေကို
ပြုလုပ်ပေးတာပဲ ဖြစ်ပါတယ်။

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y



ပေါင်းနှစ်ပြောက်စား (+-*%)စတဲ့ သက်တတွေကို operator လို ခေါ်ပြီးတော့ x, y စတဲ့ ကောင်တွေကိုတော့ operands တွေလို ခေါ်ပါတယ်။

အပြင်တော့ advance level လိုက်ချင်လို.. စာတွေဖတ်ရင် နားလ ည်အောင်လို.. သုံးတ်တဲ့ အသုံးလေးတွေ ခေါ်လေးတွေ ကို မြင်က အာင် ရေးပေးထား တာပါ။ operator ဆိုတာဘာလဲ operand ဆို တာဘာလဲ ဘယ်လို ခေါ်လေးမောင်လိုပါ။

ဟုတ် code လေးတွေ ရေးပြီး လက်တွေ့စမ်း သပ်ရှာည့်ရာရအောင်။ ;-)

3.1.1 Addition (+)

```
>>> x = 10
>>> y = 20
>>>
>>> print x + y
30
>>>
```

operator	= +
operand	= x,y
result	= 30



X = 10 ဆိုပြီးတော့ variable x ထဲကို 10 ထည့်လိုက်ပါတယ်။ y ထဲ ကိုတော့ 20 ထည့်ထားပါတယ်။ အဲနောက် **print x + y** နဲ့ xနဲ့y ပေါင်းပြီး ရလာတဲ့ resulte ကို ထုတ်ပြထားပါတယ်။

Addition operator ဖြစ်တဲ့ အပေါင်း (+) သည် Integer တွေပေါင်းရင်တော့ calculation လုပ်ခဲ့ပြီး str တွေကို ပေါင်းရင်တော့ ဒီတိုင်းပဲ နှုတ်ဆက်ပြီး ပေါင်းရေးလိုက်တာပါပဲ။ အောက်က ကုတ်ကို စမ်းကြားထဲပါ။

```
a="5"
b="6"

print ("the result of str a(5) + b(6) is " + a+b)

print ("the result of int a(5) + b(6) is " + str(int(a)+int(b)))
```

RUN

```
the result of str a(5) + b(6) is 56
the result of int a(5) + b(6) is 11
>>>
>>> |
```

3.1.2 Subtraction (-)

Python ကို ဒီတစ်ခါဂျာ script အနေနဲ့ ရေးကြည့်ရှုရအောင်.. editor တစ်ခုနဲ့ python code တွေ ကို ရေးလိုက်ပါ။ ကိုယ်ကြိုက်တဲ့ နာမည်နဲ့ save ပါ။ ရုဒ် ဥပမာဏာတော့ sub.py ဆိုပြီး save ထားပါတယ်။

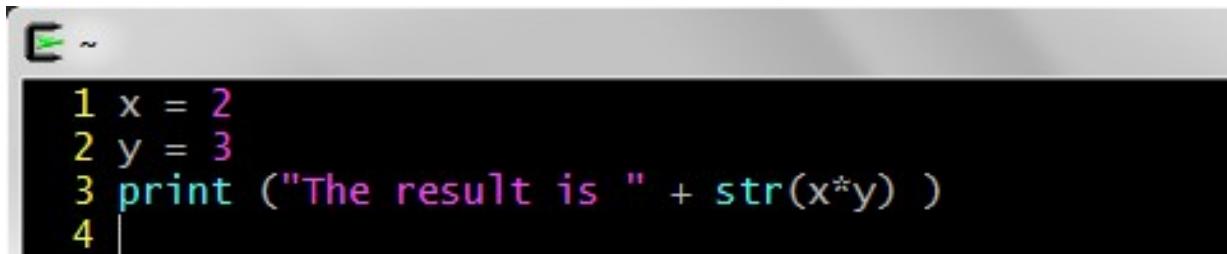
```
a = 20
b = 5
z = a - b
print z
```

ပြီးတော့ python sub.py ပြီး run လိုက်ရင် အခြေ 15 ဆိုပြီး ရပါတယ်။

```
[alice@host-1-185 ~]$ gedit sub.py
[alice@host-1-185 ~]$ python sub.py
15
[alice@host-1-185 ~]$
```

3.1.3 Multiplication (*)

Multication ဆိတဲ့ အတိုင်း ပြောင်မှာပေါ့။ python မှာ အကြောက်အတွက် *(start) သက်တ ကို သုံးပါတယ်။ အောက်က code လေးကို ရေးပြီး Multi.py ဆိတဲ့ နာမည် နဲ့ သိမ်းလိုက်ပါတယ်။ ကိုယ်စိတ် ဖြို့က်နာမည်ပေးလို့ ရတယ်နော်။

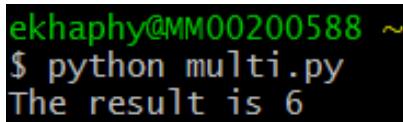


```

1 x = 2
2 y = 3
3 print ("The result is " + str(x*y) )
4

```

ဒုက္ခာင်းပြောက် code ဖြစ်တဲ့ `print ("The result is " + str(x+y))` အကြေားကို နဲ့လေး စာပြန်နေး ပါမယ်။ `x` နဲ့ `y` သည် int ဖြစ်တာမို့ သူတို့ကို ပေါင်းလို့ ရလာတဲ့ `result` သည်လဲ int ပဲ ဖြစ်ပါတယ်။ အဲတာပြောင့် `print` ကို သုံးပြီး ထုတ်တဲ့ အချိန်မှာ အမျိုးအစား တူမှ အလုပ် လုပ်တာမို့ `str()` ကိုသုံးပြီး string အနေနဲ့ ပြောင်းလိုက်ပါတယ်။ အလယ်က အပေါင်းလေးကတွော "The result is " ဆိတဲ့ `string` နဲ့ နောက် integer ကနေ ဖြစ်လာတဲ့ `string` နှစ်ခုကို ပေါင်းပြီး `print` ထုတ် ပေးတာပါ။

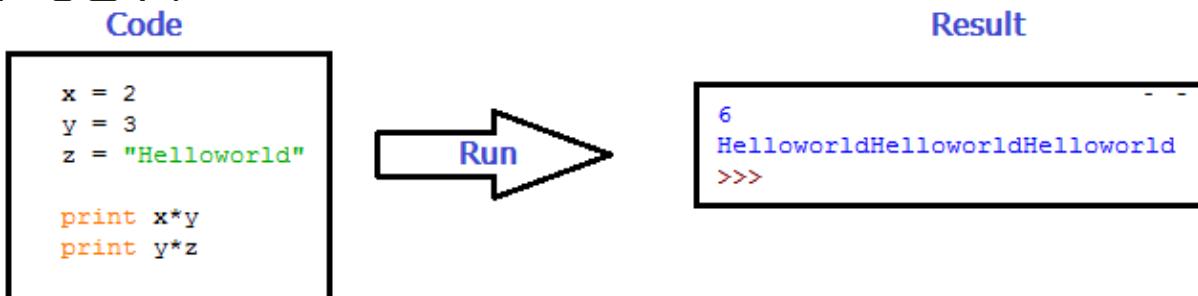


```

ekhaphy@MM00200588 ~
$ python multi.py
The result is 6

```

ပြောက်တဲ့ အခါမှာ int * int တင်မဟုတ်ပဲ int * str လဲ ပြောက်လို့ရပါတယ်။ အောက်က ကုတ်ငါးကိုစမ်းကြေည့်လို့ ရပါတယ်။



နောက်ထပ် အဆင့်လေးနည်းနည်း လောက် ထပ်တိုးကြည့်ရာရအောင်။ ဒုက္ခာင်းကြိုတင် သက်မှတ်ထားတဲ့ တန်ဖိုးတွေကနေ တွက်ထုတ်နေတာ။ ဆိုတော့ အဲလို့ မဟုတ်ပဲ User ဆိုကောင်းမှု input (operand) ယူပြီး တွက် ထုတ်ကြည့်ရာရအောင်။

Python User Input

Python 2.7 မှာဆိုရင် user ဆိုကနေ input ယူဖိအတွက် သုံးနိုင်တဲ့ function က နှစ်ခု ရှိတယ်။
input() နဲ့ .raw_input() တို့ ပါဖော်ပါတယ်။ အဲနှစ်ကောင်ဘယ်လို ကွာခြားလဲဆိုတာကို တစ်ချက်ကြေည့် ဖြော်ရှာရအောင် များ။

```
>>> abc = input("Please enter your input : ")
Please enter your input : 123
>>> type(abc)
<type 'int'>
>>> abc
123
>>>
>>> abc = input("Please enter your input : ")
Please enter your input : '123'
>>> type(abc)
<type 'str'>
>>> abc
'123'
>>>
```

`abc = input("Please enter your intput : ")` ලදී. රෙපු: enter ගෙන්ලදිග්නතාක් `Please enter your input :` සිංහලා පෝළාවිලිමුවයි॥

ဘာလို့လဲဆိုတော့ python ရဲ့ interpreter သည် line by line / statement by statement အလုပ်လုပ်ပါတယ်။ ခုက interactive mode ဆိုတော့ တစ်စင်ဒြောင်းရေးရင် တစ်စင်ဒြောင်း လုပ်သွားတာကို ပိုမြီးမြင်သာပါတယ်။ script mode မှာရာတော့ python အနေနဲက code တွေကို တစ်ခုချင်း အလုပ်လုပ် (execute) လုပ်သွားသော်လဲ.. output ကိုပဲ မြင်ရတာပေါ့။

အိုကေ ဆက်မယ်.. အဲလို့ Please enter your input : ခါးပြီး တောင်းတွေ 123 လို့ enter ခေါက်ပြီး input အနေနဲ့ ထည့်လိုက်ပါ။ အဲတာဆိုရင် abc ဆိုတဲ့ variable ထဲကို user ကနေထည့်လိုက်တဲ့ input တစ်ခု ပေါ်သွားပါပြီ။ အိုကေဘင်သည့် int (integer) အနေနဲ့ ပင်တာလား str (string) အနေနဲ့ ပင်တာလား သိချင်ရင် type() ကိုသုံးပြီး abc ကို စစ်ရှေ့ညွှပ်ပါ။ int type ဖြစ်နေတာကို တွေ့ရပါမယ်။ အဂါရောင်နဲ့ hight light လုပ်ပေးထားပါတယ်။ abc ထဲက data ကို ထုတ်ချေရင် >>> (chevron) မှာ abc လို့ရေးပြီး enter ခေါက်လိုက် ယုံပါပဲ။

ဒါ integer ထည့်လိုက်တွော ပြန်ထွက်လာတာသည်လဲ intergerပါ။ string ထည့်ရင်ကော ဘာပြန်ထွက်လာမလဲ စမ်းပြောလိုပါ။ **abc = input("Please enter your intput : ")** လို့ရေးပြီး enter ခေါက်လိုက်တာနဲ့ input တောင်းရင် '123' ဆိုပြီး single quote နဲ့ ရေးလိုက်ပါက python သည် အဲသည့် 123 ကို string အနေ နဲ့ ထည့်တယ်လို့ ယူဆပါတယ်။ input() ကိုသုံးပြီး abc ထဲ ကို ထည့်လိုက်တဲ့ အချိန် မှာ ဘာအမျိုး အစားလဲ သိ ခဲင်ရင် type() ကိုသုံးပြီး စစ်ပြောလိုပါက str (string) ဖြစ်နေတာကို တွေ့ရမှာပါ။ abc ထဲ က data ကို ပြန်ထွက်ကြောလိုပ်ရင်လဲ '123' ဆိုပြီး ထွက်လာတာကို တွေ့ရမှာပဲ ဖြစ်ပါတယ်။

Input() ကို စမ်းပြီးပြီ ဆိုတော့ raw_input() ကို ဆက်ပြီး စမ်းကြေည့်ရှုပါ။ xyz = input("Please enter your input : ") လို့ ရှိနိုင်ပါ။ Please enter your input : ဆိုပြီး user ဆီကနေ input တောင်းပါလိမ့်မယ်။ အဲဒေသချက် 123 လို့ ရှိတဲ့ပြီး enter ခေါက်လိုက်ရင် variable xyz ထဲကို အဲdata ပင်သွားပါလိမ့်မယ်။ str လား int လား သိချင်ရင် type() ဖြောင့်လိုက်တဲ့ အခါမှာတော့ str အနေနဲ့ သိမ်းဆည့်းတာကို တွေ့ရပါတယ်။ xyz ထဲမှာ ရှိတဲ့ data ကို ပြန်ထုတ် ဖြောင့်ရင်လဲ '123' ဆိုပြီး string အနေနဲ့ ပဲ တွေ့ရပါတယ်။

```
>>> xyz = raw_input("Please enter your input : ")
Please enter your input : 123
>>> type(xyz)
<type 'str'>
>>> xyz
'123'
>>>
```

နောက်တစ်ခါ '123' ဆိုပြီး string အနေနဲ့ input ထည့်လိုက်ပါတယ်။ type() နဲ့ ဖြောင့်ရင် string ဖြစ်တယ် ဆိုတာကို တွေ့ရမှာပါ။ အဲတာအပြင် သိမ်းတဲ့ အခိုင်မှာ သူသည် '(single quotes)' တွေကိုပါ ထည့်သိမ်းပါတယ်။ အဲဒေသချက် xyz ထဲ က data ကို ပြန်ထုတ်တဲ့ အခိုင်မှာ "' 123 ' " ဆိုပြီး မြင်ရတာပါ။

```
>>> xyz = raw_input("Please enter your input : ")
Please enter your input : '123'
>>> type(xyz)
<type 'str'>
>>> xyz
"'123'"
>>>
```

Python2.7 အောင် user input Function တွေ ပြီးသွားပြီ ဆိုတော့ python3.x အတွက် ဆက်ပြီး လေ့လာပုက္ၣ် ရာရအောင်။ ဒါ ရွှေနံတိုးဝင်တဲ့မှာ ရှိတဲ့ Python3.7 နဲ့ စမ်းပြထားပါတယ်။

```
>>> a = input("Please enter : ")
Please enter : 123
>>> type(a)
<class 'str'>
>>> a
'123'
>>>
```

input() function ကို သုံးပြီး input တောင်းတဲ့ အခိုင်မှာ integer အနေနဲ့ 123 ဆိုပြီး ထည့်လိုက်ပါတယ်။ type() function နဲ့ ပြန်ပြီး စစ်ကြည့်တဲ့ အခိုင်မှာ str အနေနဲ့ ပဲ တွေ့ရပါတယ်။ a ထဲ က data ကို ပြန်ထုတ်ရင် လဲ '123' ဆိုပြီး ထွက်လာပါတယ်။

```
>>> a = input("Please enter :")
Please enter : '123'
>>> type(a)
<class 'str'>
>>> a
"'123'"
>>>
```

ဒီတစ်ခါမှာတော့ '123' ဆိုပြီး string အနေနဲ့ variable a ထဲကို user ကနေ ရှိက်တဲ့ input ကို ထည့်လိုက်ပါတယ်။ type() နဲ့ ပြောလျှင် str ဆိုတာကိုပဲ တွေ့ရပါတယ်။ a ထဲက data ကို ပြန်ထုတ်ပြေလျှင်တဲ့ အခါမှာတော့ " '123 ' " ဆိုပြီးတော့ '(single quotes) တွေ့ကိုပါ ထည့်သိမ်းထားတာကို တွေ့ရပါမယ်။

Input ယူတဲ့ နည်းတွေကို သိသွားပြီ ဆိုတော့ Multiplication ကို ပြန် စမ်းပြောလျှင်ရာရ အောင်။

```
ekhaphy@MM00200588 ~
1 a = input("Enter first input : ")
2 b = input("Enter second input : ")
3
4 result = a * b
5
6 print ("The result of first * second is => " + str(result))
7
```

အပေါ်က ကုတ်တွေကို multi_input.py ဆိုတဲ့ နာမည်တဲ့ save လိုက်ပါ။ အဲ script ဖိုင်ကို python multi_input.py ဆိုပြီး execute လုပ်လိုက်ပါ။ အောက်ပါ result အတိုင်း ရရှိလာပါလိမ့်မယ်။

```
ekhaphy@MM00200588 ~
$ vim multi_input.py

ekhaphy@MM00200588 ~
$ python multi_input.py
Enter first input : 10
Enter second input : 3
The result of first * second is => 30

ekhaphy@MM00200588 ~
$
```

အဲဆို ငြောက်တာလဲ ပြီးပြီ ဆိုတော့ နောက်ထပ် ရှန်နေတဲ့ Division(/), Modulus(%), Exponentiation(**), Floor division or quotient(//) တွေကို script နဲ့ တစ်ခါတည်း ရေးပြောလျှင်ရာ ရအောင် ;)

***** Python 2.7(dvision, Modulus, Exponentiation) *****



အောက် ကုတ်ကို ထုံးစံ အတိုင်း ရေးပြီး save ပါ။ ရွှေ့နှံလောကတွေ့ math.py ဆိုပြီး သိမ်းချွဲပါတယ်။

```

1 x = input("Enter Num1 : ")
2 z = input("Enter Num2 : ")
3
4 print ("The result of Num1/Num2 is " + str(x/z))
5 print ("The result of Num1%Num2 is " + str(x%z))
6 print ("The result of Num1**Num2 is " + str(x**z))
7 print ("The result of Num1//Num2 is " + str(x//z))
8

```

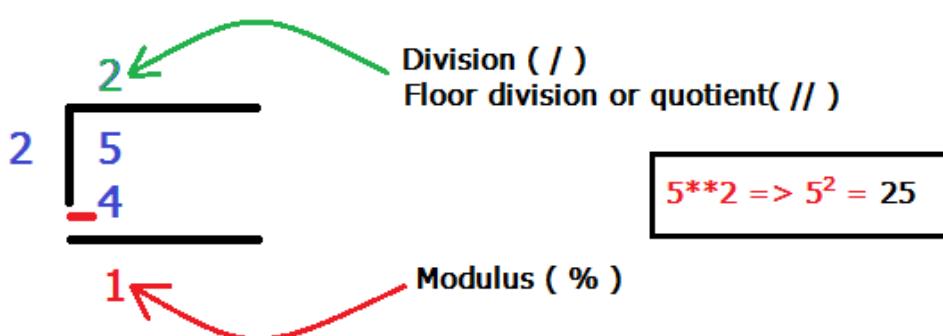
အဲနောက် script ကို execute လုပ်လိုက်တွေ့ Num1 အတွက် ၅ ပေးပြီ Num2 အတွက် ၂ ထည့်ကောင်းလိုက်ပါတယ်။

```
ekhaphy@MM00200588 ~
$ vim math.py
```

```
ekhaphy@MM00200588 ~
$ python math.py
Enter Num1 : 5
Enter Num2 : 2
The result of Num1/Num2 is 2.5
The result of Num1%Num2 is 1
The result of Num1**Num2 is 25
The result of Num1//Num2 is 2
```

```
ekhaphy@MM00200588 ~
$
```

ရလာတဲ့ result သည် Num1 (5) ကို Num2 (2) နဲ့ စားလိုက်ရင် division or quotient သည် 2 ဖြစ်ပြီး modulus သည် 1 ဖြစ်ပါတယ်။ ပြီးတွေ့ Num1 to the power Num 2 => 5^2 ဖြစ်တာမူး 25 ဖြစ်ပါတယ်။ ဒါက Python 2.7 မှာ မူး division နဲ့ Floor division or quotient နဲ့ ကဲ တူတာပါ။



***** Python 3.7(dvision, Modulus, Exponentiation) *****

Python 3.7 မှာဆိုရင် quotient ကို . အသမကိန်း (float) နဲ့ ပြတာမို့ division နဲ့ မတူတွော ပါဘူး။ ပြီးတော့ ရေးမထုတ်မှာကလဲ နဲ့နဲ့လေး ကျွိုးသွားပါတယ်။ input() သည် Python 2.7 မှာ တုန်းက int သွင်းရင် int အနေနဲ့ သိမ်းပြီး str သွင်း str ဝင်ပါတယ်။ ဒါ Python 3.7 မှာရာ ဘယ် input နေဖော် str အနေနဲ့ ပဲ သိမ်းပါတယ်။ အဲတာ ငြောင့် Math calculation တွေလုပ်မယ်ဆို string ဖြစ်လို့ ကတော့ ဘယ်လိုမှ အဆင်မပြေပါဘူး.. int() ကို သုံးပြီး ရလာတဲ့ input ကို str to int ပြောင်းပေးရပါမယ်။ အောက်က ကုတ်ကို ဤ ပြုပါ။

```

x = int(input("Enter Num1 : "))
z = int(input("Enter Num2 : "))

print ("The result of Num1/Num2 is " + str(x/z))
print ("The result of Num1%Num2 is " + str(x%z))
print ("The result of Num1**Num2 is " + str(x**z))
print ("The result of Num1//Num2 is " + str(x//z))

```

Math.py ဆိုတဲ့ နာမည်နဲ့ပဲ သိမ်းလိုက်တာမှို့.. Python3 math.py ဆိုပြီး run လိုက်ပါ။

```

ekhaphy@MM00200588 ~
$ vim math.py

ekhaphy@MM00200588 ~
$ python3 math.py
Enter Num1 : 5
Enter Num2 : 2
The result of Num1/Num2 is 2.5
The result of Num1%Num2 is 1
The result of Num1**Num2 is 25
The result of Num1//Num2 is 2

ekhaphy@MM00200588 ~
$ |

```

ဟုတ်ပြီး ခုဆိုရင် division သည် float တန်ဖိုး ဖြစ်တဲ့ 2.5 ကို ပေးပါတယ်။ quotient ကတော့ 2 ပါ။ အဲလောက်ဆို ရမယ် လို့ မျှော်လင့်ပါတယ်။ တစ်ခြား ဂဏန်းတွေနဲ့လည်း စမ်းပြောည့်စေရင်ပါတယ်။ happy coding ;-)

3.2 python Comparison operator

Python မှာရှိတဲ့ variable/operand တွေကို နိုင်းယျဉ်တဲ့ ပေးတဲ့ operator တွေက အောက်ပါ။ အတိုင်း ဖြစ်ပါတယ်။ သူတို့၏ symbol သက်တော့ အတိုင်း အလုပ် လုပ်တာပါ။ ကုတ်လေးတွေ ရေးပြီး စမ်းကြားရှုရအောင် ဖြာ။

<i>op</i>	<i>meaning</i>	<i>True</i>	<i>False</i>
<code>==</code>	<i>equal</i>	<code>2 == 2</code>	<code>2 == 3</code>
<code>!=</code>	<i>not equal</i>	<code>3 != 2</code>	<code>2 != 2</code>
<code><</code>	<i>less than</i>	<code>2 < 13</code>	<code>2 < 2</code>
<code><=</code>	<i>less than or equal</i>	<code>2 <= 2</code>	<code>3 <= 2</code>
<code>></code>	<i>greater than</i>	<code>13 > 2</code>	<code>2 > 13</code>
<code>>=</code>	<i>greater than or equal</i>	<code>3 >= 2</code>	<code>2 >= 3</code>

Comparisons with int operands and a bool result

ဒု basic အနေနဲ့ OP 6 ခုကို စမ်းပို့အတွက် interactive mode မှာ စမ်းကြားရှုပါမယ်။ ထို့လိုက်တဲ့ ရလာတဲ့ input တွေပေါ်မှတ်ညွှန်ပြီး condition ကို စစ်ပိုမယ်။ condition မှန်တယ် ဆိုရင် True (boolean) ကို ရမှာ ဖြစ်ပြီး.. Condition မကိုက်ရင် ထော့ false ပါ။.. **Comparison Operator** ဆိုတဲ့ အတိုင်း **operands** တွေကို compair(နိုင်းယျဉ်) လုပ်တာပါ။ လက်တွေ ကုတ်တွေ ရေးကြားရင် မှာဟိုထဲ ပိုမြင်ပါလိမ့်မယ်။ စမ်းကြားရှုရမှာ ဖြစ်ပါ။

```
>>> x = 5
>>> y = 5
>>>
>>> x == y
True
>>>
>>> x != y
False
>>>
```

X = 5, y = 5 ဆိုပြီး x နဲ့ y ထဲကို 5 တွေထဲည့်လိုက်ပါတယ်။ အဲနောက် x == y ဆိုပြီး x ထဲမှာရှိတဲ့ value နဲ့ y ထဲမှာရှိတဲ့ == ဆိုတဲ့ operator ကိုသုံးပြီးတော့ နိုင်းယျဉ်ကြားရှုပါမယ်။ == ဆိုတာသည် ညီလား ဆိုတာကို စစ်တာပါ။ x == y သည် 5 == 5 ဖြစ်တာမို့ condition ကိုက်ညီတဲ့အတွက် True ဆိုပြီး ရပါတယ်။



ဒီနေရာမှာ တစ်ခုမှတ်ရမှာသည့် **`==`** သည် condition ကိုစစ်တဲ့ operator ဖြစ်ပြီးတော့ **`=`** သည် assignလုပ်ပေးတဲ့ ကောင်ပါ။ အဲနောက် **`x != y`** ဆိုပြီး condition ကိုစစ်ပြုလိုက်ပါတယ်။ ဒီနေရာမှာ ! သည် not ဆိုတဲ့ အဓိပ္ပာယ်ဖြစ်တာမို့ **`x is not equal y ?`** ဆိုပြီးတော့ စစ်ပြုလိုတဲ့ အဆိုနှင့် 5 != 5 သည် ညီနေတာမို့ condition မကိုက်တွေ့ယဲပဲ **`false`** ရပါတယ်။ တစ်ကယ်လို `!=` နဲ့ စစ်ပြီး True ရချင်တယ် ဆိုရင်ငဲ့ `x, y` ထဲကို မတူညီတဲ့ ကေန်းတန်ဖိုးတွေ ထည့်လိုက်ပါ။

```
>>> x = 2
>>> y = 3
>>>
>>> x != y
True
>>>
```

equal နဲ့ not equal ကိုတွေ့ စမ်းပြီးသွားပြီးဆိုတွေ့ Greater than / Less Than ကို ဆက်စမ်းပြုလိုရနှို

• `x = 10, y = 20` ထည့်ထားပြီး.. `x > y` လို့ စမ်းလိုက်ရင် ဘာအဖြစ်ထွက်မလဲ ? `x < y` ဆိုရင်ကော့? အဖြေကို စိတ်ထဲကနေ မှန်းပြုလိုထားနော့.. အောက်မှာ ကုတ်လေးရေးပြီး စမ်းပြုလိုမယ် ကိုယ်ထင်တာမှန် မမှန်ပေါ့။

```
>>> x = 10
>>> y = 20
>>>
>>> x > y
False
>>>
>>> x < y
True
>>>
```

`x` ထဲမှာက ၁၀ ရှိတယ် `y` ထဲမှာက ၂၀ ရှိတယ်။ `x > y` (`x` ၏ `y` ထက်ပြီးလား) ဆိုတွေ့ condition မကိုက်တဲ့ အတွက် `false` ပါ။ `x < y` ဆိုရင်တွေ့ `y` ထဲက ၂၀ သည် `x` ထဲမှာရှိတဲ့ ၁၀ ထက်ပြီးတာမို့ condition မှန်တဲ့ အတွက် `True` ဖြစ်ပါတယ်။

နောက်တစ်ခါ `>=` (Greater than or equal) နဲ့ `<=` (less than or equal) ကို စမ်းပြုလိုရအောင်။ `x >= y` မှာ condition စစ်တဲ့ အဆိုနှင့် `x <= y` ထက်ပြီးခဲ့ရင်ပဲ ဖြစ်ပါမဲ့ ညီခဲ့ရင်ပဲ ဖြစ်ပြီ။ ဘာလိုလဲဆိုတွေ့ `>=` ပဲ အဓိပ္ပာယ် Greater than or equal ဆိုတဲ့ အတိုင်းပြီးတာ နဲ့ ညီတာ တို့ထဲက condition တစ်ခုရ နဲ့ ကိုက်ခဲ့ရင် `True` ပါ။ ဆိုတွေ့ `x` ထဲကို ၂၀ ထည့်ပြုလိုရအောင်.. `y` ကို တော့ အရင်အတိုင်း ၂၀ အနေနဲ့ပဲ ထားခဲ့လိုက်ပေါ့။ အဲဆို condition စစ်ရင် `True` ရရမယ်နော့။



```
>>> x = 20
>>> x >= y
True
>>>
```

x ကို ၃၀ ပေးကြည့်မယ် ဆိုရင် True ဦးမှာလား.. ?

```
>>> x = 30
>>>
>>> x >= y
True
>>>
```

$x = 30, x \geq y$ ဆိုရင်လဲ True ပဲ ရပါတယ်။ x ထဲက ၃၀ သည် y ထဲက ၂၀ ထက် ကြီးတာမို့.. True ပါ။

```
>>> x <= y
False
>>> y = 40
>>> x <= y
True
```

$X \leq y$ (less than or equal) နဲ့ စစ်တဲ့ အဆိုန်ဂျာ x ထဲက dataဖြစ်တဲ့ ၃၀ သည် y ခဲ့ ၂၀ နဲ့ နိုင်းယူဉ်တဲ့ အခါ မှာတော့ less than ဆိုတဲ့ condition နဲ့လဲ မညီ။ Equal ညီလား ဆိုတော့လဲ တန်ဖိုးတွေက မတူတာမို့။ False ပဲ ရပါတယ်။ true ရစေချင်ရင် တန်ဖိုး တူအောင်ဖြစ်ဖြစ် y တန်ဖိုးကို x ထက်ကြီးအောင် ဖြစ်ဖြစ် လုပ်လို က်ရင် condition မှန်သွားမှ True ထွက်ပါတယ်။

3.3 Python assignment Operator

Operator	Name	Example	Equivalent
<code>+=</code>	Addition assignment	<code>i += 8</code>	<code>i = i + 8</code>
<code>-=</code>	Subtraction assignment	<code>i -= 8</code>	<code>i = i - 8</code>
<code>*=</code>	Multiplication assignment	<code>i *= 8</code>	<code>i = i * 8</code>
<code>/=</code>	Float division assignment	<code>i /= 8</code>	<code>i = i / 8</code>
<code>//=</code>	Integer division assignment	<code>i //= 8</code>	<code>i = i // 8</code>
<code>%=</code>	Remainder assignment	<code>i %= 8</code>	<code>i = i % 8</code>
<code>**=</code>	Exponent assignment	<code>i **= 8</code>	<code>i = i ** 8</code>

ဟိုအရေးမှာတုန်းက simple assignment operator လေးတစ်ခုဖြစ်တဲ့ = (equal) အလုပ် လုပ်ပုဂ္ဂိုလ်များမှာတိတယ် ဟုတ်? သူက `i = 5` မှာဆိုရင် ညာဘက်က 5 သည် ဘယ်ဘက်က variable i ထဲကို assign လုပ်ပေးရမယ် value တစ်ခုပါတယ်။ ပြောချင်တာက ညာဘက်မှာရှိတာသည် assign လုပ်မယ့်ကောင် ဘယ်ဘက်ကကောင်သည် assign လုပ်ခံရမယ့်ကောင်.. အလယ်က နေရာကတွော assignment operator အတွက်ပဲ ဖြစ်ပါတယ်။

ခုဆက်လေးလာမယ့် assignment operator တွေကို compound assign operator တွေလို့ ပြောနိုင်ပါတယ်။ ဘာလို့လဲဆိုတော့ သူတို့ပဲ ပုံမှန်အတိုင်း data value ကို assign ပေးတာမျိုးပဲ မဟုတ်ဘူး.. တစ်ခြား ကောင်တွေရဲ့ ဂုဏ်သွေးဌာန (ပေါင်းနှုတ်မြောက်စား စတာတွေ) ပါဝင်ပေါင်းစပ်နေလိုပါ။ `i += 8` ဆိုရင် သူအလုပ်လုပ်သွားတဲ့ ပုံစံသည် `i = i + 8` ပါ။ python interpreter သည် ညာကနေ ဘယ်ကို သွားတာမို့ 8 နဲ့ i ထဲမှာရှိတဲ့ တန်ဖိုးကို ပေါင်းပြီး i ထဲ ပြန်ထည့်လိုက်တာပါ။ မြင်သာအောင်ပြောင်ရင် program တစ်ခါ run ပြီးတိုင်း တစ်တိုးချင်တာမျိုးမှာဆို `i = 0, i += 1;` မှာဆိုရင် i ထဲမှာက initial တန်ဖိုးသည် 0 ပါ အဲတောက်မှ တစ်ခါ run တိုင်း တစ်ပေါင်းချင်တာမို့ `i += 1 (i = i + 1)` လို့ ရေးလိုက်တာပါ။ အကယ်လို့ တစ်ခါ run ပြီးတိုင်း ၂ တိုးချင်တယ်ဆိုရင်တွော `i += 2` ပေါ့။ user ဆိုက input တောင်းမယ်.. ပြီးတွော operator တွေ အကုန် ပါအောင် script လေးရေးကြေည့်ရာရအောင်။

```

1 x = int(input("Enter number : "))
2
3 x += 1
4 print ("The result of of x += 1 => (x = x + 1)  is : " + str(x))
5
6 x **= 2
7 print ("The result of of x **= 1 => (x = x ** 1)  is : " + str(x))
8
9 x -= 1
10 print ("The result of of x += 1 => (x = x - 1)  is : " + str(x))
11
12 x *= 2
13 print ("The result of of x *= 2 => (x = x * 2)  is : " + str(x))
14
15 x /= 3
16 print ("The result of of x /= 3 => (x = x / 3)  is : " + str(x))
17
18 x %= 3
19 print ("The result of of x %= 3 => (x = x % 3)  is : " + str(x))
20
21 x //= 3
22 print ("The result of of x // = 3 => (x = x // 3)  is : " + str(x))

```

အပေါ်က code တွေကို run လိုက်ရင် အောက်ပါ အတိုင်း result ထွက်ပါတယ်။ ဟုတ် မဟုတ် ကိုတစ်ဆင့်ချင်း ပြန် ကြေည့် ရာရအောင်။



```
ekhaphy@MM00200588 ~
$ vim assi_opt.py

ekhaphy@MM00200588 ~
$ python assi_opt.py
Enter number : 3
The result of of x += 1 => (x = x + 1)  is : 4
The result of of x **= 1 => (x = x ** 1)  is : 16
The result of of x += 1 => (x = x - 1)  is : 15
The result of of x *= 2 => (x = x * 2)  is : 30
The result of of x /= 3 => (x = x / 3)  is : 10
The result of of x %= 3 => (x = x % 3)  is : 1
The result of of x //= 3 => (x = x // 3)  is : 0

ekhaphy@MM00200588 ~
```

Line 1 => ပထမဆုံး user ကနေ ရတဲ့ 3 ကို x ထဲကို ထည့်ပါတယ်။

Line 3 => $x += 1$ ဆိုတဲ့အတွက် x ထဲက 3 နဲ့ 1 နဲ့ ကိုပေါင်းပြီး x ထဲကိုပြန်ထည့်ပါတယ်။ ခုလက်ရှိ x ရဲ့ တန်ဖိုးသည် 4 ပါ။

Line 4 => x ထဲမှာ ခုလက်ရှိ ရှိနေတဲ့ တန်ဖိုးကို print ထုတ်ပါတယ်။ x သည် int ဖြစ်ပြီး " နဲ့ ရေးထားတာက တွေသည် string ဖြစ်တာမှား၊ data type တူအောင် str() ကိုသုံးပါတယ်။

Line 6 => $x **= 2$ ($x = x ** 2$) သည် X^2 ဖြစ်တာမှား လက်ရှိ x တန်ဖိုးက ငဲ ဆိုတော့ အဖြေက 16 ပါ။

Line 7 => x တန်ဖိုး ၁၆ ဖြစ်သွားတာကို print ထုတ်ပေးတာပါ။

Line 9 => $x -= 1$ ဆိုတော့ x(၁၆) ထဲက ၁ နှုတ်ပြီးတဲ့ x ထဲကိုပြန်ထည့်ပေးပါ။ လက်ရှိ x တန်ဖိုး ၁၅ ပါ။

Line 10 => ခုလက်ရှိ x ($x -= 1$ ငြောင့် ရလာတဲ့ x) ကို print ထုတ်ပါ။

Line 12 => $x *= 2$ ဆိုတော့ x(15) ကို 2 နဲ့မြောက်၏ အဖြေ 30 ကို x ထဲ ပြန်ထည့်

Line 13 => x ကို ထုတ်ပြ

Line 15 => $x /= 3$ မှာ x(၃၀) ကို ၃နဲ့ စားပြီး ရလာတဲ့ quotient(10) ကို x ထဲ ပြန်ထည့်

Line 16 => လက်ရှိ x တန်ဖိုးကို ထုတ်ပါ

Line 18 => $x \% 3$ ဆိုတော့ x(10) ကို 3 နဲ့ စားပြီး ရလာတဲ့ အကြွွင်း 1 ကို x ထဲ ထည့်ပါ။

Line 19 => ခုလက်ရှိ x ထဲမှာ ရှိနေတဲ့ data ကို print ထုတ်ပါ

Line 21 => $x // 4$ က ရှာတော့ x(1) ကို 3 နဲ့ စားတော့ 0 ပဲရပါတယ်။ 0 ကို x ထဲ ထည့်ပါ။

Line 1 => x တန်ဖိုးကို Print ထုတ်ပါ။

ဒုက္ခကာ ဒီလောက်ဆို သဘောပေါက်လောက်ပြီလို့ မျှော်လင့်ပါတယ်။

Logical Operator နဲ့ Bitwise Operator အကြောင်းကို နောင်ရာမှ ဆက်ရှင်းပါတယ်။ ခုလက်ရှိ အသုံးလို မပုံး Membership Operator နဲ့ Identity Operator အကြောင်းကို အရင် လွှေလာပါမယ်။

3.4 Python Membership Operator



Operator	Description
in	It returns true if value/variable is found in the sequence and false otherwise
not in	It returns true if value/variable is not found in the sequence and false otherwise

in ဆိတဲ့ Membership Operator သည် value တစ်ခု sequence ထဲမှာ ရှိမရှိ စစ်ပါတယ်။ စစ်လို့ ရှိရင် True မရှိရင် False ထုတ်ပေးပါတယ်။ ဥပမာ ပြောရရင် chir in house ? စစ်မယ့် ရာခိုင်တဲ့ ကောင်သည် chir ဖြစ်ပြီးတော့ ရှာရမယ့် နေရာသည် house ဖြစ်ပါတယ်။ in ဆိတဲ့ operatorဖြစ်တာဘူး။ ရှိရင် true ထွက်လာပါလိမ့်မယ်။ logic အနေနဲ့ ရှင်းပြတာပါ။.. အောက် ကုတ်လေးနဲ့ စမ်းကြာည့်ပါ။

```
>>> house = "At my house, we have 4 chir, 2 master bed room and..... "
>>>
>>> 'chir' in house
True
>>>
```

house ဆိတဲ့ string variable ထဲမှာ 'chir' ပါမပါကို စစ်ပါတယ်။ တွေ့တဲ့အတွက် true ရပါတယ်။ in operator သည် ရှိတာကို စစ်တာရိုး.. မပါတဲ့ 'TV' ဆိတာကို ရှာရင်တော့ မရှိတဲ့ အတွက် False ပါ။

```
>>> 'TV' in house
False
>>>
>>> 'TV' not in house
True
>>>
```

မရှိတာကို စစ်စေချင်ရင်တော့ not in Operator ကို သုံးပါတယ်။ အဲအခါဂျာရင်တော့ မရှိခဲ့ရင် True ရပါတယ်။

3.5 Python Identity Operator

Operator	Description
is	It returns true if two variables point the same object and false otherwise
is not	It returns false if two variables point the same object and true otherwise

Identity ဆိတာက္ခ.. သူသည် condition ကို စစ်တာမဟုတ်တော့ memory ပေါ်က object တစ်ခုရဲ့ ID ကို ဖြော်ပြုတာပါ။ နောက်သွားမယ်ထင်တယ်။ code လေးတွေနဲ့ တစ်ခုနှင့် ရင်းပြ ပေးသွား ပါမယ်။

```
>>> x = 5
>>> id(x)
2147535680L
>>
>>> y = x
>>> id(y)
2147535680L
>>
>>> z = 5
>>> id(z)
2147535680L
>>
>>> print('id of 5 =',id(5))
('id of 5 =', 2147535680L)
>>
```

`x = 5` ဆိုပြီးတော့ `x` တဲ့ကို Integer 5 ထည့်လိုက်ပါတယ်။ အဲနောက် memory ပေါ်မှာ 5 ရှိနေတဲ့ identity ကို `id()` ကို သုံးပြီးတော့ စစ်ဖြော်ပြုလိုက်ရင် 2147535680L ဆိုပြီး ရပါတယ်။ အဲတာကို `y = x` နဲ့ `x` ထဲကပေါင် ကို `y` ထဲကို ထည့်လိုက်ပါတယ်။ အဲနောက် `id(y)` ဆိုပြီးစစ်ရင်လဲ 2147535680L ဆိုတဲ့ ID ကိုပဲ ရပါတယ်။ ဒါဆို `z = 5` ဆိုပြီး `x` ထဲကနေ ထည့်တာမျိုးမဟုတ်ပဲ `int` 5 ကို `z` ထဲထည့်ရင်လဲ 2147535680L ဆိုတဲ့ ကောင်ပဲ ရပါတယ်။ ပြောချင်တာက ID သည် object တိုင်းအတွက် (ခု ဥပမာမှာဆို 5) unique ဖြစ်ပါတယ်။

- The `id()` function returns identity of the object. This is an integer which is unique for the given object and remains constant during its lifetime.

အဲဆို Identity operator လေးကို တစ်ခုက် ဆက်ဖြော်ရာအောင်။



```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
>>> x = 2
>>> y = 2
>>> z = 3
>>> print (x is y) #check if both point same memory location
True
>>> print (y is z)
False
>>> print (x is not y) #check if both point separate memory location
False
>>> print (y is not z)
True
Ln: 75 Col: 4
```

x ထဲမှာလဲ 2 ရှိတယ်။ y ထဲမှာလဲ 2 ရှိတယ်။ x is y ဆိုပြီးတော့ x ရဲ့ identity နဲ့ y ရဲ့ Identity တူလား စ စိတ္တာ True ဆိုပြီး အဖြေရပါတယ်။ 2 ဆိုတဲ့ ကောင်ရဲ့ id သည် x ထဲမှာပဲ ရှိရှိ y ထဲမှာပဲ ရှိရှိ သည်တစ်ခု ပဲမို့ အတူတူ ဖြစ်တာကြောင့် true ဆိုပြီး ရတာပါ။ y is z မှာဂျာ 2 နဲ့ 3 မို့ မတူတဲ့ အတွက် False ရပါတယ်။ y is not z ရှာတော့ မတူတာ ဟုတ် မဟုတ် စစ်တာမို့.. မတူတဲ့အတွက် False ရပါတယ်။

4. Python Decision maker

Python ရဲ့ decision maker မှာ ဘာတွေပါလဲဆိုတော့ if, if .. else, if .. elif .. else, nested if တို့ ပါဝင်ပါတယ်။ အဲကောင်တွေကို တစ်ခုချင်း လေ့လာကြော်ရာရအောင်။

4.1 if statement

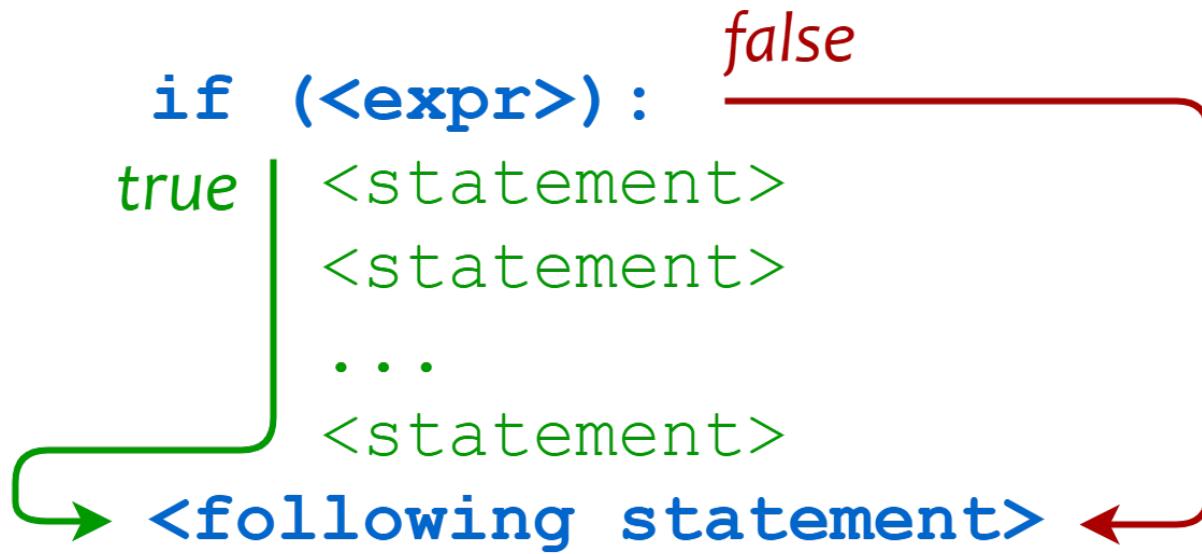
```
x = input("please enter number : ")
if (x > 5):
    print ("x is greater than 5")

if (x < 5):
    print ("x is less than 5")

print ("Program is finished")
```



If သည် သူ့နောက်က condition မှန်မှ (true ရမှ) သူနဲ့ သက်ဆိုင်တဲ့ (Block of code) statement ပေါ်
ကို ဆက်ပြီး အလုပ် လုပ်ပါတယ်။ condition မကိုက်ရင် (False ရရင်) .. သက်ဆိုင်ရာ if ခဲ့ statement
တွေကိုရော်သွားပြီးတော့ အောက်က ကောင်တွေကို ဆက်လုပ်ပါတယ်။



ဒု အပေါ်က ကုတ်လေးမှာဆိုရင် input တစ်ခုတောင်းလိုက်ပါတယ်။ အဲလိုတောင်းလိုက်လို့ ရလာတဲ့ ၆
ကာင်သည် 5 ထက်ကြိုးရင် x is greater than 5 ဆိုပြီး result ထွက်လာပါလိမ့်မယ်။ 5 ထက် ငယ်ရင် x is
less than 5 ဆိုတဲ့ result ကို ရပါလိမ့်မယ်။ line 1 မှ input တောင်းတဲ့ ကောင်ကတွေရှင်းပါတယ်။ အော
က် if သည် သူ့နောက်က () ကွင်းစ ကွင်းပိတ် ထဲက condition မှန်မှ True ဖြစ်မှ သူ အောက် statement
တွေက အလုပ် လုပ်ပါတယ်။ if ခဲ့နောက်မှာ : (full column) လိုက်ပါတယ်။ ပြီးတော့ tap တစ်ခါ ခုန်ပါတယ်။
ပြီးမှ print ထုတ်ရမယ့်ကောင်ကို ရေးပါတယ်။ ဘာလို့ လဲဆိုတွေ့ Python သည် Indentation နဲ့ အလုပ် လု
ပ်လိုပါပဲ။ ဒု code ကို အရင် run ပြီး စမ်းစစ်ခြားစွာပါစိုး.. ပြီးမှ indentation ကို ဆက်ပြီး လေ့လာပါမယ်။

serial.py	variable.py	IF_else.py
<pre> 1 x = int(input("please enter number : ")) 2 if (x > 5): 3 print ("x is greater than 5") 4 5 if (x < 5): 6 print ("x is less than 5") </pre>		<pre> C:\Python27\python.exe please enter number : 10 x is greater than 5 Process returned 0 (0x0) Press any key to continue . . . </pre>

Program ကို run လိုက်တဲ့ အချိန်မှာ..

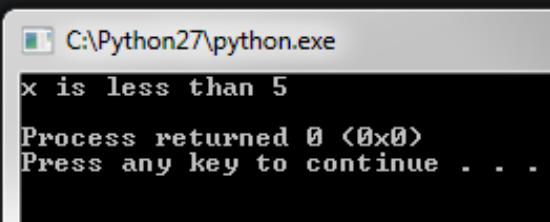
Line1 => input တစ်ခုတောင်းလို့ 10 ထည့်ပေးလိုက်ပါတယ်။ x ထဲကို int ပြောင်းပြီး ထည့်လိုက်ပါတယ်။



Line2 => ရလာတဲ့ x(10) သည် 5 ထက်ကြီးလား စစ်ပါတယ်။ int နှင့်စစ်လိုရတာပါ။ str ဆို error တက်ပါလိမ့်မယ်။ x က ကြီးတာမို့ True ရတဲ့ အတွက် **Line3 =>** if အောက် က statement သည် အလုပ်လုပ်ဖြီး "x is greater than 5" ဆိုပြီး ထွက်လာပါလိမ့်မယ်။

Line5 => if(x < 5) ဆိုပြီး x ။ ၅ ထက်ငယ်လား စစ်ပါတယ်။ ကြီးတာမို့ condition မကိုက်တဲ့ အတွက် false ရပါတယ်။ အဲဒြောင့် if(x<5) နဲ့ ဆိုင်တဲ့ statement တွေကိုဆက် မ run တော့ပဲ ထွက်သွားပါတယ်။

ဟုတ်ပြီ input လဲ မထောင်းတွောဘူး.. X is less than 5 ဆိုတာ ထွက်ချင်ရင် ဘယ်လို့ လုပ်မလဲ.. လုပ်လို့ ရတာတွေထဲကမှ လွှာယ်တာကတွော if နောက်က condition ကို True / false တန်းပေးလိုက်တာပါပဲ။



```

1  #x = int(input("please enter number : "))
2  if (False):
3      print ("x is greater than 5")
4
5  if (True):
6      print ("x is less than 5")
7

```

ကို Python ရဲ့ line of code တွေကို မလုပ် မလုပ်စေချင်တာ run တဲ့ အခါမှ ထည့်မ Run စေချင်တာမျိုးအတွက် လဲ သုံးပါတယ်။ If သည် သူ့နောက် condition မှန်မှ အလုပ်လုပ်တာမို့ if (True): နဲ့ သက်ဆိုင်တဲ့ ၆ ကာင်ပဲ အလုပ် လုပ်လို့ x is less than 5 ထွက်လာတာပါ။

Python Indentation

Python သည် တစ်ခြား programming Language တွေဖြစ်တဲ့ Java, C, C++, etc.. တို့လို ({ }) တွန်းကွင်းစ တွန်းကွင်းပိတ် တွေ သုံးပြီးတော့ code block ကို မသတ်မှတ်ပါဘူး။ Block of code ဆိုတာက Class တွေ function တွေရေးတဲ့အခါဂ္ဂ .. အဲကောင်တွေနဲ့ သက်ဆိုင်တဲ့ code တွေသည် ဘယ်ကနေ ဘယ်ထိ ဆိုတာကို နယ်မြေသတ်မှတ်ပေးဖို့ { } တွေကို သုံးရပါတယ်။ ခု အောက် ပုံ C Plus Plus မှာ if ကို သုံးထားပုံပါ။



```

1 #include <iostream>
2 using namespace std;
3
4 int main() // start of main()
5 {
6     int a = 15, b = 20;
7
8     if (b > a) // start of if
9         cout << "b is greater" << endl;
10    } // end of if
11    system("PAUSE");
12 } // end of main()
13

```

အပေါ်က ကုတ်တွေမှာဆို အရင်ဆုံး အလုပ် လုပ်မထဲ့ Main အတွက် { } ပိတ်ပါတယ်။ အဲထဲမှာမှ သူနဲ့ သက်ဆိုင်တဲ့ code တွေရှိတယ်ပေါ့။ if (b > a) ဆိုရင်လဲ condition မှန်ရင် လုပ်မထဲ့ ကောင်သည့် { } ထဲ မှာ ရှိနေတာပါ။ ပြောချိတာက condition မှန်ရင် အလုပ်လုပ်မထဲ့ ကောင်သည့် { } ထဲမှာ ရှိနေတဲ့ code တွေ အကုန်ပါ။ c, C++, java, စိတဲ့ ကောင်တွေမှာ tab ခုနဲ့ space ခြားတာသည့် အမြင်လှအောင် ကုတ်ကို ရှင်းရှင်းမြင်အောင် လုပ်ပေးတာပဲ ရှိတယ်။ space မခြား tab မခုန်လဲ မှန်ကန်စွာ အလုပ် လုပ်ပါတယ်။ Python မှာ ရွေတွာ { } တွေ မသုံးတွောပဲ white space (space /tab) နဲ့ အလုပ် လုပ်ပါတယ်။ Pytheon မှာ white space (indentation) သည့် အရမ်း အရေးပါ ပါတယ်။ Python indentation ကို နားလည့် အောင် ကုတ်လေးကို တစ်ချက် ဖြောည့်ရှုပို့။

```

1 x = int(input("please enter number : "))
2 if (x > 5):
3     print ("x is greater than 5 - first time")
4     print ("x is greater than 5 - second time")
5     print ("x is greater than 5 - third time")
6
7 if (x <= 5):
8     print ("x is less than or equal 5 - first time")
9     print ("x is less than or equal 5 - second time")
10    print ("x is less than or equal 5 - third time")
11
12 print("** Program is finished **")

```

ခဲ့ အပေါ်က ကုတ်မှာဆို Line 1 ⇒ သည့် input ကောင်ရှိ ရလာတဲ့ data ကို variable ထဲ ကို ထည့်တာပါ။



Line 2 ⇒ မှာ if ($x > 5$) : ဆိုပြီး condition ကို စစ်ပါတယ်။ ရလာတဲ့ x ထဲက ကောင်သည့် 5 ထက်ကြီးရင် သူ။ အောက် သုံးဇြောင်း အလုပ် လုပ်မှာပါ။ ဘာလို့ လဲဆိုတွော သူတို့ သည် indentation Level တူနေတာ မို့ပါ။ ဒောက်ကနေ လုပ်ချင်တဲ့ code တွေကို indentation level (number of space) တူအောင် ထားပေး၍ ပေါင်းဖြင့် same block of code ဆိုတာကို Python က နားလည်းပါတယ်။ { } တွေ မလို ပါဘူး။ အကယ်လို့ if () နောက်က condition မှာခဲ့ရင် သူ။ ဒောက်က သူနဲ့ သက်ဆိုင်ရာ ကုတ်တွေကို မလုပ်တွောပဲ.. အောက် ဆက် ဆင်းသွားပါလိမ့်မယ်။ **Line 7** ⇒ if ($x \leq 5$): မှာ condition မှန်ရင် indentation level တူတဲ့ အောက်ကသုံးဇြောင်းကို လုပ်ပြီး .. အောက်ဆက်ဆင်းသွားပါလိမ့်မယ်။ condition false ခဲ့ရင်တွော အဲ သူနဲ့ သက်ဆိုင်ရာ သုံးဇြောင်းကို မလုပ်တွောပဲ ငြို့သွားပြီးတွော အောက်ရှာနဲ့ နေတဲ့ အဇြောင်းတွေကို ဆက်ပြီး အလုပ်ပဲ လုပ်ပါလိမ့်မယ်။

ဒီမှာဆိုရင်တွော **Line 12** ⇒ “** Program is finished **” ပေါ့။ ဒီနောက်ဆုံးလိုင်းသည် if နဲ့ condition စစ်တဲ့ အထဲ မပါတာမို့ သူသည် ဘယ်ကောင် လုပ်လုပ် မလုပ်လုပ် သူကတွော print ထုတ်မှာပါပဲ။

```

1 x = int(input("please enter number : "))
2 if (x > 5):
3     print ("x is greater than 5 - first time")
4     print ("x is greater than 5 - second time")
5     print ("x is greater than 5 - third time")
6
7 if (x <= 5):
8     print ("x is less than or equal 5 - first time")
9     print ("x is less than or equal 5 - second time")
10    print ("x is less than or equal 5 - third time")
11
12 print("** Program is finished **")

```

C:\Python27\python.exe
please enter number : 3
x is less than or equal 5 - first time
x is less than or equal 5 - second time
x is less than or equal 5 - third time
** Program is finished **
Process returned 0 <0x0> execution time
Press any key to continue . . .

ခုလောက်ဆို python ရဲ့ indentation ကို မျက်စိလဲ သဘောလောက်တွော မြင်ပြီးလို့ ယုံပါတယ်။ နောင် class ကွဲ function တွေရဲ့ indentation(white space) အဇြောင်းကို ပိုသိလာပါလိမ့်မယ်မှာ။။။



Other Program like C, Java use { } to define Block of Code

```

1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {                                // start of main()
6     int a = 15, b = 20;
7
8     if (b > a) {}                  // start of if(b>a)
9     cout << "b is greater - first time";
10    cout << "b is greater- second time"; Block of
11    cout << "b is greater - third time"; Code
12 }                                // end of if(b>a)
13
14     if (b < a) {}                // start of if(b<a)
15     cout << "a is greater - first time";
16     cout << "a is greater- second time"; Block of
17     cout << "a is greater - third time"; Code
18 }                                // end of if(b<a)
19 system("PAUSE");                 // end of main()
20 }
21 
```

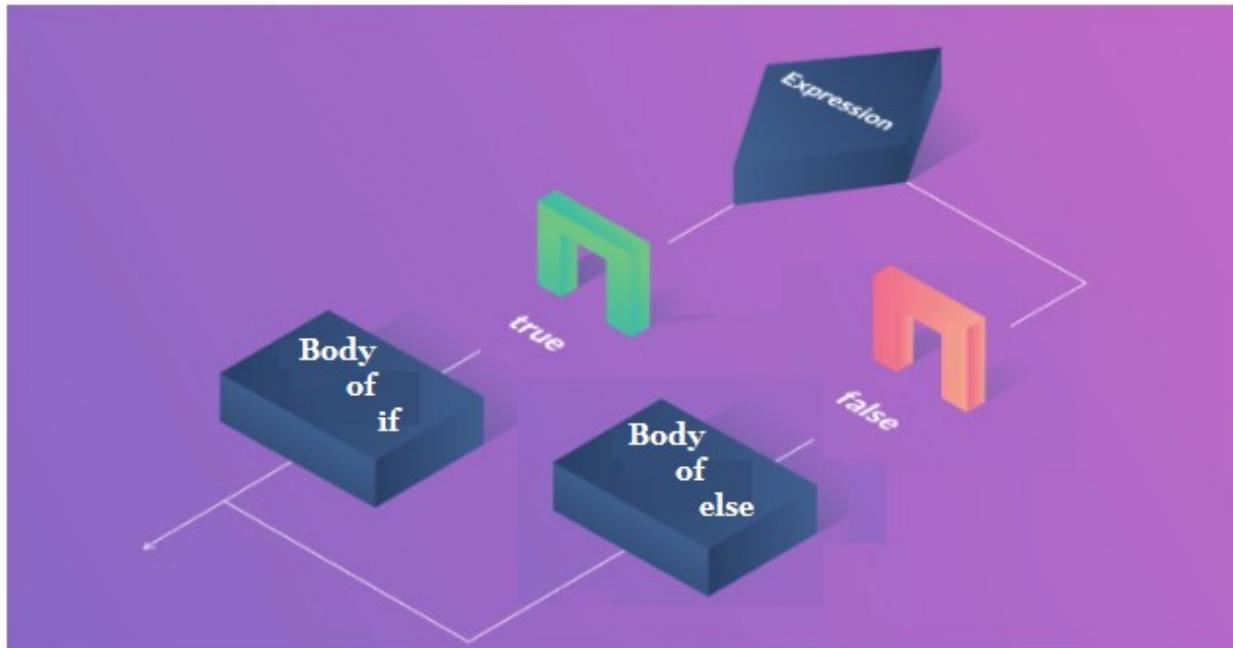
Python use indentation (white space) to define Block of Code

```

1 x = int(input("please enter number : "))
2 if (x > 5):
3     print ("x is greater than 5 - first time") Block of
4     print ("x is greater than 5 - second time")
5     print ("x is greater than 5 - third time")
6
7 if (x <= 5):
8     print ("x is less than or equal 5 - first time")
9     print ("x is less than or equal 5 - second time")
10    print ("x is less than or equal 5 - third time")
11
12 print("** Program is finished **")
13 
```

4.2 Python if else statement

Python ရဲ့ if .. else statement မှာ if statement နဲ့အဓိက ကွာခြားထဲ အချက်ကတော့ if သည် condition မှန်ရင် လုပ်မယ် မမှန်ရင် မလုပ်ဘူး ကြောသွားတယ်။ if .. else ရာတွေ့ စစ်တဲ့ condition မှာ မှန်တယ်(True)ဆိုရင် ဘာလုပ်မယ် (if statement) မှားတယ်(False)ဆိုရင် ဘာလုပ်မယ် (else statement) ဆိုတဲ့ ပုံစံပါ။



ခု ဘီယာဝယ်ဖို့ အသက်စစ်တဲ့ program လေးတစ်ခု ရေးကြော်ဗျာရအောင် ဖာ။

User ဆီကအနေ အသက်မေးမယ်။ အသက်ပေါ်မှုတည်ပြီး 18 နှစ်နဲ့ အထက်ဆို beer ဝယ်လို့ရမယ်။ 18 နှစ် ငံ အက်ဆိုရင်တွော ဘီယာမရောင်းဘူး အိမ်ပြန်ပါဆိုပြီး print ထုတ်မယ်ပေါ့ဖာ။။။ အောက်ကာကုတ်လေးကို စမ်းကြော်ဗျာရအောင်။

```

1 Age = int(input("what is your age : "))
2
3 if(Age >= 18):
4     print("Yes")
5     print("You can buy Beer")
6 else:
7     print("You can't buy Beer")
8     print("Hey, kid just go back home")

```

C:\Python27\python.exe
 what is your age : 18
 Yes
 You can buy Beer
 Process returned 0 (0x0)
 Press any key to continue . .

ရေးပြီး run လိုက်လို့ အသက်မေးရင် ၁၈ လို့ ထည့်လိုက်ရင် if(Age >= 18) condition ၏ true ဖြစ်တာမို့ သူ အောက်က code နှစ်ကြောင်းအလုပ်လုပ်ပါတယ်။

အကယ်လုံး 10 လို့ ရိုက်လိုက်ရင် ?



```
1 Age = int(input("what is your age : "))
2
3 if(Age >= 18):
4     print("Yes")
5     print("You can buy Beer")
6 else:
7     print("You can't buy Beer")
8     print("Hey, kid just go back home")
```

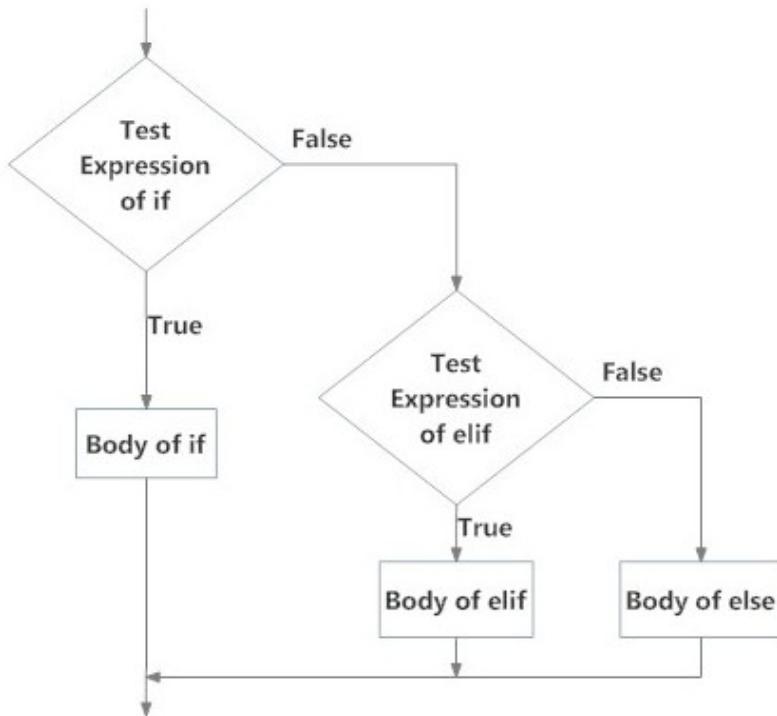
```
C:\Python27\python.exe
what is your age : 10
You can't buy Beer
Hey, kid just go back home
Process returned 0 (0x0)
Press any key to continue . . .
```

Age ≤ 10 ቁጥር የሚችሉ if(Age ≥ 18) ነው፡፡ condition ወይም False ቁጥር የሆኑን የሚከተሉትን else statement ጥሩን ለማድረግ ለማስታወሻ ይችላል፡፡

** access control for User - Exercises**

```
1 # Allowed users to login
2 allowed_users = ['bill', 'steve']
3
4 # Get the username from a prompt
5 username = raw_input("What is your login? : ")
6
7 # Control if the user belongs to allowed_users
8
9 if username in allowed_users:
10     print "Access granted"
11
12 else:
13     print "Access denied"
```

4.3 Python if .. elif .. else statement



ဒါ if elif else အလုပ်လုပ်ပုံသည် စစ်ရှင်း if နောက်က condition ကို စစ်ပါတယ်။ မှန်ရင် if-statement ကို run ပြီး if code block ကနေထွက်သွားပါတယ်။ false ဖြစ်ခဲ့ရင် အဲနောက် ပထမ elif စစ်ပါတယ်။ မှန်ရင် ပထမ elif- statement ကို run ပြီး if code block ကနေထွက်သွားပါတယ်။ false ဖြစ်ခဲ့ရင် အဲနောက် ဒုတိယ elif စစ်ပါတယ်။ မှန်ရင် ဒုတိယ elif- statement ကို run ပြီး if code block ကနေထွက်သွားပါတယ်။ အဲလိုနဲ့ နောက်ဆုံးမှာတွော condition မစစ်တွောပဲ else ရဲ့ statement ကိုထုတ်ပေးပါတယ်။ ဘဖတ်ရတာမူ နဲ့ ရူတ်သေးတယ် အောက်က ကုတ်လေးကို ဖတ်ပြီး ရေးကြော်း run ကြေည့်လိုက်ပါ။

```
1 x = int(input("What is the time? : "))
2
3 if x <= 10:
4     print ("Good morning")
5 elif x < 12:
6     print ("Soon time for lunch")
7 elif x < 18:
8     print ("Good day")
9 elif x < 22:
10    print ("Good evening")
11 else:
12    print ("Good night")
```



ကိုယ်စစ်မယ့် condition စစ်မယ့် procedure ကို အစဉ်လိုက်ရေးသင့်တယ်။ ပြောချင်တာက logic ကို စနစ်ကျစေချင်တာပါ။ ခု ဒီ example မှာဆို ပထမဆုံး 10 နဲ့ စစ်တယ် ပြီးမှ 12 နဲ့ စစ်တယ်... အဲနောက် 18, 22 အကုန် တို့လိုက်စစ်သွားတာပါ။

```

1 x = int(input("What is the time? : "))
2
3 if x <= 10:
4     print ("Good morning")
5 elif x < 12:
6     print ("Soon time for lunch")
7 elif x < 18:
8     print ("Good day")
9 elif x < 22:
10    print ("Good evening")
11 else:
12    print ("Good night")

```

ခုဆိုရင် ဖော် အဆင်ပြေနောက်လမ့်မယ်။ အကယ်လို့ $x < 22$ သည် အစီအစဉ် တရာ့ (logic ၏) မဟုတ်ပဲ အပေါ် ဆုံးရောက်နေမယ် ဆိုရင် ?

```

1 x = int(input("What is the time? : "))
2
3 if x <= 22:
4     print ("Good morning")
5 elif x < 12:
6     print ("Soon time for lunch")
7 elif x < 18:
8     print ("Good day")
9 elif x < 10:
10    print ("Good evening")
11 else:
12    print ("Good night")

```

ပုံမှာ မြင်တဲ့အတိုင်းပဲ.. $x <= 22$ ဆိုတဲ့ condition နဲ့ ကိုယ်နေသမှ သူအောက် statement ကိုပဲ လုပ်ပြီး if ခဲ့ block of code ကနေထွက်သွားပါတယ်။ ;-)

4.4 Nested if statement

Nested if ဆိတာက ifအောက်မှာမှ if condition ထပ်စစ်ထားတာပါ။ if code block (Green Block) အောက်မှာ နောက်ထပ် if code block (Red Block) ထပ်ရှုနေတာပါ။

```

1 #Boy can buy beer if their age is 18 or over
2 #Girl can only buy beer if the age is over 20 :P
3
4 Age = int(input("Enter your age : "))
5 Gender = input("Enter your Gender (Male/Female) : ")
6
7 if Age >= 18:
8     if Gender=='Male':
9         print("Your can buy beer")
10
11 if Age > 20:
12     if Gender=='Female':
13         print("You can buy beer")
    
```

ယောကျိုးလေးဆိုရင် အသက်ဘက် နှစ်နဲ့ အထက် ဖြစ်ရင် beer ဝယ်လိုဂေါ်။ မိန်းကလေးဆိုရင်တွော အသက် ၂၀ ရောက်မှ ဝယ်လိုဂေါ်။ အဲဒေါက်တွေကို user ဆီကနေ Age နဲ့ Gender ကို Input အနေနဲ့ တောင်းလိုက်ပါတယ်။ အဲတွေက် ပထမ if block သည် အသက်ကို စစ်ပြီးမှ နောက် if block သည် Gender ကို စစ်ပြီး အဖြေ result ပြန်ပေးပါတယ်။

Condition True ဖြစ်ရင်သာ output မြင်ရတာပါ။ condition False ဖြစ်ရင်တွော ဘာမှ ထွက်မလာပါဘူး.. အဲတွော နဲ့ လေး ထပ်ပြုရေးကြေည့်ရွာရအောင်။



```

1 #Boy can buy beer if their age is 18 or over
2 #Girl can only buy beer if the age is over 20 :P
3
4 Age = int(input("Enter your age : "))
5 Gender = input("Enter your Gender (Male/Female) : ")
6
7 if Age >= 18:
8     if Gender=='Male':
9         print("You can buy beer")
10    if Age > 20:
11        if Gender=='Female':
12            print("You can buy beer")
13    else:
14        print ("Sorry, you can't buy beer")

```

အပေါ်က execise ကုတ်မှာဆို if statement ပေါ်မှာ if statement တွေ ကြိမ်ကြိမ် ထပ်ပြီး စစ်ထားတာပါ။ ဘာနဲ့တူလဲဆိုတော့ filter တွေ တစ်ခုပြီး တစ်ခု ခံပေးထားသလိုပေါ့။ ဥပမာ hall ခန်းထဲက လူတစ်ယောက် ကိုရှာမယ်ဆိုရင် အရင်ဆုံး အသက် ၂၀ ကနေ ၄၀ ငြား လူတွေ ကို ဖြေည့်မယ်ပေါ့။ အဲ အသက် ၂၀ - ၄၀ လူ ထဲကမှ ဝတဲ့လူတွေကို ထပ်ဖြာည့်မယ်ပေါ့။.. အဲ လူစုထဲက က အရိုအနီး ပတ်ထားတဲ့ သူတွေကို ရွှေးထုတ်မယ်ပေါ့။ အဲဆုံး ၃၂ တွေရှာနေတဲ့ သူသည် အသက် ၂၀ နဲ့ ၄၀ ငြားက ခပ်တေ အကျိုအနီး ပတ်ထားတဲ့ သူပေါ့.. ။။ လိမ့်မျိုးပဲ.. ခုလဲ အထပ်ထပ် စစ်သွားတဲ့ သဘောပါပဲ။။

ခု စစ်တဲ့ပုံစံက boy or girl >= 18 ကြောမကြော အရင်စစ်.. ကြောရင် first priority သည် male လား စစ် male ဟုတ်တယ် မှန်တယ်ဆိုရင် print ထုတ် .. မဟုတ်ရင် အသက်ကို ထပ်စစ် ၂၀ ကြောလား ဆိုပြီးတော့.. ၆ ရှားတယ် ဆုံးမှ female ဖြစ်မှ print ထုတ်.. အဲချက်တွေနဲ့ မကိုက်ခဲ့ရင် beer ဝယ်မရပြောင်း ပြန်ပြော..

5. Python Loops

Python မှာ loop ပါတယ်။ ရတဲ့ type တွေကတော့ အောက်ပါအတိုင်းပဲ ဖြစ်ပါတယ်။

- For Loop
- While Loop
- Nested loop

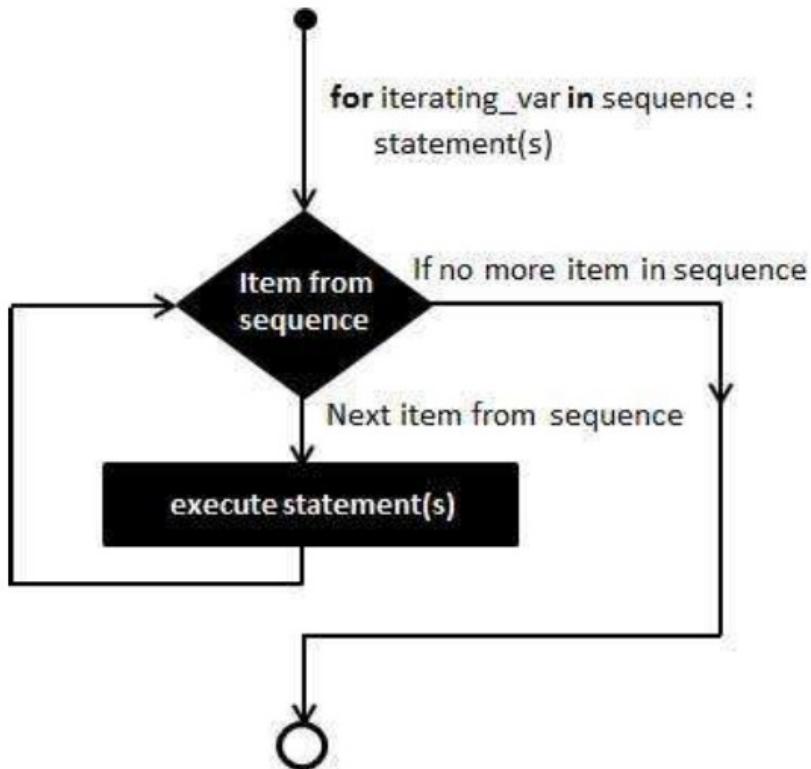
5.1 Python for loops

Python မှာ for loop သည် sequence နဲ့အလုပ်လုပ်ပါတယ်။ ခုအောက်မှာ နမူနာ code ရိုက်ပါတယ် ရိုက်ပြီး run ကြေည့်ပါ။ Python သည် simple English ရေးနေရသလိုပါပဲ.. ပြီးတော့ သူရဲ့ indentation တွေ ကလဲ.. Paragraph တွေလိုပါပဲ။။။ ရေးရင် သူ့ သဘော သဘာတ ကို နားလည်လာပါလိမ့်မယ်။

<pre>serial.py variable.py</pre>	<pre>h e l l o Process returned 0 (0x0)</pre>
<pre>1 for character in 'hello': 2 print(character) 3</pre>	

Python မှာက string ထဲမှာ ရှိတဲ့ character တွေကို sequence အနေနဲ့ သိမ်းဆည်းပါတယ်။ **for character in 'hello':** မှာ For loop အတွက် variable သည် **character** ဆိုတဲ့ ကောင်။ ဒဲကောင်ထကို **in** ဆိုတဲ့ keyword နောက်က ကောင်ကို sequence အလိုက်ထည့်ပေးပြီး Loop ပါတ်သွားတာ.. Sequence ဆုံးရင် for loop သည် stop သွားပါလိမ့်မယ်။

ဒဲ run လိုက်တဲ့ အစိန်မှာ first word ဖြစ်တဲ့ h ကို character ဆိုတဲ့ variable ထဲ ထည့်ပါတယ်။ ပြီးတော့ `print(character)` ဆိုပြီး variable ထဲမှာရှိတဲ့ data ကို print ထုတ်ပါတယ်။ ပြီးတော့ second word ဖြစ်တဲ့ e ကို character ထဲ ထည့်ပါတယ် ဒဲနောက် `print` ထုတ်ပါတယ်။ ဒဲလိုနဲ့ last word ဖြစ်တဲ့ o ကိုထုတ်ပြီးနိုင်မှာတော့ ကုန်သွားပြီ ဖြစ်လို.. For loop သည် stop သွားပါလိမ့်မယ်။ အောက်က flow chart လေးကိုကြေည့်လိုက်ရင် ရှင်းသွားပါလိမ့်မယ်။



နောက်ထပ် Example တစ်ခု ကို ထပ် စမ်းကြေည့် ရှာရအောင်။ ဒီတစ်ခါတွာ list/array တွေဘက် ကို လုပ်ဖြီး စမ်းကြေည့်ရှာရအောင်။

```

>>> mylist = ['book', 'orange', 'cloud']
>>> for x in mylist:
... [tab] print x
... [enter]
book
orange
cloud
>>>

```

ပထမဆုံး list တစ်ခု create လုပ်လိုက်ပါတယ်။ ပြီးတွေ့ list ထဲ က ကောင်တွေကို for loop ပါတ်ပိုး ထုတ်ချင်တာဖြစ်လို့.. **for x in mylist:** လို့ ရေးလိုက်ပါတယ်။ ဒီနေရာမှာ တစ်ခုမှတ်ထားရမှာက enter ငဲ ပါက်လိုက်ရင် . . . (အစက်သုံးစက်) ပေါ်လာပါလိမ့်မယ်။ space တစ်ခုက်ဖြစ်ဖြစ် tab တစ်ခါ ဖြစ်ဖြစ် ခုနှစ်ပေးပါ။ python သည် { } ထက် white space (indentation) နဲ့ လုပ်တာမို့ပါ။ အဲလို့ tab တစ်ခါ ခုနှစ်လိုက်ချင်းဖြ မြှော်၏ For loop ရဲ့ block of code နယ်မြေကို သတ်မှတ်လိုက်တာပါ။ ခုတွေ့ဗျားများပေါ်မယ့် code တစ်ပေါ်ကောင်းပဲ ရှိတာမို့ ရေးပြီး enter ခေါက်လိုက်ပါ။ အကယ်လို့ တစ်ကြောင်းမက ထဲ code တွေ for loop ထဲမှာ ငဲ ရေးရမယ့် ဆိုရင် line of code တိုင်းကို same indentation level တူဇော် ထားပေးရပါမယ်။ same indentation level ဆိုတာက space တစ်ခါ ဆို အကုန် တစ်ခါပဲ ခြားတာ.. [tab] ခုနှစ်ရင်လဲ အကုန် အတူတူ

ခုန်တဲ့ range အတူတူ ထားတာကို ခေါ်တာပါ။ တစ်ဌား programming Language (Like c++,java,etc) ငဲ တွေတူန်းက if တို့ for loop တို့ code block ကို ဘယ်က စလဲဆိုတာကို { နဲ့ သက်မှတ်ပြီး အပိတ်ကိုရွှေတွာ } နဲ့ သက်မှတ်ပါတယ်။ Python ။။ (Full-column) နောက်က same white space ရှိနေဖို့ range တူနေတဲ့ code တွေသည် code block ဖြစ်တယ် လို့ သတ်မှတ်ပါတယ်။.. ကဲ လေရှည်နေတာ နဲ့ များသွားပြီ ပြန်ဆက်ပြုကြည့်ရအောင်။ အဲလို Print ထုတ်လိုက်တာနဲ့.. List ထဲက ကောင်တွေသည် sequence အလိုက် တစ်ခု ပြီး တစ်ခု ထွက်လာပါလိမ့်မယ်။ ;-)

For loop အကြောင်း ပြောမယ်ဆိုတဲ့ .. သူနဲ့ example အနေနဲ့ တွဲ သုံးမထုံး range() အကြောင်းလေး အရင် ပြောပါရတော့။ range() function မှာ argument သုံးစုပ်တင်ပါတယ်။ သူ့ရဲ့ argument သုံးခုကတွေ့။ range(start, stop[, step]) တို့ပဲ ဖြစ်ပါတယ်။ start နဲ့ step သည် optional အနေနဲ့ တည်ရှိပါတယ်။ ထည့်မရေးရင် default အတိုင်း အလုပ် လုပ်ပါတယ်။ default အနေနဲ့ က start =0, step =1 တို့ပဲ ဖြစ်ပါတယ်။ stop ကတွေ့ ထည့်ကို ရေးပေးရမှာပါ။ အောက်က ဥပမာ လေးကို ဖြော်ပါ။

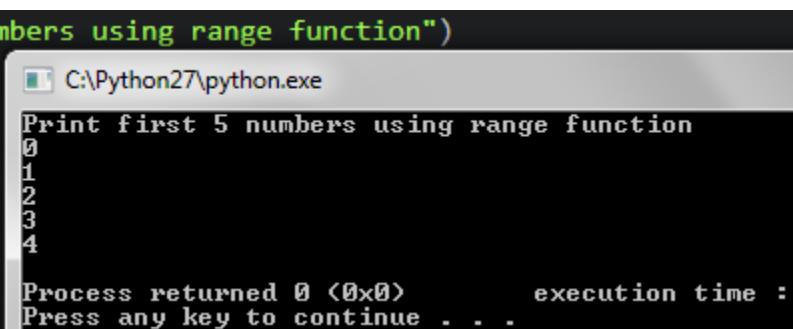
```
>>> range(5)          # range(0,5,1)
[0, 1, 2, 3, 4]
>>>
>>>
```

Range(5) လို့ရေးလိုက်တာနဲ့ default အတိုင်း start=0 ဖိုးရှိုးကနေစမယ် .. ပြီးတွေ့ sequent သည် step=1 မို့ ၁ တိုးတိုးသွားပါမယ်။ အကယ်လို့ start, stop, step တွေ အပြည့်အစုံ ရှိက်ရင်လဲ result ။ အတူတူပါပဲ။

```
>>>
>>> range(0,5,1)
[0, 1, 2, 3, 4]
>>>
```

ဟုတ်ပြီး အဲတွေ့ execise အနေနဲ့ for loop ပါတ်ပြီး print ထုတ်ရှာ့ရအောင်။

```
print("Print first 5 numbers using range function")
for i in range(5):
    print(i)
```





`range(5)` ဆိတဲ့ အတွက် 0 to n-1 အထိ `print` ထုတ်သွားပါတယ်.. ပထမ 0 သည် `variable i` ထဲ ကို ရောက်တယ်။ `print(i)` ဆိုပြီး ထုတ်တယ် အဲနောက် နောက်တစ်ခြားငါးတယ်။ ဒုတိယ 1 သည် i ထဲကိုရောက်တယ်။ `print` ထုတ်တယ်။ ငဲ ရောက်တော့ `sequence` ကုန်သွားပြီးမို့။ အဲ for loop ကနေ ထွက်သွားပါတယ်။ ပိုပြီး မြင်သာအောင် C++ ၏ for loop နဲ့ နှင့်ယုဉ်ပြောညွှန် ရှာရအောင်။ C++ တို့ရာတွေ ရောင်းတွေမှာ သင်ရတဲ့ အထဲ ပါတွေ ပိုမြင်သာအောင် နှင့်ယုဉ်ပြောနော.. Exactly the same မဟုတ်ဘူးနော..

For loop in C++

```
cout << "Print first 5 numbers";
for (i=0; i<5; i++) {
    cout << i;
}
```

For loop in Python

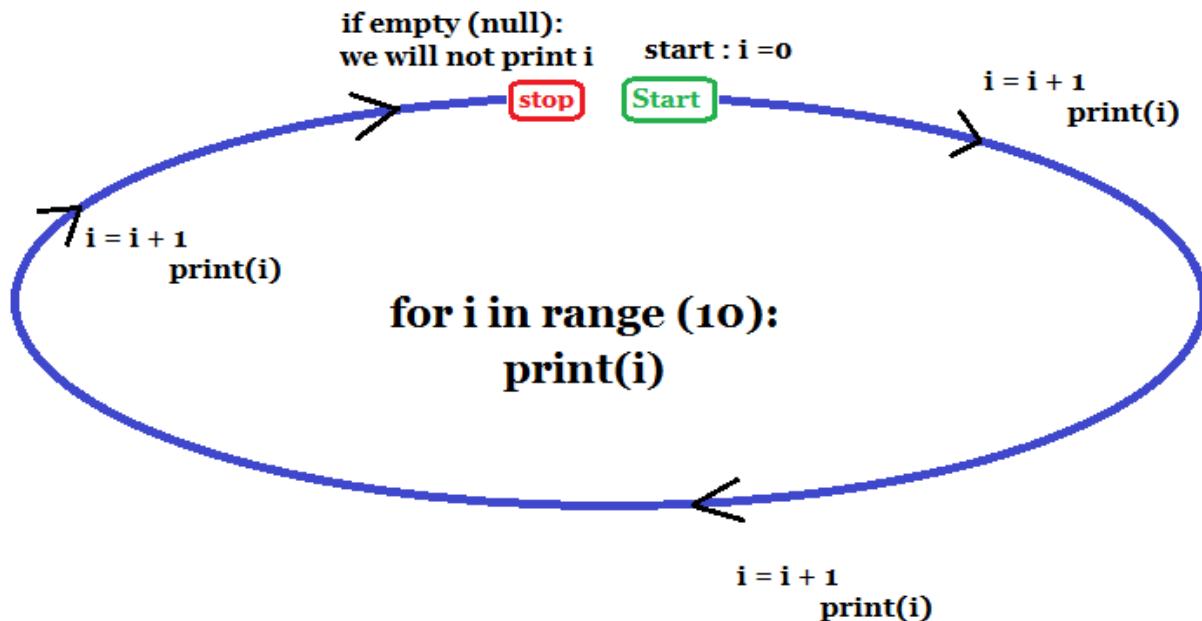
```
print("Print first 5 numbers")
for i in range(5):
    print(i)
```

C++ မှာရာတွေ `i=0` ဆိုပြီးတော့ `i` ကို zero က စမယ်ပြောတယ်။ ဘယ်နှင့်မှာ for loop ပြီးမလဲဆိုင် ။။။ `i<5` ဆိုတော့ `i=4` ထိအလုပ် လုပ်ပြီး `i=5` ဖြစ်တဲ့ အချိန်မှာ stop မယ်။ loop ပါတ်နေတာဖြစ်တဲ့ အတွက် တစ်ကြိမ် ပါတ်ပြီးတိုင်း `i++` ဆိုတဲ့ အတွက် `i` ကို 1 ပေါင်းပေါင်းပြီး တိုးသွားမယ်။

အဲလိုပဲ Python မှာရာတွေ `i` ဆိုတဲ့ `variable` ရှိတယ်။ `variable` ဖြစ်တဲ့ အတွက် ဘူးထဲ ထည့်ရမယ့် `data` ကို ဘယ်က ယူရမလဲ ဆိုတာကို `in` ဆိုတဲ့ keyword လေးနဲ့ `range(5)` ကို ညွှန်ပေးထားတယ်။ `range(5)` သည် default value တွေပါ မြင်သာအောင် ထည့်ရေးရင် `range(0, 5, 1)` ဖြစ်ပါတယ်။ 0 ကစမယ် 1 ပေါင်းသွားမယ်.. 5 မရောက်ခင်ထိ အလုပ်လုပ်မယ်။ 5 ရောက်ရင် (`range(0, 5)` ဆိုတော့) for loop သည် end သွားမယ်ပေါ့

နောက်ထပ် exercise လေး တစ်ခု လုပ်ပြောည့်ရအောင်။ 0 to 9 ကိန်းဂဏန်းလေး တွေထုတ်ပေးပါ။ အဲ အတွက် code တွေကတွေ့..

serial.py	variable.py	IF_else.py	ForLoop.py	ForLoopLesson.py	testpython.py
1 <code>print("Print 0 to 9 numbers")</code> 2 <code>for i in range(10):</code> 3 <code>print(i)</code> 4					



serial.py	variable.py	IF_else
-----------	-------------	---------

```

1 print("Print 0 to 9 numbers")
2 for i in range(10):
3     print(i)
4

```

C:\Python27\python.exe

```

Print 0 to 9 numbers
0
1
2
3
4
5
6
7
8
9

```

Process returned 0 (0x0) execution time :

အဲလို ထုတ်လိုက်ရင် ထွက်တွော ထွက်တယ် ဒါပေမယ့် ဤအားလုံးများ။ ဘာလို့လဲ ဆိုင် မှုမှ python မှာက default သည့် `print` တစ်ခါတိပြီးတိုင်း တစ်ဖြောင်း ဆင်းသွားတာလို့ လို့ပါ။ အဲတွော `print` ထုတ်ထဲ နေရာ ကုတ်လေး ထပ်ထည့်ပေးရလိမ့်မယ်။ `print` တစ်ခါတိပြီးတိုင်း တစ်ဖြောင်းမဆင်းအောင်။

*** Print without newline in Python 2.x

1 print("Print 0 to 9 numbers") 2 for i in range(10): 3 print(i), 4	<pre style="font-family: monospace; font-size: 0.8em; margin-top: 0;"> Print 0 to 9 numbers 0 1 2 3 4 5 6 7 8 9 </pre> <p style="font-family: monospace; font-size: 0.8em; margin-top: 0; margin-bottom: 0;"> Process returned 0 (0x0) Press any key to continue . . . </p>
---	--

`print(i)` နောက်မှာ , ကိုယ်မှာလေး ခံ ပေးယုံပါပဲ။

*** Print without newline in Python 3.x ***

```
1 print("Print 0 to 9 numbers")
2 for i in range(10):
3     #print(i),
4     print(i, end = " ")
5
```

Print 0 to 9 numbers
0 1 2 3 4 5 6 7 8 9
Process returned 0 <0x0>
Press any key to continue . . .

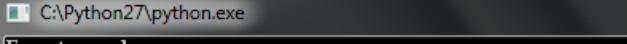
Python 3 မှာရွှေတွာ end = " " ဆိုပြီးတွာ new line မထင်းဘူး.. Space ခြားမှာ ဖြစ်တယ်လို ပြောလိုက်တာပါ။ အကယ်လို့ , နဲ့ ခြားချင်ရင်တွာ end = ',' ပေါ့။

```
1 print("Print 0 to 9 numbers")
2 for i in range(10):
3     #print(i),
4     print(i, end =",")
5
```

Print 0 to 9 numbers
0,1,2,3,4,5,6,7,8,9,
Process returned 0 (0x0)
Press any key to continue . . .

ထပ်ပြီး တော့ ဘာစမ်းကြည့်ပြီးမလဲဆိုတော့ စံကိန်းတွေ သတ်သတ်ထုတ်မယ် မကိန်းတွေ သတ်သတ် ထုတ်ကုသုပ္ပါယ်။ အောက်ကုတ်ကို မရှာသုပ္ပါယ်ပဲ စမ်းရေးရှာသုပ္ပါယ်ပဲ။

```
1 print ("Event number")
2 for i in range(0,30,2):
3     print(i+1, end = ", ")
4
5 print("\nOdd number")
6 for i in range(0,30,2):
7     print(i, end = ", ")
```



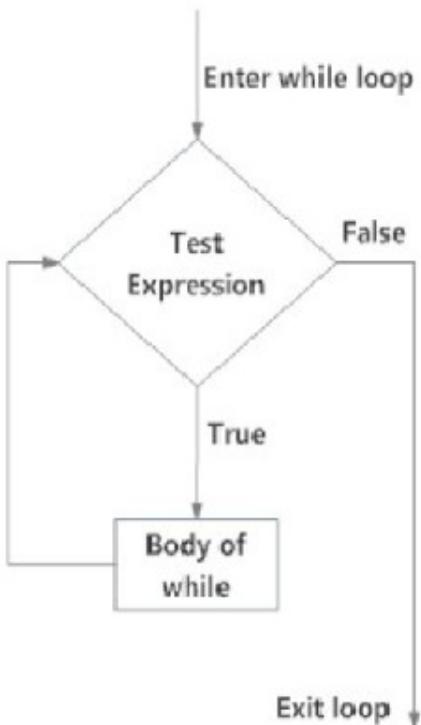
C:\Python27\python.exe

Event number
1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29.
Odd number
1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29.
Process returned 0 <0x0> execution time : 1.633 s
Press any key to continue . . .

5.2 Python While loop

English တမ္မရ ရှိတဲ့ while ကို သုံးသလိပါပဲ။ အောက်က sentence လေးကို တစ်ခုက် ပြုသွားပါ။

He read the book while waiting for the bus



Syntax of while Loop in Python

```
while test_expression:  
    Body of while
```

Simple while loop ලේ: තව්ද රුපාත්මක පෙනෙන විට මෙය නොවූ යුතු වේ.

```
1 count = 0
2 while (count <= 9):
3     print("The count is " + str(count))
4     count += 1
5
6 print ("Bye!!")
7
```

```
The count is 0
The count is 1
The count is 2
The count is 3
The count is 4
The count is 5
The count is 6
The count is 7
The count is 8
The count is 9
Bye!!
```

Line1 => variable count ଯେଣ୍ଟି 0 ଅନ୍ତର୍ଗତ ପିତାଙ୍କ॥

Line2 => While loop ପିଲିତ୍ତିପିଲାଯିଲା। ଫୋର୍ମକ କାଣ୍ଡିଶନ୍ ଆର୍ ଲୂପ ଲାଇଁ କୌଣସି 9 ଥିଲା 9 ହେବାର୍ କିମ୍ବା ଫର୍ଦିଲା ଲୁପିଙ୍ଗ ପିଲାନ୍ତିକାରୀଙ୍କିମ୍ବା ଭାବିଲା। 9 ର୍ଗ୍ରେଟାର୍ ହେବାର୍ କାଣ୍ଡିଶନ୍ ଲାଇଁ false ଅଛିଲାହାତାକୁ ଭାବିଲା while loop ଏହାର୍ ଡ୍ରୋଗ୍ରାମ୍ ପିଲାଯିଲା।



Line3 => Line 3 နဲ့ Line4 သည် while loop ရဲ့ code block ထဲမှာရှိပါတယ်။ သူတို့ နစ်ဂြောင်းသည် same indentation level မှာရှိနေတာဖို့ပါ။ print function နဲ့ the count is ဆိုပြီး လက်ရှိ count တန်ဖိုးကို ထုတ်ပေးပါတယ်။ ခုလက်ရှိ count ထဲက တန်ဖိုးသည် 0 ပါ။

Line4 => count += 1 ဆိုတာ ရွှေတုန်းကလဲ ပြောခဲ့တယ် မှတ်မိုးမယ်လို့ ထင်ပါတယ်။ count = count + 1 နဲ့ အတူတူပါပဲ။ အဲတာဂြောင့် count(0) နဲ့ 1 နဲ့ ပေါင်းပြီး count ထဲက ကို ပြန်ထည့်လိုက်တာပါ။ ဒု လက်ရှိ count ထဲက data value သည် 1 ပါ။

Line5 => count တန်ဖိုးသည် 1 ပဲ ရှိပါ။ while(count <= 9) ဆိုတဲ့ condition မှန်နေသေးတာဖို့ while loop ကနေ မထွက်သေးပါဘူး.. အဲဂြောင့် line3 ကို ပြန်သွားပါတယ်။

Line3 => လက်ရှိ count တန်ဖိုးကို ထုတ်ပါတယ်။

Line4 => count += 1 ဖြစ်တဲ့ အတွက် လက်ရှိ count(1) နဲ့ 1 ကို ပေါင်းပြီး count ထဲကိုပြန်ထည့်လိုက်ပါတယ် ခုလက်ရှိ count တန်ဖိုးသည် 2 ဖြစ်ပါတယ်။

While(count <= 9) ဆိုတဲ့ condition သည် True ဖြစ်နေသမှု looping ပါတ်နေတာကနေ မထွက်ပါတယ်။ while code block ကိုပဲ condition စစ်ရင်း အကြော်ကြော်မှုပါ။

Line3 => လက်ရှိ count တန်ဖိုး(2) ကို print ထုတ်ပါတယ်။

Line4 => count ကို 1 တိုးပါတယ်။

အဲလိုနဲ့ count တန်ဖိုး သည် 9 ကို ရောက်သွားတွော print() နဲ့ The count is 9 ကို ထုတ်ပြီးသွားခိုန်မှာ count ကို 1 ပေါင်းလိုက်ပါတယ်။ count တန်ဖိုး 10 ကို ရောက်တဲ့ အန္တိန်မှာတွော while ရဲ့ conditon သည် မကိုက်တွောတာဂြောင့် Loop ကနေ ထွက်သွားပါတယ်။

Line6 => While loop ကနေထွက်တွောမှ line 6 ကိုရောက်ပြီးတွော Bye!! ကို print ထုတ်ပေးပါတယ်။

5.2.1 Infinite loop

<pre> 1 num = 1 2 while (num == 1): 3 Name = input("Enter your name : ") 4 print ("Hello " + Name) 5 </pre>	<pre> Enter your name : Mg Mg Hello Mg Mg Enter your name : kp Hello kp Enter your name : Aung Aung Hello Aung Aung Enter your name : </pre>
---	--

ဒု num = 1 ဆိုပြီးတွော num ထဲက 1 ထည့်ပိုက်ပါတယ်။ while(num == 1) ဆိုပြီး တွော loop စပါတယ်။ ပြီးတွော Enter your name ဆိုပြီးတွော user ဆိုကနေ input နာမည်တစ်ခုတောင်းတယ်။ အဲဒု မေးမေး hello ဆိုပြီး နာမည် ကို ပြန်ထုတ်ပေးပါတယ်။ while loop ရဲ့ num==1 ဆိုတဲ့ condition မှန်နေသမှု name ငဲ တောင်းမယ် ပြီးတွော Hello ဆိုပြီး ပြန်ထုတ်မယ်။ num ထဲက data ကို ပြောင်းလဲမောမယ့် code ထည့်မရေးထားတဲ့ အတွက် while နောက်က condition သည် True ပဲ ဖြစ်နေမှာပါ။ အဲလိုနဲ့ မပြီးဆုံးနိုင်တဲ့ Loop ဖြစ်နေတာကို infinite loop လို့ ခေါ်ပါတယ်။ loop ကို break ချင်တယ် ရပ်ချင်တယ် ဆိုရင်တွော keyboard မေးမေး ctrl+C ကို နိုင်ပါ။

5.2.2 Else statement with while loop

Python မှာ else ကို while တို့ for တို့နဲ့ တွေ့ပြီး သုံးလို့ ရပါတယ်။ for loop အတွက်ဆိုရင် else ။ ဘယ်ချိန်မှာ အလုပ် လုပ်လဲဆိုတော့ for ရဲ့ sequence ထဲက item တွေကုန်သွားရင် (တစ်နည်း for loop ကနေတွေကုန်သွားရင်) else ဆိုကို ရောက်ပြီး အလုပ် လုပ်ပါတယ်။

While အတွက်ဆိုရင်လဲ သူ့နောက်က condition.. False ဖြစ်လို့ Loop ကနေတွက်သွားချိန်မှာ else သည် စပြီး အလုပ် လုပ်ပါတယ်။ if else တုန်းကလိုပါပဲ။ if ။ false ဖြစ်လို့ တွက်ရင် else ကို ရောက်မယ်လေ။

```

1 count = 0
2 while count < 5:
3     print (str(count) + " is less than 5")
4     count = count + 1
5 else:
6     print (str(count) + " is not less than 5")
7
8 print ("Bye!!")

```

0 is less than 5
1 is less than 5
2 is less than 5
3 is less than 5
4 is less than 5
5 is not less than 5
Bye!!

Process returned 0 (0x0)
Press any key to continue . . .

Count ထဲကို 0 ထည့်တယ်။ whiel loop စပါတယ်။ count < 5 ဖြစ်နေသမှု print (str(count) + " is less than 5") ဆိုပြီး print ထုတ်တယ်။ count ကို while loop တစ်ခါပါတ်ပြီးတိုင်း တစ် ပေါင်းပေါင်းသွား တယ်။ count တန်ဖိုး 5 ဖြစ်သွားတဲ့ အချိန်မှာ while နောက်က condition ဖြစ်တဲ့ count < 5 နဲ့ မကိုက်တော့တာမို့ else ကို ရောက်သွားပြီး 5 is not less than 5 ကို print ထုတ်ပေးလိုက်တယ်။

While နောက်က code block သည် while ရဲ့ condition မှန်နေသမှု အကြိမ်ကြိမ် အလုပ် လုပ်နေပြီးတော့ else ရဲ့ code block သည် while ရဲ့ condition မှာသွားမှ တစ်ကြိမ်သာ အလုပ် လုပ်ပါတယ်။ အော်မြင် ပြုပါ။ bye ဆိုပြီး print ထုတ်တဲ့ code ကတော့ အကုန် လုံပြီး သွားမှ သူဆီရောက်လာမှ အလုပ် လုပ်တာပါ။

5.3 Nested Loop

Nested loop ဆိုကတော့ loop တွေ ဆင့်ဆင့် လုပ်ထားတာကို ပြောတာပါ။

```
for iterating_var in sequence:  
    for iterating_var in sequence:  
        statements(s)  
    statements(s)
```

```
while expression:  
    while expression:  
        statement(s)  
    statement(s)
```

Nested Loop පිරින්මීමෙන් Multiplcation Table ලැබාත්තේ යුතු ක්‍රියාවලිය නොගැනීම්.

```
1 for x in range(1,11):
2     print ("Multiplication of " + str(x))
3     for y in range(1,11):
4         z = x *y
5         print (str(x) + "*" +str(y) + "=" +str(z), end = " ")
6
```

Line2 => x ထဲမှာ 1 ရှိနေတဲ့အတွက် Multiplication of 1 ကို print ထွင်ပါတယ်။

Line3 => for y in range(1,11): ပြုး loop ထပ်ပါတယ်။ y ထဲကို 1 ထည့်ပါတယ်။

Line4 => $z = x * y$ අවශ්‍ය x(1) සහ y(1) වේ ගැනීමෙන් පිටුවේ z(1) නිසා යොදාගැනීමෙන් පිටුවේ.

Line5 => `print(str(x) + "*" + str(y) + "=" + str(z), end = " ")` ଯେଣ୍ଡିଟ୍‌ଆଟ୍‌ଗୁର୍ବି 1*1= 1 ଯେଣ୍ଡିପ୍ରିସ୍ ହେବୁ
result ବାବିତାଯିଲୁ। Next sequent Item ଫ୍ରିଟ୍‌ଟୁ 2 କି y ଦେ ଅନ୍ୟାନ୍ୟ ଲିଙ୍କିପିତାଯିଲୁ। for loop ଆଟ୍‌ଗୁ item କିମ୍ବା
ଫ୍ରେକ୍ସ୍‌ଯାଃତାର୍ଥୀଙ୍କୁ loop ହାର୍କ୍‌ପିର୍ଟିପିତାଯିଲୁ।

Line5 => `print(str(x) + "*" +str(y) + "=" +str(z), end = " ")` ଯେଣ୍ଡାକୁ 1*2= 2 ଯେଣ୍ଡିଲେ: ତୋରୁ
result ବିଧିତାଯି। Next sequent Item ଫେରିଥିଲେ 3 କି y ଯାଇଲେ ଯନ୍ତ୍ରିତ ହେବାରେ କିମ୍ବା for loop ଅଟୁଳିବା କିମ୍ବା
ଫେରିବା କିମ୍ବା loop ପରିପାରିବା କିମ୍ବା

Line1 => 1 ပြီးသွားတော့ x ထဲ ကို 2 ထော်ပါတယ်။

Line2 => x ထဲမှာ 2 ရှိနေတဲ့အတွက် Multiplication of 2 ကို print ထုတ်ပါတယ်။



Line3 => for y in range(1,11): ြီး loop ထပ်ပါတယ်။ y ထဲကို 1 ထည့်ပါတယ်။

Line4 => z = x * y ဖြစ်တဲ့ x(2) နဲ့ y(1) နဲ့ ကို မြောက်ပြီး z(2) ထဲ ကို ထည့်ပါတယ်။

Line5 => print (str(x) + "*" +str(y) + "=" +str(z), end = " ") ဆိုတဲ့အတွက် $2*1= 2$ ဆိုပြီး တော့ result ရပါတယ်။ Next sequent Item ဖြစ်တဲ့ 2 ကို y ထဲ ထည့်လိုက်ပါတယ်။ for loop အတွက် item ။။။ နှုန်းတာမူး loop ဆက်ပါတယ်။

Line4 => z = x * y ဖြစ်တဲ့ x(2) နဲ့ y(2) နဲ့ ကို မြောက်ပြီး z(4) ထဲ ကို ထည့်ပါတယ်။

Line5 => print (str(x) + "*" +str(y) + "=" +str(z), end = " ") ဆိုတဲ့အတွက် $2*2= 4$ ဆိုပြီး တော့ result ရပါတယ်။ Next sequent Item ဖြစ်တဲ့ 3 ကို y ထဲ ထည့်လိုက်ပါတယ်။ for loop အတွက် item ။။။ နှုန်းတာမူး loop ဆက်ပါတယ်။

အဲလိုနည်းနဲ့ .. range(1,11) ဆိုတဲ့အတွက် .. item သည် 1 to 10 ရှိပါတယ် for loop သည် item မကုန် မချင်းပါတယ်မူးပါ။ ပထမ for loop, x= 1 အတွက် ဒုတိယ for loop သည် y = 1 to 10 ထိ looping ကို item ကုန်တဲ့ အထိ ပါတယ်ပါတယ်။ ြီးတော့ ပထမ for loop မှာ ဒုတိယ item = 2 ကို x ထဲ ထည့်ပြီး ဒုတိယ for loop ကိုပါတယ်ပါတယ်။ အဲလို အဲလို နည်းနဲ့ x=3 အတွက် y = 1 to 10, x = 4 အတွက် y = 1 to 10,...etc.. x= 10, y = 1 to 10 ထိ nested loop ပါတယ်။ အလိုပေးလေးကို ထုတ်ပေးပါတယ်။

```
Multiplication of 1
1*1= 1 1*2= 2 1*3= 3 1*4= 4 1*5= 5 1*6= 6 1*7= 7 1*8= 8 1*9= 9 1*10= 10 Multiplication of 2
2*1= 2 2*2= 4 2*3= 6 2*4= 8 2*5= 10 2*6= 12 2*7= 14 2*8= 16 2*9= 18 2*10= 20 Multiplication of 3
3*1= 3 3*2= 6 3*3= 9 3*4= 12 3*5= 15 3*6= 18 3*7= 21 3*8= 24 3*9= 27 3*10= 30 Multiplication of 4
4*1= 4 4*2= 8 4*3= 12 4*4= 16 4*5= 20 4*6= 24 4*7= 28 4*8= 32 4*9= 36 4*10= 40 Multiplication of 5
5*1= 5 5*2= 10 5*3= 15 5*4= 20 5*5= 25 5*6= 30 5*7= 35 5*8= 40 5*9= 45 5*10= 50 Multiplication of 6
6*1= 6 6*2= 12 6*3= 18 6*4= 24 6*5= 30 6*6= 36 6*7= 42 6*8= 48 6*9= 54 6*10= 60 Multiplication of 7
7*1= 7 7*2= 14 7*3= 21 7*4= 28 7*5= 35 7*6= 42 7*7= 49 7*8= 56 7*9= 63 7*10= 70 Multiplication of 8
8*1= 8 8*2= 16 8*3= 24 8*4= 32 8*5= 40 8*6= 48 8*7= 56 8*8= 64 8*9= 72 8*10= 80 Multiplication of 9
9*1= 9 9*2= 18 9*3= 27 9*4= 36 9*5= 45 9*6= 54 9*7= 63 9*8= 72 9*9= 81 9*10= 90 Multiplication of 10
10*1= 10 10*2= 20 10*3= 30 10*4= 40 10*5= 50 10*6= 60 10*7= 70 10*8= 80 10*9= 90 10*10= 100
Process returned 0 <0x0>           execution time : 0.599 s
```

ထုတ်လာတဲ့ result သည် သိပ်မလှတဲ့အတွက် code ကိုနဲ့လောက် ပြင်ကြာည့်ရှာအောင်။

```
for x in range(1,11):
    print ("Multiplication of " + str(x))
    for y in range(1,11):
        z = x *y
        print (str(x) + "*" +str(y) + "=" +str(z), end = " ")
    print("\n")
```

ပြန်ပြီးတော့ run ကြည့်လိုက်ပါ။



```

Multiplication of 1
1*1=1 1*2=2 1*3=3 1*4=4 1*5=5 1*6=6 1*7=7 1*8=8 1*9=9 1*10=10

Multiplication of 2
2*1=2 2*2=4 2*3=6 2*4=8 2*5=10 2*6=12 2*7=14 2*8=16 2*9=18 2*10=20

Multiplication of 3
3*1=3 3*2=6 3*3=9 3*4=12 3*5=15 3*6=18 3*7=21 3*8=24 3*9=27 3*10=30

Multiplication of 4
4*1=4 4*2=8 4*3=12 4*4=16 4*5=20 4*6=24 4*7=28 4*8=32 4*9=36 4*10=40

Multiplication of 5
5*1=5 5*2=10 5*3=15 5*4=20 5*5=25 5*6=30 5*7=35 5*8=40 5*9=45 5*10=50

Multiplication of 6
6*1=6 6*2=12 6*3=18 6*4=24 6*5=30 6*6=36 6*7=42 6*8=48 6*9=54 6*10=60

```

5.4 Loop control statement

Loop တွေ ပုံမှန် အတိုင်း loop နေတာ sequence အလိုက်သွားနေတာ ကို control လုပ်ဖို့ အတွက်နဲ့
statement 3 ခု ရှုပါတယ်။

- Break statement
- continue statement
- pass statement

5.4.1 Break Statement

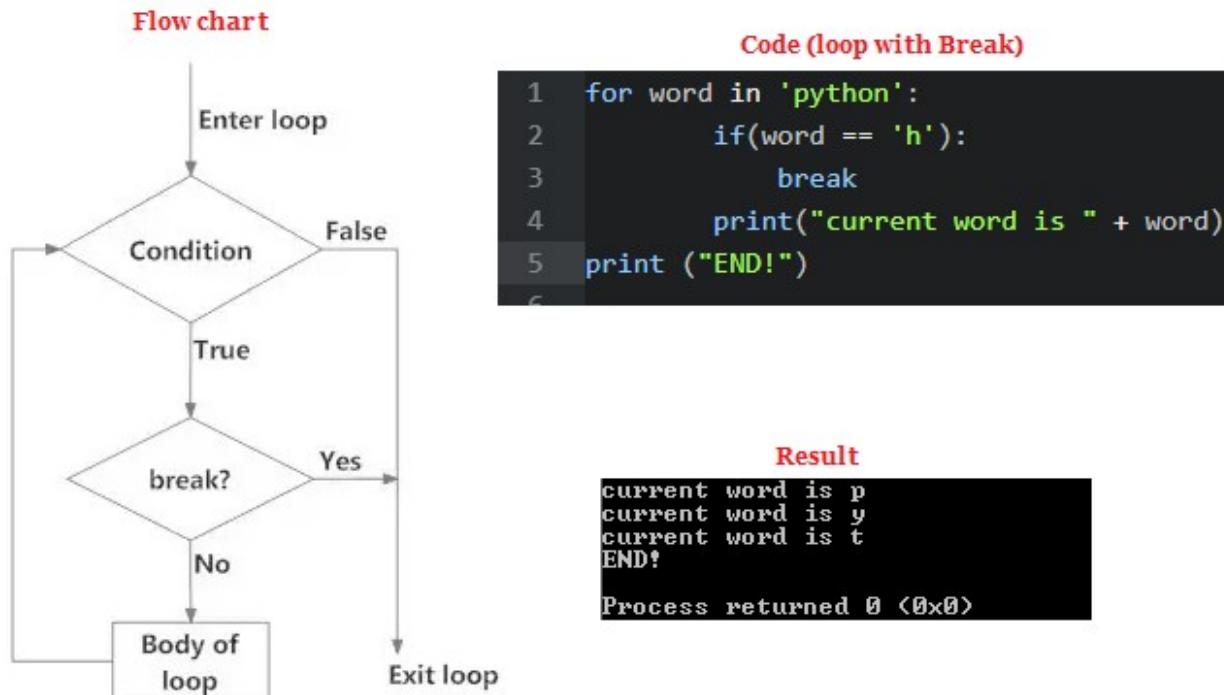
Break သည် loop တစ်ခု၏ ပုံမှန် flow ကို terminate လုပ်ဖို့အတွက် သုံးပါတယ်။ ဥပမာ ပြောရရင်
လူတစ်ယောက် စာရေးနေတယ်.. အဲချိန်မှာ ဖုန်းလာလို့ (Break) ဖုန်းကိုင်လိုက်တယ်ဆိုရင် စာရေးတယ်ဆိုတဲ့
process flow သည် ဖုန်းလာတာကြောင့် interrup ဝင်ပြီး ရပ်သွားပါတယ်။ break သည်လဲ interrup တစ်မျိုး
ပါပဲ။ loop သည် break statement ကြောင့် loop ကနေထွက်သွားပေမယ့် program ကတော့ ဆက်ပြီး run
ပါတယ်။

For loop ပါတ်ထားပါတယ်။ current word is ဆိုပြီးတော့ 'python' ကို တစ်လုံးခုင်းပြု ပေးဖို့ပါ။

<pre> 1 for word in 'python': 2 if(word == 'h'): 3 break 4 print("current word is " + word) 5 print ("END!") </pre>	<pre> current word is p current word is y current word is t current word is h current word is o current word is n END! Process returned 0 (0x0) </pre>
---	--



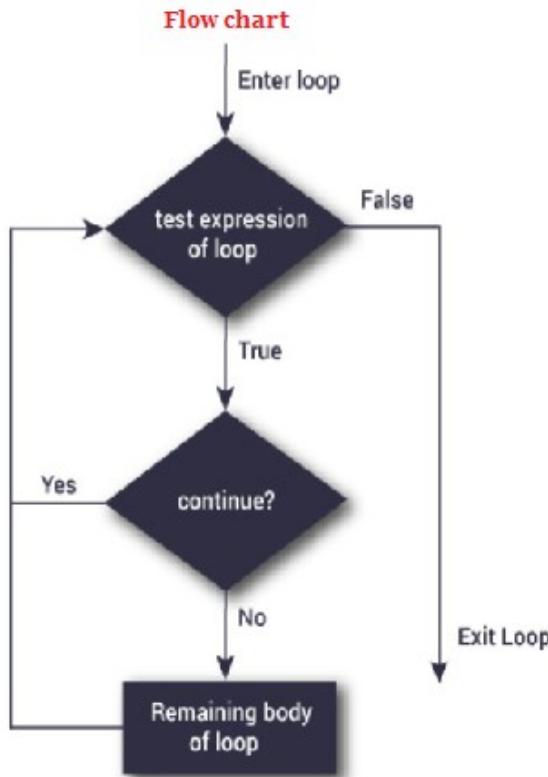
Break statement ကို If ကိုသုံးပြီးတော့ h ကို ရောက်ရင် break လုပ် နိုင်:ထားတာပါ။အောက်က ကုတ် နဲ့ chart ကို ဖြေဆုံးပါ။ p, y, t အထိပ် ထွက်လာပြီးတော့ h ကိုရောက်တဲ့ အနိမ့်ဂျာ break သွားတာကို ကျောမှာပါ။



နောက်ထပ် example တစ်ချေလောက် ထပ်ဖြေဆုံးရာရအောင်။ Quit လို့ မရှိက်မချင်း inpute ဝင်လာ မှုံstring တွေရဲ့ len ကို ဖော်ပြုပေးမယ့် .. program လေး တစ်ခု ရေးပြုဆုံးပါ။

<pre> while True: s = input('Enter something : ') if s == 'quit': break print('Length of the string is ' + str(len(s))) print('Done') </pre>	<pre> Enter something : Hello, nice to see you Brother Length of the string is 30 Enter something : how do you do? Length of the string is 14 Enter something : hi Length of the string is 2 Enter something : quit Done Process returned 0 <0x0> execution time : 22 Press any key to continue . . </pre>
--	---

5.4.2 Continue Statement



Continue statement ໂດຍ່ານວິທີ Break ລື້ອີ້ນ ກວດ ຖຸກວູ້ວ່າມີວິທີໃຫຍ່ຈະເລີກຕົວຢ່າງດີ. Continue ກົດເຮັດວຽກກຳນົດ skip ລົບວູ້ວ່າຕ້າມື້ອີ້ນ. ອາຍຸລື້ອີ້ນລະສິບຕູ້ວາ ເພີ້ມ example ມາ ດູວ່າລົບຮັດ ບໍ່ມີກຳນົດ Loop ປີ່ເກີດຕູ້ວາ ທີ່ ກົດ `if (word == 'h'): continue` ລື້ອີ້ນ code ເຊິ່ງນີ້ word = h ກົດ ເຮັດວຽກຕູ້ວາ ເຊີ່ນມີມາ print ຕູ້ຕົວຕູ້ວາ ເພີ້ມເຮັດວຽກຕູ້ວາປໍ ເພີ້ມ loop ອາຍຸການ ກົດ ປຸກເຮັດວຽກວູ້ວ່າປີເຕີຍ. result ກົດ ດູວ່າລົບຮັດ ລື້ອີ້ນ h ອາດຖານກ ມາຫຼຸດຫຍາວ່າຕ້າມກົດ ຕູ້ມີກຳ.

5.4.3 Python Pass Statement



```
>>>
>>> for i in "Python":
...
...
File "<stdin>", line 3
    ^
IndentationError: expected an indented block
>>>
>>> for i in "Python":
...     pass
...
>>>
```

ဘယ်လိုနေရာတွေမှ သုံးလဲဆိုတွာ ကိုယ်က Program တစ်ခုရေးနေတယ်ပေါ့။ အဲခြိန်မှာ မှ functionကန်ဖို့ function တွေ loop တွေကို reserve လုပ်ထားချင်တယ်။ future purpose အတွက် ထည့်ရေးမယ်ပေါ့ ဒီတိုင်း function/loop ပဲ ထားခဲ့ရင်လဲ error တက်ပါတယ်။ အဲတွက်ကြောင့် အောက်က လုပ်မယ့် action ကို pass အနေနဲ့ ထားခဲ့ခြင်းဖြင့် program ကို error လဲ မတက်သလို ဘာ effect မှ မဖြစ်စေတွောပါဘူး။

6. Python - Date & Time

Network အတွက်ဆိုရင် backup လုပ်တဲ့နေရာ.. Security protocol တွေ enable တဲ့နေရာ MS လုပ်မယ့်နေရာ တွေမှာ ဆို င် date&time သည် အရေးကြီးပါတယ်။ အဲကြောင့် လျှော့ကြော် ရာရအောင် မှာ။ Date and time အတွက် ဆိုရင် Python မှာ ရှိတဲ့ datetime ဆိုတဲ့ module ကို သုံးပါမယ်။ ပထမဆုံး module ကို import လုပ်ပါ။ အဲလို import လုပ်တာက ဘာနဲ့ တူလဲဆိုတွာ c++, c တို့မှာ #include <iostream> ဆိုပြီးတွေ့ header ကို ကြော့ပြာ ပေးရသလိုမှိုးပါပဲ။ အဲရှာမှ သူနဲ့ သက်ဆိုင်တဲ့ cin/cout စတာင် တွေကို ခေါ်သုံးလို ရသွားတာပါ။

Access current Datetime :

- Module : **datetime**
- Class : **datetime**
- Method : **today()**

```
>>> import datetime
>>>
>>> print(datetime.datetime.today())
2019-04-04 15:00:14.365000
>>>
```

ပထမဆုံး datetime Module ကို import လုပ်ပါ။ import လုပ်တယ်ဆိုတာက အဲmodule ကြီး တစ်ခုလုံးကို ခုက္ခာယ်ရေးနေတဲ့ ဒီမဟုတ် run နေတဲ့ program/script ထဲ ကို ထည့်လိုက်တာမဟုတ်ပါဘူး။ Python ရဲ ငါအဲဒီ moduel တွေကိုတွေ့ သုံးမယ်လို ကြိုပြောထားတဲ့ ပုံစံပါ။ အောက်က screenshot ကို တစ်ချက် ဖြော်ပြု ဖြော်လိုက်ပါ။



```
>>> print(datetime.datetime.today())
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'datetime' is not defined
>>>
>>> import datetime ✓
>>> print(datetime.datetime.today()) ✓
2019-04-07 18:55:01.229529
>>>
```

အဲလို import လုပ်ထားမှ module ထဲက ကောင်တွေကို သုံးလို ရမှာပါ။ ခု ဂြွန်တို့တွေ

လုပ်ချင်တာသည် current date နဲ့ time ကို ထုတ်ချင်တာပါ။ `print(datetime.datetime.today())` ဆိတဲ့ code သည် datetime module ထဲက date time class ရဲ့ today() ဆိတဲ့ Method ကိုခေါ်သုံးပြီး print ထုတ်လိုက်ပါတယ်။ ခုကာလက်ရှိ date (year, month, day) နဲ့ time (hr, min, sec) ကို ရရှိတဲ့ အခါ ဖြေည့်လိုက်ရင် time မှာ က micro sec အထိပြု နေတာကို တွေ့ရမှာပါ။ အောက်ဆုံးက အသမ တွေနဲ့ ဖော်ပြုပေးထားပါတယ်။

`print(datetime.datetime.today())` ကို အကြိမ်ကြိမ် ဆက်တိုက်ထုတ်ဖြေည့်ရင် micro sec လေးတွေ ပြောင်းလဲ နေတာကို တွေ့ရမှာပါ။

```
>>> print(datetime.datetime.today())
2019-04-04 15:22:55.031000
>>> print(datetime.datetime.today())
2019-04-04 15:22:55.337000
>>> print(datetime.datetime.today())
2019-04-04 15:22:55.661000
>>> print(datetime.datetime.today())
2019-04-04 15:22:56.035000
>>> print(datetime.datetime.today())
2019-04-04 15:22:56.388000
>>> print(datetime.datetime.today())
2019-04-04 15:22:56.697000
>>> print(datetime.datetime.today())
2019-04-04 15:22:56.974000
>>> print(datetime.datetime.today())
2019-04-04 15:22:57.251000
```

`print(datetime.datetime.today())` ဆိတဲ့ code အပြင် `print(datetime.datetime.now())` ကိုလဲ သုံးလို ရပါတယ်။

```
>>> print(datetime.datetime.today())
2019-04-04 16:40:14.274000
>>> print(datetime.datetime.now())
2019-04-04 16:40:17.937000
>>>
```

6.1.1 Python Datetime, Date, Time

ရလာတဲ့ result မှာ date ကော time ကော အကုန် ပါနေတာကို နေတာကို တွေ့ရပါတယ်။ date ကို ပဲ ဗြာည့်ချင်တယ်ဆိုရင်တွော..

```
>>> print(datetime.date.today())
2019-04-04
>>>
```

print(datetime.date.today()) ကို သုံးပြီး ထုတ်လိုပါတယ်။ အဲလို print ထုတ်လိုလဲ ရသလို object တစ်ခု create လုပ်ပြီးတွောလဲ print ထုတ်လိုပါတယ်။ ခုခ္ဓိနှာတွေ module တွေ class တွေ function တွေ object ဆိုတဲ့ စကားလုံးတွေကို စိမ့်နေပြီးမှာပါ။ Function ခန်း class ခန်း ရောက်ရင်တွော ရှင်းသွားပါလိမ့်မယ်။ စိတ်မပူပါနဲ့။

```
>>> Mytime = datetime.date.today()
>>> print ("Current time is " + str(Mytime ))
Current time is 2019-04-04
>>>
```

current date တွေ ထည့်ထားမယ် Mytime ဆိုတဲ့ object ကို date class ထဲက today() method ကို သုံးပြီး ဖန်တီးလိုက်ပါတယ်။ print ("Current time is " + str(Mytime)) ဆိုပြီးတွော mytime object ထဲမှာ ရှိတဲ့တဲ့ date ကို ထုတ်ပြပါတယ်။

6.1.2 Date object to represent a date

အခုလို့ လက်ရှိ ရှိနေတဲ့ date, time တင်မဟုတ်ပဲ ကိုယ့်စိတ်ကြိုက် data ထည့်ပြီးလဲ format ရာ ငါ အင် ပြန်ထုတ်လိုပါတယ်။

```
>>> import datetime
>>> crossnet = datetime.date(2019, 4, 4)
>>> print(crossnet)
2019-04-04
>>>
```

crossnet = datetime.date(2019, 4, 4) ဆိုပြီးတွော Argument သုံးခဲတွော ထည့်ပေးရပါမယ် year, month, day အတွက်ပါ။ ထည့်လိုက်တဲ့ date ကို ပြန်ထုတ်လိုက်တဲ့ အခါဂျာ 2019-04-04 ဆိုပြီး format တစ်ရာ ပြန်ရပါတယ်။ crossnet ဆိုတဲ့ variable သည် date object ပဲ ဖြစ်ပါတယ်။ ဒါ ရာန်တို့ သုံးနေတဲ့ import အပြင် နောက်ထပ် သုံးလို့ရတဲ့ import ပုံစံလေးတွေ ရှိပါသေးတယ်။ တစ်ချက်လောက် လေးလာကြားလုပ်ရအောင်များ။

import vs from import

import [module_name] (example .. import sys) ကိုတော့ အပေါ်က ကုတ်တွေမှာ သုံးခဲ့ပါတယ်။ ဒါ import လုပ်တဲ့ နည်းသည် module ကို import လုပ်တာပါ (as per theory => sys module ရဲ့ OBJECT ကို ခဲ့လိုက်ရှိ program မှာ သုံးဖို့ sys နာမည် နဲ့ bind (ပေါင်း) လိုက်တာပါ။) ဒီနည်းလမ်းသည် sys ထဲမှာ ရှိတယ် ဘယ်ကောင်ကိုမဲ့ direct access လုပ်ခွင့်မရှိပါဘူး။ အဲအတွက်ကြောင့် sys ထဲ မှာ ရှိတဲ့ ကောင်ကို လုမ်းပြီးတော့ ခေါ်သုံးမယ် ဆိုရင် ရွှေ့ကနေ sys. (ကိုယ်သုံးမထုံးကောင် ရဲ့ module) ထည့်ပေးရပါတယ်။ အဲတာကြောင့် version ကို ခေါ်မယ်ဆိုရင် sys ထဲမှာ ရှိတဲ့ ကောင်ကို ခေါ်တာဖြစ်လို့.. Sys.version လို့ ချိလိုက်မှ ချက်ရှိရှိ Python version ။ GCC version တို့ကို မြင်ရတာပါ။

```
ekhaphy@MM002005:88 ~
$ python
Python 2.7.14 (default, Oct 31 2017, 21:15:21)
[GCC 6.4.0] on cygwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>>
>>> version
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'version' is not defined
>>>
>>> sys.version
'2.7.14 (default, Oct 31 2017, 21:15:21) \n[GCC 6.4.0]'
>>>
```

အောက်က ပုံလေးနဲ့ရှင်းပြထားတာကို ပြောည့်လိုက်ရင် ပိုရှင်းသွားပါလိမ့်မယ်။



ခုက္ခန်းတို့တွေမှာ ပါသလဲ သုံးချိုတယ် ဆိုရာပါစိုး.. အဲထဲမှာ လက်တောာရယ် နာရူတိရယ် Iron Man ရယ် ရှိတယ်။ အထုတ်ထဲက ပစ္စည်းကို ယူမယ်ဆိုရင် အထုပ်ကို အရင်ဖွင့်ပြီးမှ ယူလို့ရတာပါ။ လက်ငတဲ့ကို ယူမယ်ဆိုရင် အဆင့်နစ်ဆင့်ရှိပါတယ်။

1. လိမ့်းရောင် box ကို ယူပါ
2. လိမ့်းရောင် အထုပ်ကို ဖွင့်ပြီး အထဲက laptop ကို ယူရသလိုမိုးပဲ့..

Python import အတွက် လက်တွော ယူသလို အဆင့်နစ်ဆင့် နဲ့ ဥပမာ ပြန်ပြပါမယ်။

1. `import sys (sys ဆိုတဲ့ module ကို import လုပ်ပါ)`
2. `Sys.version (version ကိုယူသုံးမယ်ဆိုရင် (ခု အပေါ်က example မှာဆို laptop ပေါ့..) ကိုယ် ဘယ်က ကောင်ကို ယူမယ်ဆိုတာ တစ်ဆင့်ခံပြီး ရေးပေးရမယ်။ sys ဆိုတဲ့ module ထဲက version ကို ယူမှာဖြစ်လို့ sys.version လို့ ရေးရတာပါ။ version ခဲ့ result ကို ရလာပါလိမ့်မယ်`

အဲလိုပဲ နာရူတိကို ရဖို့ယူဖို့ အတွက် ဆိုရင် ခရမ်းရောင် အထုပ်ကို အရင်ယူပြီး ဖွင့်ရပါမယ် ပြီးမှ အရှင်ကို ယူရတာပါ။ ထိနည်းတူစွာပဲ.. time module ထဲက `gmtime()` ကိုယူသုံးချင်တယ်ဆိုရင် (`import time` အတွက်ဆိုရင်) `time.gmtime()` လို့ ရေးလိုက်မှ `year 2019 ဆိုတို့ month 4 တို့ စသည်ဖြင့် ထွက်လာတာပါ။ ဒီတိုင်း gmtime() ပဲ ရေးလိုက်ရင်တွော gmtime ကို မသိတဲ့ အကြောင်း error တက်ပါလိမ့်မယ်။ ခရမ်းရောင် အထုပ်ကို မဖွင့်ပဲ နာရူတိကို ယူလို့ မရသလိုပေါ့မာ။`



```
>>> import time
>>> time.gmtime()
time.struct_time(tm_year=2019, tm_mon=4, tm_mday=4, tm_hour=18, tm_min=3, tm_sec=10, tm_wday=3, tm_yday=94, tm_isdst=0)
>>>
>>> gmtime()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'gmtime' is not defined
>>>
```

ဟုတ်ပြီး ခဲ့`import` ကို သုံးတာရဲ့ ကောင်းတာ မကောင်းမယ် ထင်တာလေးတွေကို နှင်းယူဉ်ဖြောက်၍ ရှာရအောင်။

**** import [module] ****

`Import` ဆိုပြီး နောက်က `module` ပဲ ရေးထားတဲ့ အတွက် အဲ `module` အောက်က ကောင်တွေကို ငါသုံးလို့ ရတယ်။

ဒါပေမယ့် တစ်ခုခုဆို `module.[something]` ဆိုပြီး ရေးနေရတာက စာရေးရတာ `code` ရှည်တယ်ပေါ့ များ .. အဲလိမ့်းအတွက် ဖြေရှင်းလို့ရမယ့် နည်းတစ်ခုတွော ရှိတယ်များ .. ဥပမား `as` ကို သုံးရမှာပါ။

```
>>> import datetime as dt
>>> print(dt.datetime.today())
2019-04-05 01:22:09.483023
>>>
```

`import datetime as dt` ဆိုတဲ့ `code` မှာ `as` သည် ဘာကိုလုပ်ပေးတာလဲ ဆိုတွော `datetime` ဆို
မြတ်module ကို ခေါ်မယ် ညွှန်းဆိုမယ် ဆိုရင် `dt` အအနေနဲ့ သက်မှတ်တယ်လို့ .. ပြောလိုက်တာပါ။ ပုံမှန်ဆိုရ`datetime.datetime.today()` လို့ ခေါ်ပြီး current date & time ကို ကြည့်ရမှာ ဖြစ်ပောက်၍ `as` ပြောင့်မို့
`Datetime` module နေရာမှာ `dt` လို့ သုံးလို့ ရသွားပါတယ်။ ကိုယ့် သက်တနဲ့ ကိုယ် တိုတိရေးလို့ ရတာပေါ့။

**** from module import [] ****

ကိုယ့် သုံးချင်တဲ့ `item` ကို ပဲ ယူတာဖြစ်လို့ သုံးမယ့် `item` တွေကို `control` လုပ်နိုင်တယ်။
သူက `code` ခေါ်သုံးတဲ့ အနီးနဲ့ ရေးရတာသက်သာတယ်။ အရွှေက `box` အထုပ် ဥပမာ နဲ့ ပြန်ပြောရရ
`from module import []` သည် `box` ကို ဖွင့်ပေးထာနဲ့ တူပါတယ်။ ဒီတိုင်း ယူသုံးလိုက်ယုံပါပဲ။

```
>>>
>>> from sys import version
>>> version
'2.7.14 (default, Oct 31 2017, 21:15:21) \n[GCC 6.4.0]'
```

from sys import version ဆိတဲအတွက် sys ဆိတဲ module ထဲက version ကို import လုပ်ထားတာပါ။ အဲအတွက်ငြောင့် version လို့ ခေါ်လိုက်တာနဲ့ error မတတ်ပဲ .. ခုလက်ရှိ version ကို ပြေားပါတယ်။

ဒါပေမယ့် item အသစ်တစ်ခု သုံးချင်တိုင်း import လုပ်ပေးနေရတာကတော့ အလုပ် ရှုတ်ပါတယ်။ import ထပ်လုပ်မပေးရင် define လုပ်ပေးထားပါဘူး ဆိုပြီးတော့ error တက်လာပါလိမ့်မယ်။

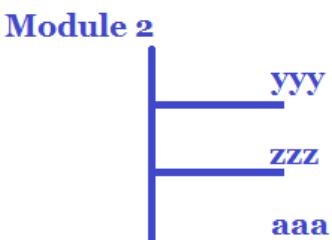
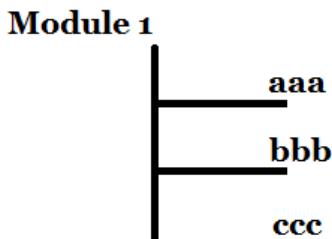
```
>>> platform
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'platform' is not defined
>>>
>>> from sys import platform
>>> platform
'cygwin'
>>>
```

From sys import platform ဆိုပြီး sys moduel ထဲက platform ကို import လုပ်လိုက်မှ သုံးလို့ ရသွားတာပါ။ ဒါ ကျွန်ုတ်ဘာ python သုံးနေတဲ့ platform သည် cygwin ဖြစ်တာမို့ cygwin ဆိုပြီး result ထွက်လာတာပါ။

import [module] တုန်းရာက module name ကို ရှုံးမှာ .(dot) ထည့်ပြီး ရေးသုံးရပါ တယ်။ from [module] import [..] ရာတွေ့ ဘယ် module က ကောင်ကို import လုပ်ပါတယ် လို့ ငြော်ကြေားတာဖြစ်လို့.. ရှုံးမှာ module နာမည်ထည့်စရာ မလိုတွောပဲ direct access ရတာပါ။

*** >> **Using from module import *** << is not Best practice ***

Import * ကို ဖြစ်နိုင်ရင် မသုံးစေချင်ပါဘူး. Module ကို ယူတာမဟုတ်ပဲ.. Module ထဲက ရှိသမ္မ item တွေအကုန်ကို ခုလက်ရှိ run၆၆ / ရေးနေ တဲ့ ထဲ ကို ဆွဲသွင်းလိုက်တာပါ။ ရတိတွေ့ တွေးကြော်ရင် အ ဆင်ပြေ နေသလိုပဲ နော့။ ဒါပေမယ့် သူက နာမည်တူတာတွေရှိရင် overwrite တဲ့ ပြဿနာရှိပါတယ်။



```

from module1 import *
from module2 import *

# ခုသည်ဥပမာများရင် module1 & module2
ထဲက ကောင်တွေကိုအကုန် ကို import လုပ်ထားတာပါ
>>> bbb      # call bbb from module1
>>> zzz      # call zzz from module2
>>> aaa      # # how do you think?
  
```

aaa လို့၏လိုက်တဲ့အချင့်မှာ module2 သည်
 နောက်ဆုံး importလုပ်တာမို့.. module2 က aaa ကို
 လုမ်းချေပါလိမ့်မယ်။ name conflict ဖြောင့် module1
 က aaa ကို module2က aaa က overwrite
 လုပ်သွားပါတယ်။

Module ထဲက ကောင်တင် နာမည်တူနိုင်ပုံတင် မကသေးပါဘူး.. ကိုယ့် variable နဲ့ နာမည်တူခဲ့ရင် ပြဿနာ
 ရှိနိုင်ပါတယ်။ အောက်က ဥပမာ ကို ပြောလိုပါ။

```

Type "help", "copyright", "credits" or "license" for more information.
>>> platform = 'parrotsec'
>>> from sys import *
>>>
>>> platform
'cygwin'
>>>
>>> platform = 'parrotsec'
>>> platform
'parrotsec'
>>>
  
```

```

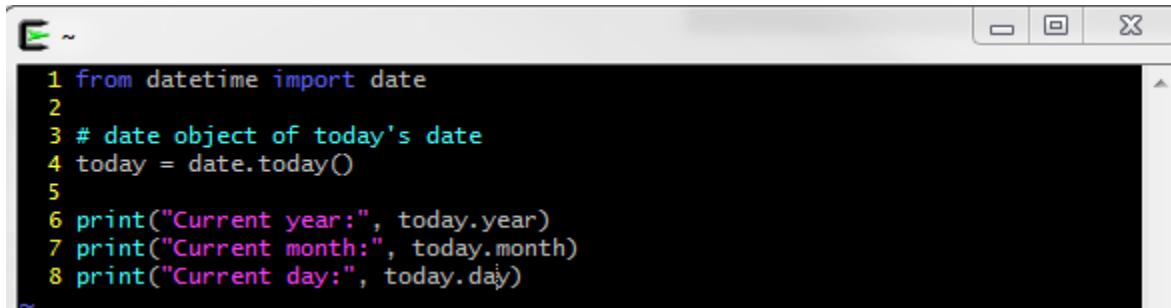
platform = 'parrotsec'
from sys import *
  
```

အပေါ်က နှစ်ပြောင်းမှာဆုံး platform ထဲကို 'parrotsec' ဆိုတဲ့ string ထည့်ထားပါတယ်။ နောက်ပြီး sys
 module မှာရှိတဲ့ ကောင်တွေ အားလုံးကိုလဲ import လုပ်ထားပါတယ်။ sys ထဲ က platform လဲ အပါ အပင်
 ပေါ့။ အဲတော့ဆုံး >>> မှာ platform လို့၏လိုက်ရင် ဘယ်ကောင်ရဲ့ result ထွက်လာမလဲ ဆိုတော့ 'cygwin'
 ဆိုပြီးတော့ ရပါတယ်။ sys module ထဲ က ကောင်ကို ချေသုံးလိုက်တာပဲ ဖြစ်ပါတယ်။ from sys import *
 သည် platform = 'parrotsec' ကို overwrite လုပ်လိုက်တဲ့ သဘောမျိုး ဖြစ်သွားလိုပါ။

နောက်တစ်ခါ platform = 'parrotsec' လို့ ရေပြီး ပြန်ထုတ်ကြေည့်ရင်တော့ platform = 'parrotsec' က ပြန်ပြီး overwrite ဖြစ်သွားတာမို့ parrotsec ဆိုတဲ့ အဖြေရပါတယ်။

IMPORT အကြောင်းတော်ကောလေး သိသွားပြီဆိုတော့ datetime ဘက်ပြန်လှည့်ရာရအောင်။

ဒုလက်ရှိ ရှိနော်တဲ့ year, month, day တွေကို တစ်ခုချင်း ထုတ်ကြေည့်ရာရအောင်။



```

E ~
1 from datetime import date
2
3 # date object of today's date
4 today = date.today()
5
6 print("Current year:", today.year)
7 print("Current month:", today.month)
8 print("Current day:", today.day)
~
```

Line1 => datetime ဆိုတဲ့ module ထဲက date class ကို import လုပ်လိုက်ပါတယ်။

Line3 => # သက်တန်း comment ပေးတာပါ။ PYTHON သည် CODE အနေနဲ့ထည့်ပြီးမ run ပါဘူး။

Line4 => date object အတွက် today နာမည် နဲ့ variable တစ်ခုကို သုံးပြီး တည်ဆောက်လိုက်ပါတယ်။ today ထဲမှာ ခုလက်ရှိ year, month, date တို့ ပါဝင်ပါတယ်။

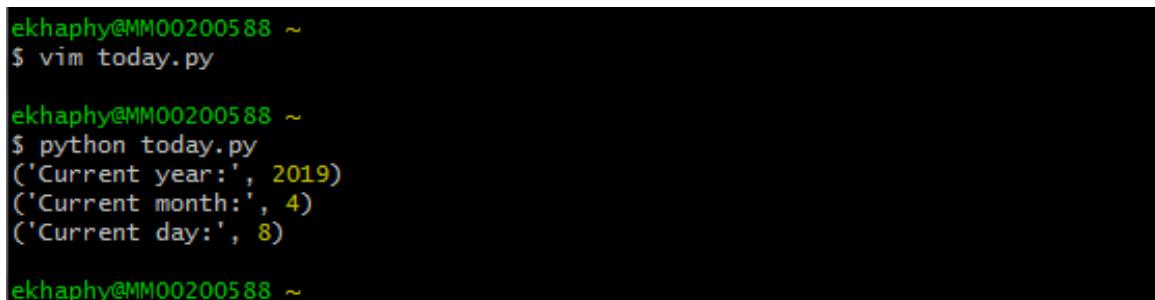
Line6 => today.year ဆိုပြီတော့ ခုလက်ရှိ ခုနှစ်ကို ထုတ်ပါတယ်။

Line7 => today.month ဆိုပြီတော့ ခုလက်ရှိ လ ကို ထုတ်ပါတယ်။

Line8 => today.day ဆိုပြီတော့ ခုလက်ရှိ နေ့ ရက် ကို ထုတ်ပါတယ်။

Today နောက်က ဘာမှမပါဘူးဆိုရင်တော့ [print(today)] ခုလက်ရှိ နှစ် လ ရက် အကုန်ကို ထုတ်ပြုပါလိမ့်မယ်။

Text editor တစ်ခုနဲ့ save ပြီး python နဲ့ run လိုက်ရင် အောက်ပါအတိုင်း result ထွက်ပါလိမ့်မယ်။



```

ekhaphy@MM00200588 ~
$ vim today.py

ekhaphy@MM00200588 ~
$ python today.py
('Current year:', 2019)
('Current month:', 4)
('Current day:', 8)

ekhaphy@MM00200588 ~
```

အကယ်လို့.. ခုလက်ရှိ အနီးနဲ့ပဲ ထုတ်ချင်တယ်ဆိုရင်တော့ `print(datetime.now().time())` ပါ

လက်ရှိ ရောက်နေတဲ့ ခုနစ် လ ရင် ကိုလဲ `print(datetime.now().date())` ဆိုပြီး ထုတေလို့ ရပါတယ်။ အော မေးမ screenshot ကို ဖြေဆုံးပါ။

```
>>> from datetime import datetime
>>>
>>> print(datetime.now().time())
01:59:32.116603
>>>
>>> print(datetime.now().date())
2019-04-08
>>>
>>> print(datetime.now())
2019-04-08 01:59:50.322403
>>>
```

6.2 Format date using strftime()

`strftime()` Method သည် `date` ကြောင်း `time` တွေကို string အနေနဲ့ ပြန်ထုတ်ပေးတဲ့ ကောင်ပါ။ အောက်က ကုတ်တွေကို ကိုယ့်စိတ်ကြိုက်နာမည် တစ်ခုနဲ့ `save` ပါ။ ရွှေ့နောကတွေ့ `kp.txt` ဆိုပြီးတော့ သိမ်းချွဲပါတယ်။

```
1 from datetime import datetime
2
3 now = datetime.now() # current date and time
4
5 year = now.strftime("%Y")
6 print("year:", year)
7
8 month = now.strftime("%m")
9 print("month:", month)
10
11 day = now.strftime("%d")
12 print("day:", day)
13
14 time = now.strftime("%H:%M:%S")
15 print("time:", time)
16
17 date_time = now.strftime("%m/%d/%Y, %H:%M:%S")
18 print("date and time:",date_time)
19 |
20 dt_with_month = now.strftime("%d/%b/%Y, %H:%M:%S")
21 print("date and time with Month:",dt_with_month)
22
```

Line1 => ၁ datetime module ထဲက datetime class ကို import လုပ်လိုက်တာပါ။



Importing datetime class from datetime module.

```
from datetime import datetime
```

Line3 => datetime အတွက် object တစ်ကို တည်ဆောက်လိုက်ပါတယ်။ now ဆိုတဲ့ variable သည် ခုလက်ရှုံး date နဲ့ time ပါဝင်နေတဲ့ datetime object တစ်ခုပါ။

Datetime object containing current date and time.

```
now = datetime.now()
```

Line5 => strftime() method ကိုသုံးပြီးတော့ သူမှာရှိတဲ့ format code တွေနဲ့.. ကိုယ်ပေါ်စေချင်တဲ့ format နဲ့.. Date, time ကို အောင်ပြလို့ရပါတယ်။

Contains formatted string.

```
year = now.strftime("%Y")
```

Format code. %Y formats to year.

strftime() ရဲ့ format code တွေသည် အမှားငြား စုပေါင်း စပ်ပေါင်းပြီးတော့ ကိုယ့်စိတ်ဖြိုက် ပြစေချင်တဲ့ အစီအစဉ် လိုက် အတိုင်းရေးပေးနိုင်ပါတယ်။

```
date_time = now.strftime("%m/%d/%Y, %H:%M:%S")
```

04/08/2019 02:48:04

Kp.txt ကို python အနေနဲ့ run လိုက်ရင် အောက်ပါအတိုင်း result ရပါလိမ့်မယ်။

```
C:\Users\ekhaphy>python3 kp.txt
year: 2019
month: 04
day: 08
time: 02:48:04
date and time: 04/08/2019, 02:48:04
date and time with Month: 08/Apr/2019, 02:48:04
```

ခုသုံးဖြစ်ခဲ့တဲ့ format code လေးတွေပါ။

- %Y - year [0001,..., 2018, 2019,..., 9999]
- %m - month [01, 02, ..., 11, 12]
- %d - day [01, 02, ..., 30, 31]
- %H - hour [00, 01, ..., 22, 23]
- %M - month [00, 01, ..., 58, 59]
- %S - second [00, 01, ..., 58, 59]
- %p - Locale's AM or PM.[AM, PM]
- %b - Abbreviated month name [Jan, Feb, ..., Dec]

6.3 strftime()

strftime() တုန်းက datetime ကို string အနေနဲ့ ပြောင်းခဲ့တာပါ။ ဒါ strftime() ၏ မြော်မြော် string ကို datetime အနေနဲ့ ပြန်ပြောင်းပေးတာပဲ ဖြစ်ပါတယ်။ **Line no.1** မြော်မြော် datetime module ထဲက datetime ကို import လုပ်တာပါ။

```

1 from datetime import datetime
2
3 date_string = "21 June, 2018"
4
5 print("date_string =", date_string)
6 print("type of date_string =", type(date_string))
7
8 date_object = datetime.strptime(date_string, "%d %B, %Y")
9
10 print("date_object =", date_object)
11 print("type of date_object =", type(date_object))
```

Line no.3 မှာ datetime ပြောင်းမယ့် string တစ်ခု ကို date_string ဆိုတဲ့ variable ထဲ ကို ထည့်ထားပါတယ်။ **line5** မှာ date_string ထဲမှာ ရှိတဲ့ string ကို Print ထုတ်ပြထားပါတယ်။ **Line6** က ရာဇ္ဈား မှာ မြော်မြော် date_string က ဘာ data type ဖြစ်တယ်ဆိုတာကို သိအောင် print ထုတ်ပြထားပါတယ်။

Line no.8 မှာ strftime() ကိုသုံးပြီးတော့ date_string ကို format code တွေသုံးပြီး datetime အဖြစ် ပြန်ပြောင်းလိုက်ပါတယ်။ date_object variable ထဲ ကို ထည့်ပါတယ်။ **strftime()** မှာ ရှုကဗော် ငါသည် data string ဖြစ်ပြီးတော့.. နောက်ကကောင်သည် format code ဖြစ်ပါတယ်။ **line 10** ၏ date_object ထဲမှာ ရှိနေတဲ့ အတောက် ထုတ်ပြတာဖြစ်ပြီးတော့ **line 11** သည် date_object သည် ဘာ typeဖြစ်တယ်ဆိုတာကို ထုတ်ပြတာပါ။

```
'date_string =' , '21 June, 2018')
('type of date_string =' , <type 'str'>)
('date_object =' , datetime.datetime(2018, 6, 21, 0, 0))
('type of date_object =' , <type 'datetime.datetime'>)

Process returned 0 (0x0)      execution time : 0.410 s
Press any key to continue . . .
```

6.4 Current time of time zone

မတူသီတဲ့ နေရာဒေသက နိုင်ငံတွေရဲ့ real time (လက်ရှိ အနိုင်) ကို သိဖို့ဆိုရင် တွေ့၍ pytz module ကို သုံးပြုတွေ့၍ ထုတ်လို့ ရပါတယ်။ ဒါပေမယ့် ဒီနေရာမှ တစ်ခုရှိတာက pytz module သည် built-in ပါမလာပါဘူး။ အဲတွေ့ သူ့ကို ထပ်သွင်းမှ ရမှာပါ။ အလူယ်ဆုံးနှင့်းကတွေ့၍ pip ကို အသုံးပြု၍ ပြီးတွေ့၍ ထည့်လိုက်တာပါပဲ။ Python3.4 နောက်ပိုင်း version တွေမှာတွေ့၍ pip သည် install လုပ်ပြီးသားပါ။ python 2.7 လို့ အရင်တုန်းက version တွေကတွေ့၍ pip ကို ထပ်ထည့်ပေးရပါတယ်။

** pip installation **

<https://bootstrap.pypa.io/get-pip.py> ကနေပြီးတွေ့၍ python script လေးကို အောင်ပြု၍ run လိုက်ရင် ရပါပြီး။ အသေးစိတ် သိချင်ရင်တွေ့ .. အောက်ပါလုပ်ကနေတစ်ဆင့် ဖတ်ရှုညွှန်ပါ။

<https://www.makeuseof.com/tag/install-pip-for-python/>

ဟုတ်ပြီး pip ကို သွင်းပြီး သွားပြုးနောက်.. Pip ကိုသုံးပြုးတွေ့၍ python module/package တွေကို install လုပ်ပါမယ်။ သာမှမခေါ်ပါဘူး.. pip install [module name] ပါ

```
C:\python3\Scripts>pip.exe install pytz
Collecting pytz
  Downloading https://files.pythonhosted.org/packages/61/28/1d3920e4d1d50b19bc5d24398a7cd85cc7b9a75a490570d5a30c57622d34/pytz-2018.9-py2.py3-none-any.whl (510kB)
    100% |████████████████████████████████| 512kB 1.8MB/s
Installing collected packages: pytz
Successfully installed pytz-2018.9
C:\python3\Scripts>cd
```

```
C:\Python27\Scripts>pip.exe install pytz
Collecting pytz
  Using cached https://files.pythonhosted.org/packages/61/28/1d3920e4d1d50b19bc5d24398a7cd85cc7b9a75a490570d5a30c57622d34/pytz-2018.9-py2.py3-none-any.whl
Installing collected packages: pytz
Successfully installed pytz-2018.9
You are using pip version 18.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Linux အတွက်ကတော့ `apt-get install python-pip`, `apt-get install python3-pip` ပါ။

pytz ဆိုတဲ့ ကောင်သွင်းပြီးနောက်မှတော့ timezone တွေရဲ့ current time ကို သိရမို့ code တွေရေးရာတို့..

Line 1 => datetime module ଓ କି datetime class କ୍ରି import ଲାଗିପିତାଯି।

Line 2 => pytz module ന് import ലൂട്ടിതയ്॥

Line 4 => `pytz.timezone()` အတွက် object ကို `tz_NY` ဆိုတဲ့ variable ကို သုံးပြီး တည်ဆောက် လိုက်ပါတယ်။

```
1 from datetime import datetime
2 import pytz
3
4 tz_NY = pytz.timezone('America/New_York')
5 datetime_NY = datetime.now(tz_NY)
6 print("NY time:", datetime_NY.strftime("%H:%M:%S"))
7
8 tz_London = pytz.timezone('Europe/London')
9 datetime_London = datetime.now(tz_London)
10 print("London time:", datetime_London.strftime("%H:%M:%S"))
```

Line 5 => ဒီနေရာမှာ `datetime.now()` နဲ့ `datetime.today()` နစ်ခုထဲက `now()` ကို ဘာလို့ သုံးလဲဆိုတောာ `today` သည့် keyword argument ကို လက်မခံနိုင်ပါဘူး။ `Now` မှာ မြတ် keyword argument တွေကို လက်ခံပါတယ်။ `default` အရ `datetime.now(tz = None)` ဖြစ်တာဖို့.. `datetime.today()` နဲ့ `result` ချင်းသွားတူနေတာပါ။ ခု `now()` `timezone` သတ်မှတ်လိုက်တာပါ။ `variable datetime_NY` ထဲမှာ `NewYork` မှာ ပါတ်သက်တဲ့ `date, time information` တော့ ထည့်လိုက်ပါတယ်။

Line 6 => strftime() ကို သုံးပြီးတော့ NewYork time ကို print ထုတ်လိုက်ပါတယ်။

```
NY time: 17:20:57
London time: 22:20:57

Process returned 0 (0x0)      execution time : 1.601 s
Press any key to continue . . .
```

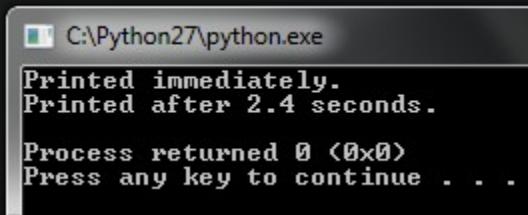
6.5 Python sleep()

Python ရဲ sleep() function ကို program အကြီးကြီးတွေ run ထဲ အခါ processing လုပ်ရတာများ လို့.. အသေးစိတ် code တွေကို ဆင်မ run ပဲ ခန် နားပြီးမှ ဆင် ရန် ချင်တာမိုးပဲ ဖြစ်ဖစ်

device(router/swtich) တွေမှာ show command တွေ config တွေရှင် 1sec 2sec လောက် စောင့်တာမျိုးပဲ ဖြစ်ဖြစ် စတဲ့ code ကို ဆက်မသွားပဲ ခန် wait စောင့်တာမျိုးတွေမှာ sleep ကို သုံးပါတယ်။

```

1 import time
2
3 print("Printed immediately.")
4 time.sleep(2.4)
5 print("Printed after 2.4 seconds.")
```



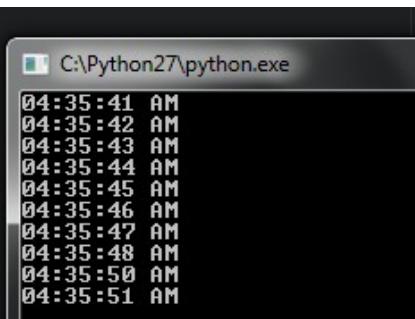
The terminal window shows two lines of output: "Printed immediately." followed by "Printed after 2.4 seconds.". Below the output, it says "Process returned 0 <0x0>" and "Press any key to continue . . .".

အပေါ်က example ကုတ်အရဆိုရင် sleep ကို သုံးဖို့အတွက် time module ကို import လုပ်ထားပါတယ်။ printed immediately ကို program စစ်ဆေး run တဲ့အချင်မှာ print ထုတ်ပါတယ်။ time.sleep(2.4) ဆိုတဲ့ code ကိုရောက်တဲ့အချင်မှာ python သည် 2.4 sec လောက် ခန်ရပ်သွားပါတယ်။ ပုံးမှ အောက်ရှာနိန္တဲ့ codeဖြစ်တဲ့ print("Printed after 2.4 seconds.") ကို execute လုပ်ပါတယ်။

နောက်တစ်ခု 1sec ခြားတစ်ခု အချင်ကိုပေးနေမယ်။ နာရီစန်ဆန် script တစ်ခုလောက် while loop ကို သုံးပြီး ရေးပြောင့်ရာရအောင်။

```

1 import time
2
3 while True:
4     localtime = time.localtime()
5     result = time.strftime("%I:%M:%S %p", localtime)
6     print(result)
7     time.sleep(1)
```



The terminal window shows a series of time outputs from 04:35:41 AM to 04:35:51 AM, each on a new line. The sleep(1) command causes the loop to pause for one second between each iteration.

Line no.1 => time module ကို import လုပ်ပါတယ်။ **Line no.3** while ခဲ့ condition နေရာမှာ True ဖြစ်နေတာမို့ infinite loop ပါတယ်နေပါလိမ့်မယ်။ **Line no.4 =>** localtime ဆိုတဲ့ variable ထဲက time ထဲ localhost() ကို ခေါ်ပြီးတော့ ခုလက်ရှိ ရှိနေတဲ့ အချင် data ကို ထည့်ပိုက်ပါတယ်။ datetime module သုံးလဲရတယ်နော့။ import ထပ်လုပ်ရတာပဲ ရှိမှာ။ **line no.5 time** ခဲ့ strftime () ကိုသုံးပြီး localtime ထဲက dataကို format ချပါတယ်။ ရလာတဲ့ ကောင်ကို result ဆိုတဲ့ variable ထဲကို ထည့်ပါတယ်။ **line no.6** format တစ်ရာနဲ့ရလာတဲ့ အချင်ကို print ထုတ်ပါတယ်။ **line no.7** တစ်စတုန်နားပြီး while ခဲ့ condition သည် True ပဲ ဖြစ်နေတာမို့.. loop စက်ပါတ်နေပါတွေတယ်။

7. Python File I/O (input/output)

Python မှာ `input` လုပ်တာသည် `raw_input()` ရယ် `input()` ရှိတယ်. အရောမှာလဲ အကြိမ်ကြိမ် သုံးခဲ့တော့ အထူးတစ်လည်ထပ်မရှင်းတွေဘူး.. ြီးတွော `output` ထုတ်ရင် `print()` ကိုသုံးြီး ထုတ်ခဲ့ရာတယ်။ `print` ကတော့ နဲ့ လေး ဖြည့်ပြောချင်တာရှိတယ်။

Line no.4 မှာရွေတွေ { } တွေထဲမှာ index No. တွေပါထည့်ထားပေးလို့.. {0} မှာ Index.0 ။ movie ကိုထည့်ပြီးတွေ {1} မှာတွေ Index.1မှာရှိနေတဲ့ books ကို ထည့်ပြီး print ထုတ်လိုက်မှာပါ။

```
1 print('I love {} and {}'.format('movies','books'))
2 # Output: I love movies and books
3
4 print('I love {0} and {1}'.format('movies','books'))
5 # Output: I love movies and books
6
7 print('I love {1} and {0}'.format('movies','books'))
8 # Output: I love books and movies
```

အဲတာအပ်၏ keyword argument တွေကို သုံးပြီးတော့လဲ print ထုတ်မထုံး string ကို format ရေးလိုပါတယ်။ ခုအောက်က example မှာဆို {name}ရယ် {greeting}ရယ် ရှိပါတယ်။ အဲအတွက် .format မှာ name အတွက်က Dear လို့ပဲ ရေးပေးထားပြီး Greeting အတွက်က GoodNight ဆိုပြီး keyword အတွက် သက်ဆိုင်ရာ data တွေ ထည့်ပေးထားပါတယ်။

Python ጉልፎች፡ የዚህ print() ምሳሌ በ “Hello” ነው እና የዚህ ደንብ እንደሚሆን ይህንን የሚመለከት ይችላል፡፡

ကိုယ်ပေါ်စေချင်တဲ့ data တွေကို .format()ထဲ မှာရေး .. ပြီးတော့ ထုတ်မယ့်အနီးနှင့်ရှာ ပေါ်စေချင်တဲ့ data အကြောင်း { } သုံးပြီးတော့ format ရှိက်ပါ။

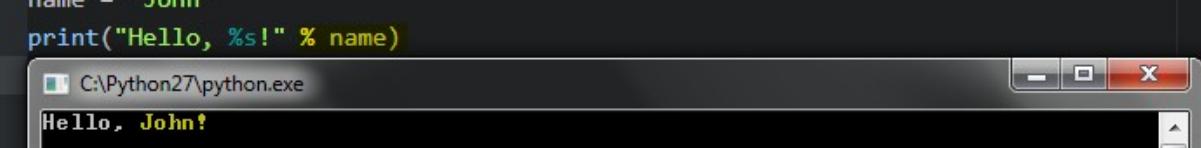
```
1 print('Hello {name}, {greeting}'.format(greeting = 'GoodNight', name = 'Dear'))
2 # output = Hello Dear, GoodNight
```

ခုအပေါ်မှာ ပြောခဲ့တဲ့ နည်းလမ်းကို သုံးလို့ ရသလို C programming မှာ တုန်းကလို % operator ကို သုံးပြီးတော့လဲ print ထုတ်မယ့် string ကို format ချ လိုပါတယ်။

```
1 # This prints out "Hello, John!"
2 name = "John"
3 print("Hello, %s!" % name)
```

Name ဆိုတဲ့ variable ထဲ ကို 'John' ဆိုတဲ့ string value တစ်ခု ထည့်ထားပါတယ်။ print ထုတ်လိုက်တဲ့ အနီးနှင့်မှာ %s ဆိုတာသည် string variable မှာ ရှိတဲ့ data ကို ထုတ်ရမယ့် နေရာကို ညွှန်းဆိုထားတာပါ။ အဲတာကြောင့် Hello, နောက်မှာ John ဆိုပြီး ထွက်လာတာပါ။

```
1 # This prints out "Hello, John!"
2 name = "John"
3 print("Hello, %s!" % name)
4
```



ပိုပြီး နားလည်သွားအောင် နောက်ထပ် example တစ်ခုနဲ့ ထပ် စမ်းကြေည့်ရှာရအောင်။ ခုဒီ ဥပမာ မှာ ဆိုရင် %s နဲ့ %d ဆိုတာရှိတယ်.. %s သည် string variable ထဲ ရှိတဲ့ data (MgMg)ကို ထုတ်မယ့် နေရာဖြစ်ပြီး တော့ %d သည် decimal integer (ဒီနေရာမှာတွေ့ 27) ကိုထုတ်မယ့် နေရာကို သတ်မှတ်ပေးထားတာပဲ ဖြစ်ပါတယ်။ ထုတ်ရမယ့် နေရာပြီးတော့ ထုတ်မယ့်ကောင်တွေကို % နောက်ကနေ ရေးပေးထားပါတယ်။ ငါအောက်ကာကုပ်လေးကို run ကြားလိုက်ပါ။

```
1 print ('My name is %s. I am %d year old!' %('MgMg', 27))
2 #output >> My name is MgMg. I am 27 year old!
```

ခုအောက်က list မှာ မှုမှု % operator တွေနဲ့ သူတို့ ရဲ့ အသုံးတွေပဲ ဖြစ်ပါတယ်။

Escape	Description	Example
%d	Decimal integers (not floating point)	"%d" % 45 == '45'
%i	Same as %d	"%i" % 45 == '45'
%o	Octal number	"%o" % 1000 == '1750'
%u	Unsigned decimal	"%u" % -1000 == '-1000'
%x	Hexadecimal lowercase	"%x" % 1000 == '3e8'
%X	Hexadecimal uppercase	"%X" % 1000 == '3E8'
%e	Exponential notation, lowercase "e"	"%e" % 1000 == '1.000000e+03'
%E	Exponential notation, uppercase "E"	"%E" % 1000 == '1.000000E+03'
%f	Floating point real number	"%f" % 10.34 == '10.340000'
%F	Same as %f	"%F" % 10.34 == '10.340000'
%g	Either %f or %e, whichever is shorter	"%g" % 10.34 == '10.34'
%G	Same as %g but uppercase	"%G" % 10.34 == '10.34'
%c	Character format	"%c" % 34 == '' '
%r	Repr format (debugging format)	"%r" % int == "<type 'int'>"
%s	String format	"%s there" % 'hi' == 'hi there'
%%	A percent sign	"%g%%" % 10.34 == '10.34%'

အရှေ့မှာတုန်းက detail မပြောခဲ့ရတော့ .. အဲတာလေးကို သိစေချင်လို့ပါ။ ကဲ အမိက အကြောင်း အရာ ဖြစ်တဲ့ file တွေ ဖွင့်တာပိတ်ထား data အသွင်း အထုတ် လုပ်တာ။ ဖုက်ပစ်တာ စတာတွေကို လေးလာ ရှာဖို့။

7.1 Python File Handling

ဟုတ်ပြီး file ဆိုတာ ဘာလဲဆိုတော့ non-volatile memory (eg., Hard disk) ပေါ်မှာ နာမည်ပေးပြီး data တွေ သိလောင် သိမ်းဆည်း ထားတဲ့ ကောင်တွေကို ဖိုင် လို့ ခေါ်ပါတယ်။ အရင်ဆုံး file တစ်ဖိုင်ကို handle လုပ်မယ်ဆို သိသင့်တဲ့ function သည် open() ပဲ ဖြစ်ပါတယ်။ read မှာပဲဖြစ်ဖြစ် write မှာပဲ ဖြစ်ဖြစ် အေရင်တွေ့ open ရမှာပဲလေး။ open မှာ parameter နှစ်ခု ပါ ပါတယ်။ open လုပ်မယ့် filename ရယ် ဘယ်လို့ mode နဲ့ဖွင့်မလဲ ဆိုတာပါ။ သူ့ရဲ့ syntax သည် `open("filename", "mode")`ပဲ ဖြစ်ပါတယ်။ mode မှာ မေတ္တာများ.. စသည်ဖြင့် အလုပ် လုပ်မယ့် mode ကို ရေးပေးရတာပါ။ အကယ်လို့ .filename ပဲ ရေးထားတယ်ဆိုရင် default mode သည် 'rt' ဖြစ်ပါတယ်။ 'r' သည် read ကို ကိုယ်တွေ့ပြုပြီး 't' သည် text ဖိုင်ကို ကိုယ်တွေ့ပြုပါတယ်။

`myfile = open("demofile.txt")` ဆိုပြီးတော့ open() function နောက်မှ ဖွင့်မယ် ဖိုင် နာမည်လေး ရေးပေးလိုက်ပါတယ်။ open() ကို ခေါ်လိုက်တဲ့ အနှစ်မှာ myfile ဆိုပြီးတော့ return ပြန်လောမယ့် ကောင်အတွက် object တစ်ခု တည်ဆောက်လိုက်ပါတယ်။ အဲနောက် အဲဖိုင်ထဲမှာ ဘာတွေရှုလဲ သိဖို့ အတွက် read() method ကို သုံးပြီးတော့ ဖတ်ရှုည့်ပါမယ်။ အပေါ်မှာလဲ ပြောခဲ့တယ်နော.. Mode ပြောမထားရင် default



mode သည် 'rt' ဖြစ်တာဖို့.. ဖော်လို့ရပါတယ်။ တစ်ခုမှတ် ထားရမှာက open() လုပ်ပြီးရင်တွာ best practice အနေနဲ့ ပြန်ပိတ်ခဲ့ဖို့လိုတယ်နော်။ close() function ကို သုံးပြီး ပိတ်ခဲ့ပါ။

7.1.2 read file

```
1 myfile = open("kp.txt")
2 #same as >> myfile = open("kp.txt", "rt")
```

print(myfile.read()) ဆိုပြီးတွေ့၍ read() method ကို သုံးပြီး ဖော်လို့ရလာတာတွေကို user မြင်ရအောင် print ထုတ်ပြုပါတယ်။

```
1 myfile = open("kp.txt")
2 #same as >> myfile = open("kp.txt", "rt")
3 print(myfile.read())
4
```

C:\Python27\python.exe
Hello, Bro.
Nice to see you!
How do you do?

read() နဲ့ ဖော်လိုက်တဲ့ အချိန်မှာ kp.txt ထဲမှာ ရှိသမှု အကုန် ထွက်လာပါတယ်။ အဲလိုမဟုတ်ပဲ ကို ထိဖတ်ချင်သလောက် စာလုံးအရေးအတွက်နဲ့ပဲ ဖြစ်ဖြစ် တစ်ကြောင်းချင်းပဲ ဖြစ်ဖြစ်လဲ ဖော်လို့ရပါတယ်။ read(5) ဆိုရင်တွာ 5 သည် ဖော်မယ့် စာလုံးအရေအတွက် ၅ လုံး ဖော်မယ်လို့ ပြောတာပါ။ အကယ်လို့ 10 ဆိုရင် စာလုံးရေ ၁၀ လုံး ထိ ဖော်ပြီး print ထုတ်ပေးသွားမှာပဲ ဖြစ်ပါတယ်။

```
1 myfile = open("kp.txt")
2 #same as >> myfile = open("kp.txt", "rt")
3 print(myfile.read(5))
4
```

C:\Python27\python.exe
Hello
Process returned 0 (0x0)
Press any key to continue . .

readline() ကတွေ့၍ တစ်ကြောင်းချင်းစီ ဖော်ပြီးတွေ့၍ print ထုတ်ပေးတဲ့ ပုံစံပါ။ readline() တစ်ခါ လို့ ပထမတစ်ကြောင်းဖော်တယ်.. နောက်တစ်ခါ readline() ဆို နောက် ဒုတိယ တစ်ကြောင်း ဖော်တယ် .. မေးလိမ့်နဲ့.. တစ်ကြောင်းချင် ဖော်သွားပါတယ်။

```
1 myfile = open("kp.txt")
2 #same as >> myfile = open("kp.txt", "rt")
3 print(myfile.readline())      #print first line
4 print(myfile.readline())      #print second line
5
```

C:\Python27\python.exe
Hello, Bro.
Nice to see you!
Process returned 0 (0x0)

ဒါမှုမဟုတ် for loop ပါတ်ပြီးတွေ့လဲ text ဖိုင်ထဲမှာ ရှိသမှု လိုင်းတွေကို တစ်ကြောင်းချင်း မကုန် မရ၏။ loop ပါတ်ပြီး ထုတ်သွားလို့ ရပါတယ်။

```
1 myfile = open("kp.txt")      Hello, Bro.  
2 for x in myfile:           Nice to see you!  
3     print(x)                How do you do?
```

ဂုဏ်ပြုက ဖိုင်တွေ အားလုံးသည် same directory/folder ထဲမှာ ရှိနေတာပါ။ တစ်ခြား different ထဲမှာ ရှိနေတယ် ဆိုရင်တွေဘူး Path လမ်းကြောင်းကို ထည့်ရေးပေးဖို့ လိပ်တယ်။ python 2.7 မှာက ပုံမှန် path လမ်းကြောင်းကို ရေးရင် ရပေးမလို့ 3.7 မှာရှာတွော \ (single slash) တွေနေရာမှာ double slash (\\) နဲ့ ရေးပေးရပါတယ်။ အောက် က screen shot ကို လေးလာ ဖြေထုတ်ပါ။ C: drive ပါမောင်။ Users အောက်က ekhaphy ပါမောင်။ Dowloads ဆိုတဲ့ folder/directory ထဲက differentpathfile.txt လေးကို လုမ်းပြီးတွော open() လိုက်တာပါ။

```
1 #for Python 3.7
2 myfile = open("C:\\\\Users\\\\ekhaphy\\\\Downloads\\\\differentpathfile.txt")
3
4 #For Python 2.7
5 myfile = open("C:\\Users\\ekhaphy\\Downloads\\differentpathfile.txt")
6 print(myfile.read())
7
```

This is first Line
This is second Line
This is third Line
This is F

ဖိုင်တွေ ဖွင့်ထားလား ပိတ်ထားလား .. ပြီးတွော ဘာ mode နဲ့ အလုပ်လုပ် နေလဲ file name စသည်။ အင် ။.. စစ်ချင်ရင်တွော .. နာမည်စစ်မယ်ဆို fileobject.name() ။ ဖွင့်တာ ပိတ်တာ စစ်မယ်ဆို fileobject.closed ။ mode ကို စစ်မယ်ဆို fileobject.mode() ။ စသည်ဖြင့် print ထုတ်ကြည့်နိုင်ပါတယ်။

```
5 myfile = open("kp.txt")
6 print "Name of the file: ", myfile.name
7 print "Closed or not : ", myfile.closed
8 print "Opening mode : ", myfile.mode
```

Name of the file: kp.txt
Closed or not : False
Opening mode : r

Process returned 0 (0x0) execution

Press any key to continue . . .

သာလို့ အဲလိုတွေစစ်လဲ ဆိုတော့ ကိုယ့် script အရ ဖွင့်တဲ့ mode တွေ file ပိတ်ပြီးပြီလား ဖွင့်ထားတုန်းလား.. စသည်၍ စစ်ဖိုလိုလာရင် စစ်လို့ ရအောင်ပါ။

စစ်ဆေးမှာ တုန်းကလဲ ပြောခဲ့တယ်နော.. ဖိုင်တွေကို ဖွံ့ဖြိုးရင် ပြန်ပိတ်ဖို့.. ပိတ်တာကတော့ လွယ်ပါတယ်။ syntax :: fileobject.close() ပါ။



```
4 #For Python 2.7
5 myfile = open("kp.txt")
6 print "Name of the file: ", myfile.name
7 print "Closed or not : ", myfile.closed
8 print "Opening mode : ", myfile.mode
9
10 myfile.close()
```

အောက်က ဖယ်စာတွေသာ python ရဲ့ file mode တွေနဲ့ သူတို့၏ လုပ်ဆောင်ရွက်တွေပါ။

7.1.2 Python file mode

Mode	Description
'r'	Open a file for reading. (default)
'w'	Open a file for writing. Creates a new file if it does not exist or truncates the file if it exists.
'x'	Open a file for exclusive creation. If the file already exists, the operation fails.
'a'	Open for appending at the end of the file without truncating it. Creates a new file if it does not exist.
't'	Open in text mode. (default)
'b'	Open in binary mode.
'+'	Open a file for updating (reading and writing)

7.1.3 Python write file

write() method ။ မြတ်သူ open လုပ်ထားတဲ့ file မှာ data string တွေ သွားရေးပေးမယ့် ။ ကောင်ပဲ ဖြစ်ပါတယ်။ ဒီနေရာမှာ တစ်ခုမှတ် ထားရမှာက file သည် text ဖိုင် သာ မကပဲ binary file လဲ ဖြစ်နေနိုင်တာပါပဲ။ write() ရဲ့ syntax သည် fileobject.write(string) ပဲ ဖြစ်ပါတယ်။

```

serial.py          ForLoop.py      ForLoopLesson.py    testpython.py      kp.txt        FileBas
1 # Open a file
2 myfile = open("kp.txt", "wt")
3 myfile.write( "Python is a great language.\nYeah its great!!\n");
4
5 # Close open file
6 myfile.close()

```

ခုခြုံအပေါ်က ဥပမာမှာ ဆိုရင် kp.txt ဖိုင် အတွက် write mode နဲ့ myfile ဆိုတဲ့ object တစ်ခု တည်ဆောက်လိုက်ပါတယ်။ အဲနောက် write() ထဲ မှာ ထည့်ရေးမထုတ် စာသားတွေကို ရေးပါတယ်။ write() မှာ \n ကို သုံးပြီး တစ်ငြောင်း ချင်းဆီ ဖြစ်အောင် ရေးထားပါတယ်။ code တွေကို execute လုပ်ပြီးနောက် kp.txt ဖိုင် ကို ဖွင့်ကြည့်လိုက်ရင် အောက်ပါ အတိုင်း တွေ ရမှာပဲ ဖြစ်ပါတယ်။

```

serial.py          ForLoop.py      ForLoopLesson.py    testpython.py      kp.txt        FileBas
1 Python is a great language.
2 Yeah its great!!
3

```

ခုခြုံအက်က example မှာရှာတွော this is the line 1, this is the line 2to.....this is the line 10 ထိ ထုတ်အောင် range() ကို သုံးပြီးတွော loop ပါတ်ပြီး write ထားတာပါ။

```

serial.py          ForLoop.py      ForLoopLesson.py    testpython.py      kp.txt        FileBas
1 f= open("kp.txt", "w+")
2 for i in range(10):
3     f.write("This is line %d\n" % (i+1))
4 f.close()

```

Result ကိုသွားကြည့်တဲ့ အချိန်မှာ ဘယ်လောက် အကြိမ်ကြိမ် run run .. this is the line 1, this is the line 2to.....this is the line 10 ဆိုတဲ့ ကောင်သည် မပြောင်လဲ ပဲ ရှိနေမှာပါ။ ပြောချင်တာပါ.. Overwrite ဖြစ်နေတွော ထပ် မတိုးတာကို ပြောတာပါ။ w+ နဲ့ရေးရင် .. သူသည် file နာမည် (kp.txt) နဲ့ရှိရင် အဲ ဖိုင်ကို ဖွင့်ပြီး သွား write ပါတယ်။ မရှိရင် create လုပ်ပြီး write လုပ်ပါတယ်။

serial.py	ForLoop.py	ForLoopLesson.py	testpython.py	kp.txt	FileBas
1 This is line 1					
2 This is line 2					
3 This is line 3					
4 This is line 4					
5 This is line 5					
6 This is line 6					
7 This is line 7					
8 This is line 8					
9 This is line 9					
10 This is line 10					
11					

အဲလို overwrite မဖြစ်စေခဲ့ရင် ဖိုင်ကို append mode နဲ့ ဖိုင်ကို ဖွင့်မှ ရပါမယ်။ f = open("kp.txt","a+") ဆိုပြီးတော့ w နေရာမှာ a ကို ပြောင်းထားရှာည့်ပြီး နှစ်ကြိမ် လောက် run ရှာည့်ပါ။ overwrite မဖြစ်တော့ပဲ.. ထပ် ထပ် တိုးပြီး ရေးသွားတာကို တွေ့ရမှာပါ။

7.2 Python OS module

Python ရဲ့ OS module မှာရှိထဲ method တွေသည် file-processing လုပ်ရာမှာ အထောက်အကူ၏
ပုဂ္ဂနယ်။ ဘယ်လိုမျိုးတွေလဲဆိုတော့ ဖိုင်တွေ ဖုက်တာ နာမည်ပြောင်းတာ directory တည်ဆောက်တာ စ
တဲ့ နည်းလမ်းမျိုးတွေပဲ ဖြစ်ပါတယ်။ ဆက်ပြီး လေ့လာရာဉ်ရှာ ရအောင်ဖြာ။

7.2.1 Renaming and deleting file

rename() method သည် သူ.နာမည် အတိုင်းပဲ နာမည်ပြောင်းပေးမှာပါ။ သူ.syntax မြတ်မြတ် os.rename(current_file_name, new_file_name)ပဲ ဖြစ်ပါတယ်။ လက်ရှိ ပြောင်းမယ့် file ကို အရှေ့ကရေး ပြောင်းချင်တဲ့ နာမည်ကို နောက်က ရေးလိုက်ယုံပါပဲ။ os module ကိုတွော import လုပ်ပေးရမယ်နော်။ အဲတော့မှ ဒါ os ထဲက method တွေကို သုံးလှ့ ရမှာပါ။

```
1 import os  
2  
3 # Rename a file from kp.txt to reKP.txt  
4 os.rename( "kp.txt", "reKP.txt" )
```

Remove() Method

`remove()` method ကတော့ ဖိုင်တွေကို ဖျက်ပစ်တဲ့ နေရာမှာသုံးတာပါ။ သူ့ syntax မှာ မြန်မာစာတွေကို `os.remove(file_name)`ပါ .. ဖျက်မယ့်ကောင်ကို `remove()` မှာ ရေးပေးလိုက်ယုံပါပဲ။

7.2.2 Python Directory Process

`mkdir()`

`mkdir()` method ကို လက်ရှိ directory မှာ directory တစ်ခု ထပ်ဆောက်ဖိုအတွက် သုံးပါတယ်။ `mkdir` ဆိုတာသည် `make directory` ပါ။ သူ့ syntax ကတော့ `os.mkdir("newdir")`ပါဖြစ်ပါတယ်။

`chdir()`

`chdir()` method ရာတော့ ခုလက်ရှိ ရှိနော်တဲ့ directory ကနေ တစ်ခြား directory ကို သွားချင် ပြေ ဟင်းချင်တဲ့ အခါမြို့မှာ သုံးပါတယ်။ သူ့ syntax မှာ မြန်မာစာတွေကို `os.chdir("newdir")`ပါဖြစ်ပါတယ်။ သူသည် different path လမ်း ပြောင်းမှာ ရှိတဲ့ directory တွေကို သွားရောက်နိုင်ပါတယ်။

```
os.chdir("/home/newdir")
```

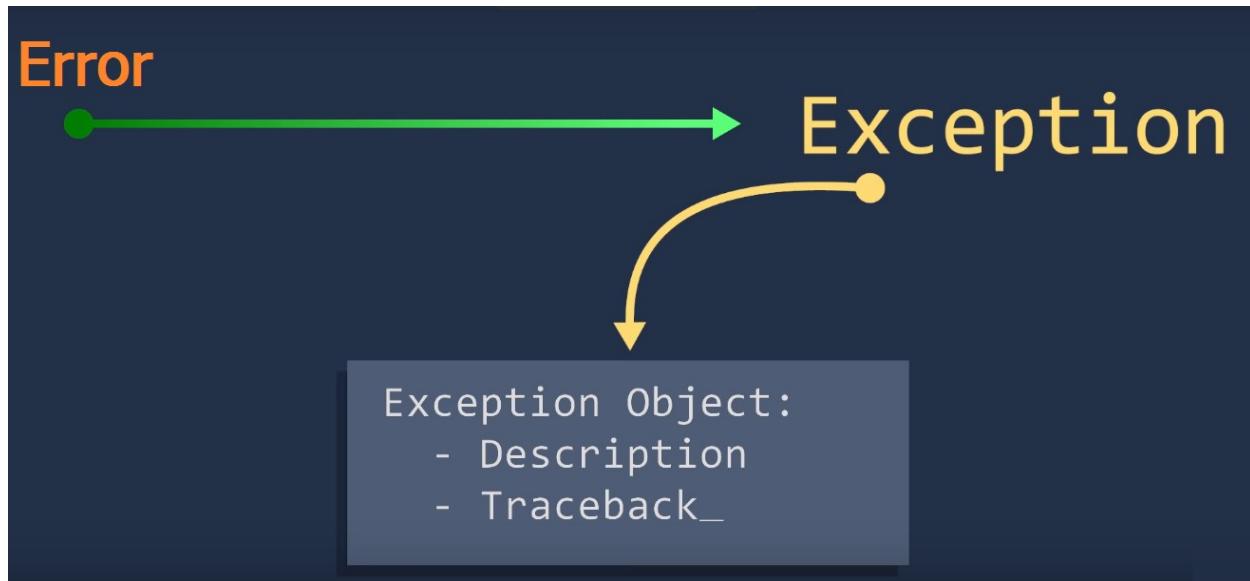
`rmdir()`

`rmdir()` ဆိုတာသည် `remove directory` ပါ။ သူသည် မလိုတော့တဲ့ ဖျက်စေချင်တဲ့ directory က တွေကို ဖျက်တဲ့ နေရာမှာ သုံးပါတယ်။ သူ့ syntax မှာ မြန်မာစာတွေကို `os.rmdir('dirname')`ပါဖြစ်ပါတယ်။

8. Python Error Handling

Python ရဲ့ interpreter ကနေပြီးတော့ error တွေ တွေ့ရင် exception ကို `raises` လုပ်ပါတယ်။ Python ရဲ့ runtime မှာ ဖြစ်တဲ့ error တွေကို `exception` တွေလို့ ခေါ်ပါတယ်။ ဖြစ်တဲ့ error အမျိုးအစားငါး မူတည်ပြီးတော့ Python မှာ built-in exception တွေ ရှိပါတယ်။ ဥပမာ ပြောရင် ကိန်းတစ်ခုကို zero နဲ့ သွားစားတာမျိုးဆို `ZeroDivisionError`, မရှိတဲ့ file တစ်ခုကို သွားဖွင့်လိုက်မျိုးမှာဆိုရင် `FileNotFoundException`, မရှိသေးတဲ့ module ကို `import` လုပ်ခဲ့ရင် `ImportError` စသည်ဖြင့်.. အကြောင်းအရာပေါ်မူတည်ပြီး error မျိုးစုံတက်ရာပါတယ်။

အဲလို `runtime error` တွေ ဖြစ်လာတာနဲ့ Python သည် `exception object` ကို `create` လုပ်ပါတယ်။ အဲ error exception ကို `handle` မလုပ်ခဲ့ရင် Python သည် `traceback` လုပ်လို့ရအောင် ဘယ်နေရာ မှာ ဘယ်လိုဖြစ်တယ် ဘာကြောင့်ဖြစ်တယ်ဆိုတာကို `print` ထုတ်ပေးပါတယ်။



Example အနေနဲ့ `1/0` ဆိုပြီးတော့ 1 ကို 0 နဲ့ စားခိုင်းပါတယ်။ 1 ကို 0 နဲ့ စားလို့ မရဘူးဆို Python အနေနဲ့။ သိတော်မြတ်တဲ့အတွက် `exception` သည် `error` ပေါ်မှုတည်ပြီး ထွက်လာတာမူလိုက်လိုက် `ZeroDivisionError` ဆိုတဲ့ `exception` ကို အောက်ပါ အတိုင်း မြင်တွေ့နိုင်ပါတယ်။ `line 1` ဆိုတာကတော့ ဒု မှ python ကို စဖွင့်ဖွင့်ချင်း ရေးလိုက်တာမူလိုက်။ `Line 1` ဖြစ်တာပါ။ `enter` ခေါက်ပြီးမှ ရေးပြောသူလို့ `error` ။ ။ ။ ရင် `enter` နှစ်ခေါက်ပြီး သုံးပြေားမြောက်မှာ ရေးတဲ့ `code` ။ `error` တက်ရင် `line 3` ဆိုပြီး မြင်ရမှာပဲ ဖော်ပါတယ်။

```
>>> 1/0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero
```

Python မှာ built-in အနေနဲ့ ရှိနေတဲ့ `exception` အမျိုးအစားတွေကို အောက်ပါတိုင်း တွေမြင်နိုင်ပါတယ်။ `exception` တွေနဲ့ သူတို့ကို ဖြစ်စေတဲ့ .. `error` တွဲပဲ ဖြစ်ပါတယ်။

Exception	Cause of Error
<code>AssertionError</code>	Raised when <code>assert</code> statement fails.
<code>AttributeError</code>	Raised when attribute assignment or reference fails.



EOFError	Raised when the <code>input()</code> functions hits end-of-file condition.
FloatingPointError	Raised when a floating point operation fails.
GeneratorExit	Raise when a generator's <code>close()</code> method is called.
ImportError	Raised when the imported module is not found.
IndexError	Raised when index of a sequence is out of range.
KeyError	Raised when a key is not found in a dictionary.
KeyboardInterrupt	Raised when the user hits interrupt key (<code>Ctrl+c</code> or <code>delete</code>).
MemoryError	Raised when an operation runs out of memory.
NameError	Raised when a variable is not found in local or global scope.
NotImplementedErr or	Raised by abstract methods.
OSError	Raised when system operation causes system related error.
OverflowError	Raised when result of an arithmetic operation is too large to be represented.
ReferenceError	Raised when a weak reference proxy is used to access a garbage collected referent.
RuntimeError	Raised when an error does not fall under any other category.

StopIteration	Raised by <code>next()</code> function to indicate that there is no further item to be returned by iterator.
SyntaxError	Raised by parser when syntax error is encountered.
IndentationError	Raised when there is incorrect indentation.
TabError	Raised when indentation consists of inconsistent tabs and spaces.
SystemError	Raised when interpreter detects internal error.
SystemExit	Raised by <code>sys.exit()</code> function.
TypeError	Raised when a function or operation is applied to an object of incorrect type.
UnboundLocalError	Raised when a reference is made to a local variable in a function or method, but no value has been bound to that variable.
UnicodeError	Raised when a Unicode-related encoding or decoding error occurs.
UnicodeEncodeError	Raised when a Unicode-related error occurs during encoding.
UnicodeDecodeError	Raised when a Unicode-related error occurs during decoding.
UnicodeTranslateError	Raised when a Unicode-related error occurs during translating.



ValueError	Raised when a function gets argument of correct type but improper value.
ZeroDivisionError	Raised when second operand of division or modulo operation is zero.

8.1 Python try except

အပေါ်မှာ ဆိုခဲ့သလို ပဲ.. Error ရှိရင် exception ရှိတာမိုး.. ဖြစ်လာတဲ့ exception ကို handle မလုပ်ရင် ခုလက်ရှိ run နေတဲ့ process သည် error ဘာကြောင့်ဖြစ်တဲ့ ဆိုတဲ့ error message ကို မြင်ရပြီးက တဲ့ stop ဖြစ်သွားပါတယ်။ ပြောရင်တော့ crash ဖြစ်သွားတာပေါ်များ..

အဲတာကြောင့် error ဖြစ်နိုင်တဲ့ နေရာတွေကို exception ကို handle လုပ်သင့်ပါတယ်။ exception ကို handle လုပ်မယ်ဆို ပထမဆုံး **try** **except** block နဲ့ simple code လေး စရေးပြီး လေ့လာသင် ယူ ကြားလုပ်ရအောင်များ။

```

1 import sys
2
3 try:
4     result = 50/0
5     print ("The result is " + str(result))
6 except:
7     print("Oops!",sys.exc_info(),"occured.")
8
9
10 C:\Python27\python.exe
Oops! <<class 'ZeroDivisionError'>, ZeroDivisionError('division by zero'), <traceback object at 0x00BC1080> occured.

```

အရှေ့က example တစ်ခုဖြစ်တဲ့ 0 (Zero) နဲ့ တားလို့ ရလာတဲ့ exception ကို ပြန်စမ်ကြည့် ရာရဇ် အောင်။

sys module ကို import လုပ်ပါ။ line no.7 မှာရှိတဲ့ sys.exc_info() ၏ sys module ရှိမှ သုံးလို့ ရမှာပါ။ try: block သည် တစ်ကယ် run မယ့် code တွေ ထားမယ့်နေရာပါ။ except: block ၏ အောက်တွေ try block ၏ အောက်တွေ error တက်လာမှ ဖြစ်တာလဲ။ exception ကို handle လုပ်ဖို့အတွက် run မှာပါ။ try: block မှာ error မတက်ခဲ့ရင် except: သည် ဘာမှမလုပ်တော့ပဲ ရှုံးသွားပါတယ်။

ဒဲ **try: block** မှာက result = 50/0 ဆိုတဲ့ အတွက် 50 ကို 0 (Zero) နဲ့ တားတဲ့ အတွက် error ၏ ကိပ်လိမ့်မယ်။ ပုံမှန်ကုတ်ဆိုရင်တော့ error တက်ပြီး crash ဖြစ်သွားမှာပါ။ ခဲ့က **try: except:** ကို သုံးပဲ

ဒီတွေ exception ကို handling လုပ်တာမူး.. Crash မဖြစ်တွေပဲ.. **except : block** ကို ရောက်သွားပါတယ်။

`print("Oops!", sys.exc_info(),"occurred.")` ဆိတဲ့ code အရ .. `sys.exc_info()` သည် ချလက်ရှိ handle လုပ်နေတဲ့ exception နဲ့ပါတ်သက်တဲ့ information 3 ခုကို tuples အနေနဲ့ ထောပြပေးပါတယ်။ (`<class 'ZeroDivisionError'>`, `ZeroDivisionError('division by zero')`, `<traceback object at 0x007E2030>`) ဆိတဲ့ error နဲ့ပါတ်သက်တဲ့ info ရုရှိပေးပါတယ်။

```
try:
    # Runs first
    < code >
except:
    # Runs if exception occurs in try block
    < code >
```

အဲလို error info သုံးခဲတကမှ ကိုယ်ကြိုက်နစ်ရာကို လဲ စိတ်ကြိုက်ထုတ်လို့ ရပါတယ်။ အောက်က ကုတ်ကို ကြေည့်ပါ။

```
1 import sys
2
3 try:
4     result = 50/0
5     print ("The result is " + str(result))
6 except:
7     print("Oops! This is first except info " + str(sys.exc_info()[0]) + " occurred.")
8     print("Oops! This is second except info " + str(sys.exc_info()[1]) + " occurred.")
9     print("Oops! This is third except info " + str(sys.exc_info()[2]) + " occurred.")
10
11
12 # C:\Python27\python.exe
# Oops! This is first except info <class 'ZeroDivisionError'> occurred.
# Oops! This is second except info division by zero occurred.
# Oops! This is third except info <traceback object at 0x00861148> occurred.
```

ခုအပေါ်က example ကုတ်မှာရာတွေက `sys.exc_info()`[0], [1] စသည်ဖြင့် ရေးပြီးတွေ `tuple` ထဲမှာ ရှိနေတဲ့ 3 ခုကို index အလိုက်ထုတ်သွားတာပါ။

Except can Handle more than one exception?

ဒဲ `except:` သည် Error တစ်ခုထက်မက ထွက်လာင်ကော့ အလုပ်လုပ်လား စမ်းကြေည့်ရာရအောင်။ အောက်က ကုတ်လေးကို editor တစ်ခုခု နဲ့ ရေးပြီး save လိုက်ပါ။

Line no.3 မှာ data မတူတာတွေကို random list ဆိုပြီး create လုပ်ထားပါတယ်။ line no.8 မှ formula မှာ input အနေနဲ့ ထည့်ဖိုးအတွက်ပါ။

List ထဲက data တွေ တစ်ခုပြီး တစ်ခု ထည့် ထည့် သွားဖိုးအတွက် for loop ပါတယ်။ try: block ထဲ က code ကို အရင် run တဲ့အတွက် print အကြောင်းကတွာ အဆင်ပြေပါလိမ့်မယ်။ ခုလက်ရှိ list ထဲက data entry ကို ထုတ်ပေးတာပါ။

နောက်တစ်ကြောင်းမှာရာ r = 1/int(entry) code အတွက် entry ထဲက data ကိုထုတ်လိုက်တဲ့ အခိုန်မှာ a ဖြစ်ပါတယ်။ a integer ဖြောင်းပြီး 1 ကိုသွားစားတဲ့ အတွက် error တက်ပါတယ်။ try: block မှာ error မှ ကိုလို့ except: block ကို ရောက်သွားပါတယ်။

```

1 import sys
2
3 randomList = ['a', 0, 2]
4
5 for entry in randomList:
6     try:
7         print("The entry is", entry)
8         r = 1/int(entry)
9         break
10    except:
11        print("Oops!",sys.exc_info()[0],"occurred.")
12        print("Next entry.")
13        print()
14 print("The reciprocal of",entry,"is",r)
15

```

except: block မှာက sys.exc_info()[0] ကို သုံးပြီး exception တစ်ခုကို ထုတ်ပြထားပါတယ်။ ခုလက်ရှိ error နဲ့ ပါတာသက်လို့ handle လုပ်ထားတဲ့ exception သည် (a နဲ့ စားတာဖြစ်လို့ Entry = a) value error ဖြစ်ပါတယ်။ 1 စားတဲ့ value သည် a(integer) ဖြစ်တာမို့ပါ။ print("Oops!", sys.exc_info(),"occurred.") ဆိုပြီးတွေ့ လက်ရှိ exception info ကို print ထုတ်ပြပါတယ်။

```

The entry is a
Oops! <class 'ValueError'> occurred.
Next entry.

The entry is 0
Oops! <class 'ZeroDivisionError'> occurred.
Next entry.

The entry is 2
The reciprocal of 2 is 0.5 ✓

Process returned 0 (0x0)      execution time : 0.208 s
Press any key to continue . . .

```

random list ထဲမှာ empty မဖြစ်သေးတာမို့ Loop ဆက်ပါတယ်။ try: block မှ code တွေ ကို အရင် run ပါတယ်။ ခုခုတိယ entry ဖြစ်တဲ့ 0 (zero) ကို formula မှာထည့်လိုက်တော့ $r = 1/0$ ဖြစ်တာ မို့။ error တက်ပြီး except: block ကို ရောက်သွားပါတယ်။ ဒီတစ်ခါရွာတွော Handle လုပ်ထားတဲ့ exception object သည် $1/0$ နဲ့တော့ `ZeroDivisionError` ဆိုပြီးတော့ result ထုတ်ပေးပါတယ်။

နောက်တစ်ခါ for loop ဆက်ပါတယ်။ ဒီတစ်ခါ entry ၁၅၀၀၀၀၀၀၂ ဖြစ်ပါတယ်။ try: block ထဲက code ကို အရင်စ run ပါတယ်။ ၁ ကို တားတာက ၂ ဖြစ်တာမို့.. Error မတက်တွောပဲ.. Result ဖြစ်တဲ့ ၀.၅ ကို r ထဲ ထည့်လိုက်ပါတယ်။ try: block မတက်တာမို့ except: block က မ run ပါဘူး။

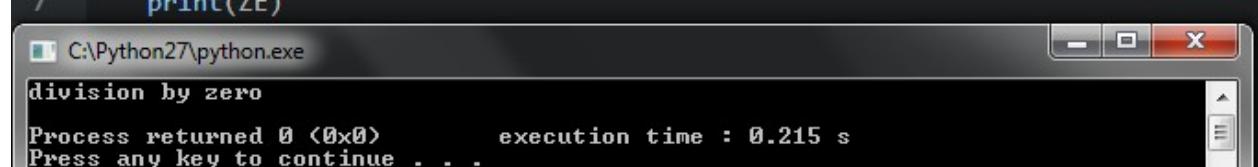
မြတ်မှုများ random list ထဲမှာ ဘာမှမရှိတော့တာမို့ for loop အဆုံးသတ်သွားပြီးတော့ for loop block ကနေထွက်ပြီး အောက် က `print("The reciprocal of",entry,"is",r)` ဆိုတဲ့ code ကို ရောက်ပါတယ်။ အတွက် `The reciprocal of 2 is 0.5` ဆိုတဲ့ result ရပြီး program သည် finished ဖြစ်သွားပါတယ်။

ဟုတ်ပြီး ခုနောက်ဆုံး run ခဲ့တဲ့ example code မှာဆို error တွေက တစ်ခုမက တက်နိုင်တဲ့အတွက် except: ဆိုပြီးရေးလိုက်တာပါ။ အကယ်လို့ ကိုယ်ကဖြစ်နိုင်ပယ့် error ကို ကြိုးသိရင်တော့ အောက်ပါ ကုတ်အတိုင်း တစ်ခုက်စမ်းရွှေ့ည့်လို့ ရပါတယ်။

```

1 import sys
2
3 try:
4     result = 50/0
5     print ("The result is " + str(result))
6 except ZeroDivisionError as ZE:
7     print(ZE)

```



ခုအပေါ်က ကုတ်အရ try: block ကတွော ရှင်းပါတယ်။ ၅၀ ကို `zero(0)` နဲ့တားတာဖြစ်လို့ error တက်ပါလိမ့်မယ်။ except block မှာရွာတွော အရင်လို့ မဟုတ်ပဲ except ZeroDivisionError as ZE: ဆိုပြီးတော့ `ZeroDivisionError` နဲ့တားရင် ဖြစ်မယ့် error ကိုဖမ်းပြီး(handle) ZE အနေနဲ့တားလိုက်ပါတယ်။ အဲနောက် ZE ကို `print` ထုတ်လိုက်တာပါပဲ။ ;-)



အဲတော့ ကိုယ်က ကုတ်ရေးမထဲသူပဲ.. ကိုယ့် script/code တွေကို ကိုယ်သုံးမထဲ design နဲ့ ကိုက်စာင် စိတ်ခြောက်ထားပါ။

ဒါ program လေးတစ်ခုလောက်ရေးကြေည့်ရှာရအောင်။ ရေးရမှာက

- User ဆီက input တစ်ခုတောင်းမထဲ
- ရလာတဲ့ input ကို result = 50/user_input ဆိတဲ့ formula ထည့်ပြီးသုံးမထဲ
- User မဲ့ quit လို့ မပြောမရှင်း program မဲ့ run နေရမယ်။
- User ထည့်လိုက်တဲ့ input ကြောင့်ဖြစ်လာတဲ့ error ကို handle လုပ်ရမယ်။ program crash ဖြစ်စေအောင်။

အိုကေ စမ်းကြေည့်ရှာပါ။ မရမှ အောက်က ကုတ်ကို ပြောလိုပါ။

```

1 import sys
2
3 print("\nProgram will quit if you enter quit!!\n")
4 while (True):
5     Num1 = input("Enter number : ")
6     try:
7         if(Num1 == 'quit'):
8             break
9         result = 50/int(Num1)
10        print ("The result is " + str(result))
11    except:
12        print ("\nError Format 1")
13        print("Oops!",sys.exc_info()[0],"occured.")
14
15    print("\n BYE!! ")
16

```

Line no.1 => sys module ကို import လုပ်ပါ။

Line no.3 => user ကို quit လိုရှိရမှ program သည် ထွက်မှာ ဖြစ်ကြောင်း print ထုတ်ပြီး ပြောပါတယ်။

Line no.4 => while loop စြေး ပါတ်ပါတယ်။ user မဲ့ quit လို့ မပြောမရှင်း အောင် ပါတ်မှာမို့ infinite loop ဖြစ်အောင် while နောက်က condition ကို boolean true ထားထားပါတယ်။

Line no.5 => User input တစ်ခုတောင်းလိုက်ပါတယ်။

Line no.6 => try block စပါတယ်။ တစ်ကယ် run မထဲ့ code တွေထည့်ပါ။

Line no.7 => user ပေးတဲ့ input သည် quit နဲ့ ညီ မညီ စစ်ပါတယ်။ မညီရင် line no.9 ကို ရော်သွားမှာ ပုံပြီး ညီခဲ့ရင်တော့ line no.8 အရ ထွက်သွားပါလိမ့်မယ်။

Line no.8 => Break statement ဖြစ်တာကြောင့် အောင် ကနေထွက်သွားပါလိမ့်မယ်။

Line no.9 => 50 ကို user ဆီကရတဲ့ input နဲ့ တားပါတယ်။ ရလာမထဲ့ အဖြေကို result ဆိတဲ့ variable ထဲ ကို ထည့်ပါတယ်။

Line no.10 => The result is ဆိုပြီးတော့ ရလာတဲ့ အဖြေကို ထုတ်ပြပါတယ်။

Line no.11 => except block စပါတယ်။ try block က ကုတ်မှားခဲ့ရင် ဖြစ်လာမယ့် exception ကို handle လုပ်ဖို့။

Line no.12 => Error ထုတ်မယ့်အကြောင်း print လုပ်ပါတယ်။

Line no.13 => sys.exc_info() ကို သုံးပြီးတော့ index[0] ထဲက exception information ကို print ထိပြုပါတယ်။

Line no.15 => user ၏ quit လိုပေးလိုက်ရင် break ကြောင့် while loop ကနေထွက်သွားရင် ထွက်သွားကြောင်းသိအောင် while loop ပြီးတော့ အောက်မှာ BYE ဆိုပြီး print ပြထားတာပါ။

Program လေးကို run ပြီး မတူညီတဲ့ input ထွေ ထည့်ကြည့်ပါပြီး.. 1, 30 တို့ထည့်ရင် ဘယ်လို အလုပ်လုပ်လဲ .. 0 တို့ 'a' တို့ ထည့်ရင်ကော့ ဘယ်လို output ထွက်လဲ.. quit လိုရိုက်လိုက်ရင်ကော့ တစ်ကယ် while loop ကနေထွက်ပြီး program end လား စမ်းကြေည့်ပါ။

```
Program will quit if you enter quit!!

Enter number : 1
The result is 50.0
Enter number : 30
The result is 1.6666666666666667
Enter number : 0
Error Format 1
Oops! <class 'ZeroDivisionError'> occurred.
Enter number : a
Error Format 1
Oops! <class 'ValueError'> occurred.
Enter number : 20
The result is 2.5
Enter number : quit
BYE!! ↗

Process returned 0 (0x0)      execution time : 36.240 s
Press any key to continue . . .
```

အချင့် မြတ်ဆောင်ရွက်ခဲ့ပါတယ်။ ဒါကိုယ်အသာဝဲ user ကို message ပေးလဲ ရတယ်နော်။ စမ်းမကြည့်ဖို့မှာ ဆိုးလို့ :-)



```
1 import sys
2
3 print("\nProgram will quit if you enter quit!!\n")
4 while (True):
5     Num1 = input("Enter number : ")
6     try:
7         if(Num1 == 'quit'):
8             break
9         result = 50/int(Num1)
10        print ("The result is " + str(result))
11    except:
12        print ("Please enter Number only except 0 ")
13
14 print("\n BYE!! ")
```

The screenshot shows a terminal window titled 'C:\Python27\python.exe' running on Windows. It displays the execution of a script that performs integer division. The user inputs various numbers, including 1, 3, 0, and 6, and receives corresponding results (50.0, 16.66666666666668, an error message 'Please enter Number only except 0', and 8.33333333333334). When 0 is entered, the program prints an error message and exits with a status of 0.

```
Program will quit if you enter quit!!
Enter number : 1
The result is 50.0
Enter number : 3
The result is 16.66666666666668
Enter number : 0
Please enter Number only except 0
Enter number : 6
The result is 8.33333333333334
Enter number : quit
BYE!!
Process returned 0 <0x0>      execution
Press any key to continue . . .
```

Error ကိုနောက်တစ်မျိုး ထပ်ထပ်လိုပါသေးတယ်။ ရွှေ့နှံတို့တွေ python interactive mode မှာ run လိုက်လို့ error တက်ရင် ထွက်တဲ့ traceback info လိုမျိုးရချင်ရင်လဲ လုပ်လိုပါတယ်။

```
>>> 1/0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero
>>>
```

အဲလိုထွက်နိုး အတွက်ကတွော traceback module ကို import လုပ်ရမယ် ပြီးတော့ traceback.format_exc() ကို သုံးပြီးတော့ sys.exc_info() လိုမျိုး restule ကို print ထုတ်ပေးပါမယ်။



```
1 import sys
2 import traceback
3
4 print("\nProgram will quit if you enter quit!!\n")
5 while (True):
6     Num1 = input("Enter number : ")
7     try:
8         if(Num1 == 'quit'):
9             break
10        result = 50/int(Num1)
11        print ("The result is " + str(result))
12    except:
13        print ("\nError Format 1")
14        print("Oops!",sys.exc_info(),"occured.")
15        var = traceback.format_exc()
16        print ("\nError Format 2")
17        print (var)
18
19    print("\n BYE!! ")
20
```

Zero input ပေးလိုက်ရင် အောက်ပါအတိုင်း result ရပါတယ်။ format နှစ်မျိုးတို့ ကွဲပြား မြင်အောင် error format 1, error format 2 ဆိုပြီး ခွဲထုတ်ထားပါတယ်။

```
Enter number : 0
Error Format 1
Oops! <<class 'ZeroDivisionError'>, ZeroDivisionError('division by zero'), <traceback object at 0x007E12B0>> occurred.

Error Format 2
Traceback (most recent call last):
  File "C:\Users\ekhaphy\Documents\atom\Python\kptest.py", line 10, in <module>
    result = 50/int(Num1)
ZeroDivisionError: division by zero
```

A ကို input အနေနဲ့ ထည့်လိုက်ချိန်မှာတွော အောက်ပါအတိုင်း result ရပါတယ်။

```
Enter number : a
Error Format 1
Oops! <<class 'ValueError'>, ValueError('invalid literal for int() with base 10: 'a''), <traceback object at 0x007E12B0>> occurred.

Error Format 2
Traceback (most recent call last):
  File "C:\Users\ekhaphy\Documents\atom\Python\kptest.py", line 10, in <module>
    result = 50/int(Num1)
ValueError: invalid literal for int() with base 10: 'a'
```



နောက်ထပ် တစ်ခုထပ်မြို့း သိပေါ်ရပ်တာက except block တွေကို တစ်ခုမက ရေးလိုရတယ်ဆိုတာကိုပါ။ အောက်မှာ နမူနာအနေဖြင့်.. အပေါ်က example ကိုပဲ ပြင်ရေးပေးထားပါတယ်။

```

1 import sys
2 import traceback
3
4 print("\nProgram will quit if you enter quit!!\n")
5 while (True):
6     Num1 = input("Enter number : ")
7     try:
8         if(Num1 == 'quit'):
9             break
10        result = 50/int(Num1)
11        print ("The result is " + str(result))
12    except ZeroDivisionError as ze:
13        print (ze)
14        print("Enter number only except 0!!")
15    except ValueError as ve:
16        print (ve)
17        print (str(Num1) + " is not acceptable value!!")
18    except:
19        print ("Error!!")
20
21 print("\n BYE!! ")
22

```

ZeroDivisionError တစ်မှာ သေချာလို့ ze အငောက့် exception ကို ဖော်ထားပြီးတော့ print ထုတ်ပြန်တယ်။ နောက်တစ်ခုပြားမှာတော့ 0 တလဲရင် ပြိုတဲ့ တို့ ထည့်လိုရအောင် ပြုပြားမှုံးမြှုပ်နှံတယ်။

ValueError ပဲတစ်နိုင်တာလို့ ve အငောက့် exception ကို ဖော်ထားပြီးတော့ print ထုတ်ပြန်တယ်။ ဒုံးနောက် အဲလို value တွေဆိုရင် acceptable မပြစ်ပြုပြားမှုံးမြှုပ်နှံတယ်။

နောက်ဆုံး except blockကျတော့ မဖော်လဲဘူး။ တဲ့ error တွေတစ်ခုခဲ့ exception ကို handle လုပ်နိုင်အောင် ထည့်ပေးထားတာပါ။

result ကိုတွော အောက်ပါအတိုင်း မြင်တွေ့နိုင်ပါတယ်။

```

Program will quit if you enter quit!!
Enter number : 0
division by zero
Enter number only except 0!!
Enter number : 3
The result is 16.666666666666668
Enter number : a
invalid literal for int() with base 10: 'a'
a is not acceptable value!!
Enter number : 3
The result is 16.666666666666668
Enter number : quit
BYE!!

Process returned 0 (0x0)      execution time : 15.192 s
Press any key to continue . . .

```

ဟုတ်မြို့း ဒီလောကဆိုရင်တွေ့ try...except ကိုတွော နားလည်သဘောပေါက်လောက်မြှုပြုလို့ ယူဆပါတယ်။ နောက်တစ်ခု ဆက်ရှုရှု့း။

8.2 Python Try.....except.....else

```

try:
    # Runs first
    < code >
except:
    # Runs if exception occurs in try block
    < code >
else:
    # Executes if try block *succeeds*
    < code>

```

Else: block သည် try block ထဲကုတ်တွေ ERROR မတက်မှ အလုပ်လုပ်တာပါ။ စမ်းကြေည့်ရာ ရင်အင်များ၏ အခင် if....else တိန်းက beer ပယ်ဖို့ အသက်စစ်ထဲ program လေးကို မှတ်မိမယ် ထင်ပါတယ်။

```

1 while True:
2     age=int(input('Enter your age: '))
3     if age <= 17:
4         print('You are not allowed to buy beer, you are too young.')
5     else:
6         print('Welcome, you are old enough. You can buy beer as you want')

```

User သည် မေးတဲ့ အတိုင်း မရှိက်ပဲ အတိုင်း twenty-two တို့ ဘာတို့ ရော်ရှိက်ရင် error တက်ပြီးတွေ့ exception object ထွေမှာပါ။ အဲတောက် handle မလုပ်ခဲ့ရင် traceback message ပြီး program သည် crash ဖြစ်သွားမှာပါ။ အောက်က result ကို ပြောပါ။

```

Enter your age: 15
You are not allowed to buy beer, you are too young.
Enter your age: 18
Welcome, you are old enough. You can buy beer as you want
Enter your age: 20
Welcome, you are old enough. You can buy beer as you want
Enter your age: twenty
Traceback (most recent call last):
  File "C:\Users\ekhaphy\Documents\atom\Python\kptest.py", line 2, in <module>
    age=int<input('Enter your age: ')>
ValueError: invalid literal for int() with base 10: 'twenty'

Process returned 1 (0x1)      execution time : 47.304 s
Press any key to continue . .

```

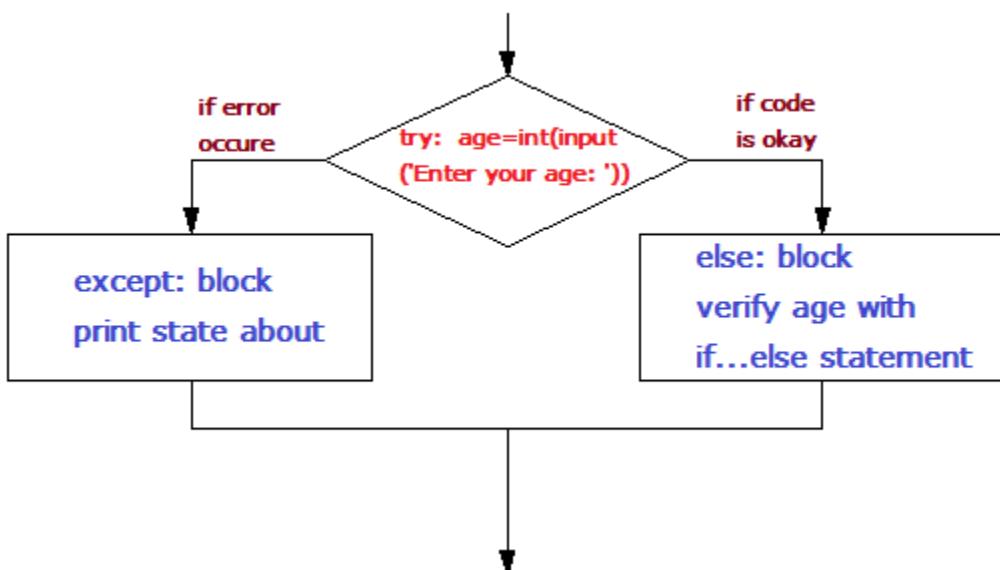
အဲတွေ့ ဘယ်လို လုပ်ရမလဲဆိုတွေ..try...except...else ကို သုံးကြေည့်ပါ။ ပထမဆုံး age ဆိုပြီး input ယူတဲ့အချိန်မှာ 1,2, စသည်ဖြင့် integer တန်ဖိုးမဟုတ်ပဲ a,b,c တို့ဖြစ်ခဲ့ရင် int() ကို သုံးလိုက်တဲ့အချိန်မှာ error တက်ပါတယ်။ အဲတွေ့ try block ထဲ မှာ error တက်နိုင်တဲ့ input ယူတဲ့ code ရေးလိုက်ပါတယ်။



ထို့။ except block မှာရွှေတွာ print ('You have entered an invalid value.') ဆိုပြီးတွာ integer ၏ နှုန်းများ မဟုတ်တဲ့ ကောင်တွေရှိရင် လက်မခံတဲ့ အကြောင်း print ထုတ်ပြပါတယ်။ else block မှာရွှေတွာမှ ဝင်လာတဲ့ input ကိုဖြေပြီး beer ထုတ်ခွဲရ မရ ကို if...else နဲ့ စစ်ပြထားတာပါ။

ဒေါ်အတွက် try သည် input ကို တောင်းပါမယ်။

Try block မှာ error တိုက်ရင် except block ကို သွာမှာဖြစ်ပြီး error မတက်ရင်တွာ else block ကို ရောက်သွားမှာပဲ ဖြစ်ပါတယ်။



အောက်က ကုတ်လေးကို run ပြီး စမ်းကြော်ရင်တွာ အလုပ် လုပ်ပဲကို ပိုပြီး နားလည်လာလိမ့်မယ်လို့ ငြောလင့်ပါတယ်။

```

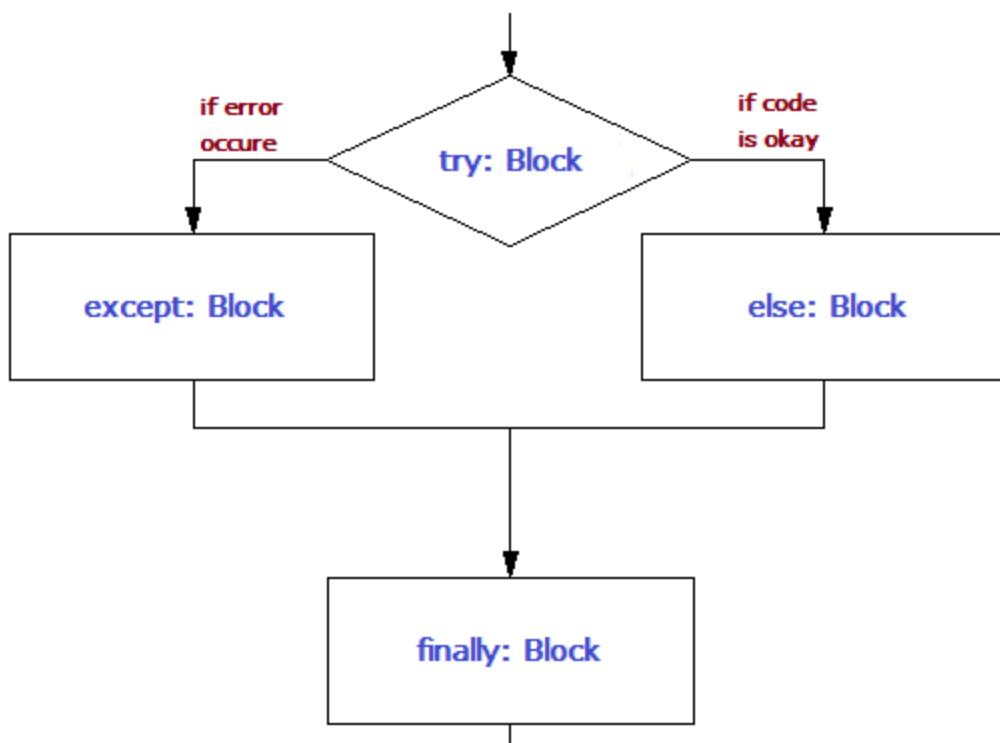
1 while True:
2     try:
3         age=int(input('Enter your age: '))
4     except:
5         print ('You have entered an invalid value.')
6     else:
7         if age <= 17:
8             print('You are not allowed to buy beer, you are too young.')
9         else:
10            print('Welcome, you are old enough. You can buy beer as you want')
  
```



User ကနေပြီးတွေ့ input အမှား ရိုက်မိရင်လဲ program သည် exception ကို handle လုပ်ထားတဲ့ အတွက် crash ဖြစ်မသွားတွေ့ပါဘူး။ အဖြောက်က result မှာ ဗြာည့်ပါ။

```
Enter your age: 15
You are not allowed to buy beer, you are too young.
Enter your age: 18
Welcome, you are old enough. You can buy beer as you want
Enter your age: 20
Welcome, you are old enough. You can buy beer as you want
Enter your age: twenty
You have entered an invalid value.
Enter your age: 25
Welcome, you are old enough. You can buy beer as you want
Enter your age:
```

8.3 Python try.....except.....else.....Finally



တစ်ခုမှတ်ရမှာက finally: block မှာရှိတဲ့ code ကို အမြတန် execute လုပ်ပါတယ်။ try:Block မှာ ရှိတဲ့ code က မှားရင် (error တက်ရင်) except: block ကို ရောက်မှာ ဖြစ်ပြီး error မတက်ရင်တွေ့ else: block ကို ရောက်ပါမယ်။ except ဖြစ်ဖစ် else ဖြစ်ဖစ် ကနေ run ပြီးရင်တွေ့ finally: block က ကိုတွောက် အပေါ်က ပုံမှ ဖော်ပြခဲ့သလိုပဲ အမြဲ execute လုပ်ကို လုပ်ပါတယ်။

Finally: Block အလုပ်လုပ်ဖုံးလေးကို အောက်က ကုတ်တွေကနေ တစ်ဆင့်လေးလာ ဤညွှန်ရအောင်များ .. ပါတဲ့ code(function, method, module) တွေကတော့ သိပြီးသွားပါ။ စာမကြောသေးဘူးထင်ရင် အကျဉ်းက အခန်းတွေကို ပြန်ဖတ်ပါ။

ခုဒီကုတ်လေးရဲ့ ရည်ရွယ်ချက်ကတော့ try: Block မှာ user ဆီကနေ အချက်အလက်တွေ တောင်းမ မေးတော့ အချက်အလက် data တွေကို username + date_month ဖိုင်နာမည့်နဲ့.. Save မယ်ပေါ့။ အဲဒေါက် input ပေးတာမှားရင် error တက်ရင် အတွက်က except: Block.. Try: block က ကောင်တွေ အကုန်မှန်ရင် else: block မှာရာမှ file ထဲကို data ထည့်မယ်။ finally: Block ကတော့ ဖိုင်ကို ဖွင့်ဆောင်ရင် ပြန်ပိတ်ပေးစိုးလိုပါတယ်။ else နဲ့ except မှာ ထည့်တာထက် finally သည် သူမှရှိတဲ့ ကုတ်ကို အမြဲ run တဲ့ အတွက် ကြောင့်ပါ။

```
try:
    # Runs first
    < code >
except:
    # Runs if exception occurs in try block
    < code >
else:
    # Executes if try block *succeeds*
    < code>
finally:
    # This code *always* executes
    < code >
```

ခု ဒီကုတ်တွေသည် finally: block ရဲ့ .. အသုံးကို မြင်သာလို့ ရေးထားပေးတဲ့ example တစ်ခုပဲ ဖြစ်ပါတယ်။ လေးလာ့ ဤညွှန်ရပါ။

```
Line no.1 => import sys
Line no.2 => from datetime import datetime
Line no.3 => try:
Line no.4=>     now = datetime.now()
Line no.5 =>     thistime = now.strftime("%d_%b_%Y")
Line no.6 =>     name = input("what is your name : ")
```

```

Line no.7 =>     filename = name + '_' + thistime + ".txt"
Line no.8 =>     myfile_o = open(filename, "w")
Line no.9=>     age = input("Enter your age? ")
Line no.10=>    location = input("Where do you live : ")
Line no.11=>    gender = input("Male/Female? ")
Line no.12=>    hourly_income = int(input("What is your basic salary income? (Hourly) "))
Line no.13=>    WorkingTime = int(input("What is total walking time of this month? "))
Line no.14=>    TotalSalary = str(hourly_income * WorkingTime)
Line no.15=> except:
Line no.16=>     print(sys.exc_info())
Line no.17=> else:
Line no.18=>     myfile_o.write("Name : " + name)
Line no.19=>     myfile_o.write("\nAge : " + age)
Line no.20=>     myfile_o.write("\nlocation : " + location)
Line no.21=>     myfile_o.write("\ngender : " + gender)
Line no.22=>     myfile_o.write("\nTotalSalary : " + TotalSalary)
Line no.23=>     print ("*****")
Line no.24=>     print ("Your data is successfully saved in " + filename + ".txt")
Line no.25=>     print ("*****")
Line no.26=> finally:
Line no.27=>     myfile_o.close()
Line no.28=> #####

```

မြတ်ရန်။ အရင်ဆုံး error တက်ရင် exception ကို handleလုပ်နိုင် မလုပ်နိုင် စမ်းကြည့်ရှု ရအောင်။ data ကို အမှားလေးတွေထည့် ကြည့်ရှုရအောင်။

```

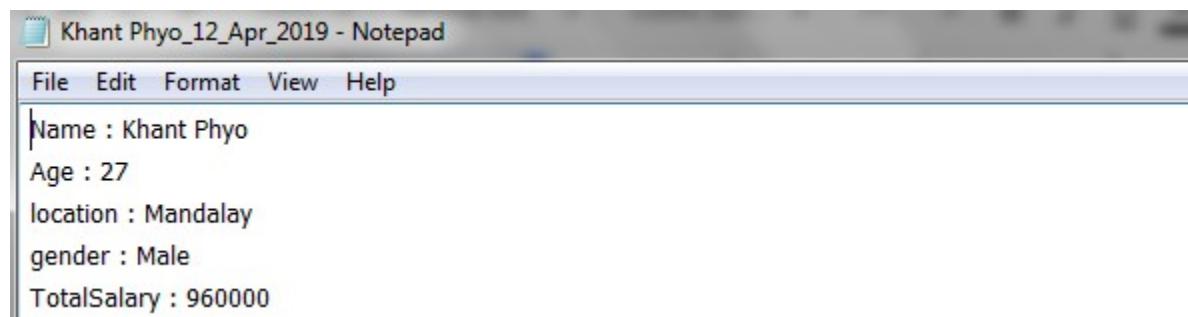
what is your name : Khant
Enter your age? 27
Where do you live : MDY
Male/Female? Male
What is your basic salary income? (Hourly) i don't know
<<class 'ValueError', ValueError('invalid literal for int() with base 10: "i do
n't know"'), <traceback object at 0x01F92148>
Process returned 0 (0x0)      execution time : 20.588 s
Press any key to continue . . .

```

Line no.12 အရ ဝင်လာမည့် input သည် integer ဖြစ်မှ အဆင်ပြေမှပါ။ character string ကို integer ပြောင်းလို အဆင်ပြောပါဘူး။ error တင်တဲ့အတွက် except: block ကို ရောက်ပြီးတော့ ValueError ဆိတဲ့ result ထုတ်လာပါတော့တယ်။

```
what is your name : Khant Phyoe
Enter your age? 27
Where do you live : Mandalay
Male/Female? Male
What is your basic salary income? <Hourly> 6000
What is total working time of this month? 160
*****
Your data is successfully save in Khant Phyoe_12_Apr_2019.txt
*****
Process returned 0 (0x0)      execution time : 16.285 s
Press any key to continue . . .
```

Run လိုက်တဲ့အမျိန်ရာ user ဆီက input တွေ ယူပြီး save လိုက်ပြီး ဆိတဲ့ result ကို တွေ့ရပါမယ်။ မြတ်မြတ် save ထားတဲ့ ဖိုင်လေးကို သွားဖွင့်ကြည့်ရှုရအောင်။



ဒီလောက်ဆိုရင်တွေ့wa Exception Handling ကို အတ်ဘလေး သဘောပေါက်ပြီး ထင်ပါတယ်။ နောက်တစ်ခန်း ဆက်ရှုပါစို့။

9. Python - Function

ဂျွှန်တော်တို့ တွေ့ အကြိမ် ကြိမ် လုပ်ရမယ့် အလုပ်တွေကို ဟိုးအရှေ့မှာတုန်းက 1oop ပါတ်ပြီး ရေးခဲ့ရာတယ်လေ.. အဲလိုမျိုးပဲ.. အကြိမ်ကြိမ် run ရမယ့် ကုတ်တွေကို စုပြီး Function တွေခဲ့ရေးရာပါမယ်။ ဘာလို့ လဲဆိုတွေ့wa code ရေးရတာ မျက်စိရင်းမယ် သူ့ သက်ဆိုင်ရာ အစုလေးနဲ့ သူရှိတွေ့wa tshoot လုပ်ရတာ error trace လိုက်ရတာလွယ်တယ်။ ပြီးတွေ့wa function ကို ခေါ်လိုက်ယုံးနဲ့ function မှာ define လုပ်ထားတဲ့ code တွေက အလုပ် လုပ်သွားမှာဖြစ်လို့ .. လူသက်သာတယ်။ စသည်ဖြင့်ပေါ်နော.. ကုတ်ရေးရင်း လက်တွေ့လှေလာကြည့်ရှုရအောင်။

ဒဲ ဂျွှန်တော်တို့ တွေ့ လက်ရှိသုံးနေတဲ့ print() တို့ input() တို့ သည် python မှာ Built-in ပါနေတဲ့ function တွေပဲ ဖြစ်ပါတယ်။ အဲ function အပြင် ကိုယ့်စိတ်ကြိုက် ကုတ်တွေနဲ့ ကိုယ်အသာ function တွေ ဖန်တီးတာကို user-defined function လို့ ခေါ်တာပါ။

9.1 User Defined Function

Function တစ်ခုကို define လုပ်တွေ့မယ်ဆိုရင် ...

- def ဆိုတဲ့ keyword နောက်မှာ ကိုယ်လုပ်မယ့် function နာမည်ရေးရမယ်။ () ပါရမယ်။
- အဲ ကွင်းစ ကွင်းပိတ်ထဲမှာ .. function အတွက် ထည့်ချင်တဲ့ parameter/argument ကို ထည့်လိုရ တယ်။
- : (full column)ရေးပြီး နောက်တစ်ကြောင်းဆင်းရင် indentation level ကို တူအောင်ထားပါ။ tab တစ်ခါခုနှင့်လဲ function block ထဲမှာ ရှိမယ့် code တွေကုန်သည် indentation level တူအောင် tab တစ်ခါ ခုနှင့်စာဆီ ခြားရမှာပါ။
- function ရဲ့ပထမဆုံး စာကြောင်းသည် optional ပါ။ ပြောချင်တာက ဒီ function သည် ဘာကို လုပ် မှုးfunction ဖြစ်တယ်။ ဒါမှာဟုတ် တစ်ခုခုပဲ့ function နဲ့ပါတ်သက်တဲ့ document လို ရေးတာ ပဲ ဖြစ်ဖြစ် စသည်ဖြင့် ရေးလို့ ရပါတယ်။
- return ပြန်တဲ့အခါက္ခ function သည် ထွက်သွားပါတယ်။ return ပဲ ရေးထားတယ် နောက်က ဘာ မပါဘူးဆိုရင် return null ပြန်ပါတယ်။

ဣဣဣ ဣဣဣ ဣ ဣ user-defined function အတွက် syntax ပုံစံကို ရေးပြပေးထားတာပါ။

```
def function_name(parameters):
    """Docstring"""
    statement(s)
    return
```

Function လေး တစ်ခု လောက် တည်ဆောက်ပြုလိုက်ရအောင်။

<pre>1 def printHW(): 2 "This prints a passed string into this function" 3 print ("Hello, World!") 4 5 printHW()</pre>	<p>Function Definition</p>
	<p>Function Call</p>

def printHW() ဆိုပြီးတော့ printHW ဆိုတဲ့ function ကို defined လုပ်လိုက်ပါတယ်။ အဲ function က ဘာကို လုပ်ဆောင်ပေးမှာလဲ ဆိုတော့ print("Hello, world!") ဆိုပြီး Hello, world ထုတ်ပေး ထို့ပါ။ ခုက function printHW() ကို python က သိအောင်ပဲ defined လုပ်ထားတာပါ။ ခေါ် ပြီး မသုံးမချင်း အလုပ် မလုပ် မပါဘူး.. line no.5 မှာ printHW() ဆိုပြီး function ကို ခေါ်လိုက်မှ အလုပ်စလုပ်ပြီး hello, world ကို print ထုတ်ပေးတာပါ။

9.2. Python Function Argument



User-defined Function ဆိတဲ့အတိုင်း ကိုယ်စိတ်ကြိုက် ကိုယ်ရေးမယ် program နဲ့ ကိုက်အောင် ဖန်တီးတာဖြစ်လို့.. Function ချဉ်းသက်သူရှိမယ်။ ဒါမှမဟုတ် function နောက် () မှာ data ပို့ပို့မြှုပ်နည်းတွေ သက်မှတ်ပေးတာမျိုး တွေ ရှိပါမယ်။ argument ထည့်တဲ့ ပုံစံပေါ်မှတ်ည်ပြီး အမျိုးအစား ငဲးတွေ ကွဲ ပါတယ်။ အောက်မှာ လေ့လာပြောလိုက်ရာရအောင် မှာ။

9.2.1 Required arguments

function မှာ data ငဲးတွေ ထပ်ထည့်ပြုလိုက်ရအောင်။ ခု ဒီ ဥပမာများတွေ printHW() မှာ var1 ဆိတဲ့ input parameter လက်ခံမထဲ့ နေရာတစ်ခု ထည့်ထားပါတယ်။ အဲအတွက် function ကို လုမ်းချေတဲ့အနီးနဲ့ ချို့ရပါမယ်။ line no.5 မှာ printHW("Hello") လို ရေးထားတဲ့ အတွက် printHW() function ကို ချို့တဲ့နေရာမှာ var1 = Hello ကို function ဆီ ပို့လွှတ်လိုက်ပါတယ်။ print(var1 + ", world") ဖြစ်တဲ့ အတွက် Hello, world ဆိတဲ့ result ထွက်လာတာပါ။ line no.6 printHW("Hi") အတွက်များ Hi, world ဆိတဲ့ result ထွက်လာပါလိမ့်မယ်။

```

1 def printHW(var1):
2     "This prints a passed string into this function"
3     print (var1 + ", World!")
4
5 printHW("Hello")
6 printHW("Hi")
7

```

ဒီတစ်ခါဂျာတွေ function မှာက return ပြန်ရင် function ကနေ ထွက်သွားတယ် ဆိတာကို လက်တွေ့သိအောင် စမ်းရေးပြောလိုပါ။ အောက်က ကုတ်မှာဆိုရင် return ရဲ့ပေါ်မှာကော အောက်မှာကော print ထုတ်ထားပါတယ်။ line no.3 ကို print ထုတ်ပြီး line no.4 ၏ return ဖြစ်တာမို့ function ၏၏၏ ထွက်သွားပြီးဖြစ်လို့ line no.5 ၏ print သည် ထွက်မလာတော့ပါဘူး... လက်တွေ့စမ်းရေးပြောလိုပါ။

```

1 def printme(strr):
2     "This prints a passed string into this function"
3     print("before return")
4     return ←
5     print (strr)
6
7 printme("Hello")

```

Function ဆွဲရဲ့.. First line မှာရေးထားတဲ့ function document ကို ဖော်ချင်တယ် ဆိုရင်တွေ့ FunctionName.__doc__ ပဲ ဖြစ်ပါတယ်။

```
>>> def printme(str):
...     "This prints a passed string into this function"
...     return
...     print("before return")
...     return
...     print (str)
...
>>> printme("Hello")
before return
>>> printme.__doc__
'This prints a passed string into this function'
>>> |
```

နောက်ထပ် example တစ်ခု နဲ့ ... ထပ်ပြီး လွှေလာဌာည့်ပါ။ ခု အောက်က code တွေမှာ ဆို function အတွက် argument နှစ်ခု လိုပါတယ်။ **fucntion** ကို ခေါ်တဲ့အခါမှာ data တန်ဖိုး နှစ်ခုနဲ့ပဲ.. အစီအစဉ် တရာ့ ခေါ်မှု ရမှာပါ။

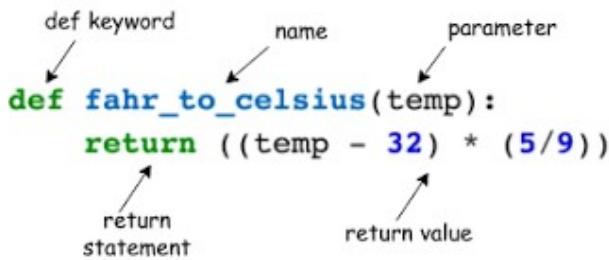
```
1 def greet(name,msg):
2     """This function greets to the person with the provided message"""
3     print("Hello",name + ', ' + msg)
4
5 greet("Monica","Good morning!")
```

ခုတိုင်းဆိုရင်တွော Hello Monica, Good morning! ဆိုတဲ့ စာသားတွေကိုလာမှာပါ။ greet() ကို ခေါ်တဲ့ အချိန်မှာ variable data တန်ဖိုးက တစ်ခုပဲ ရှိနေခဲ့မယ် ဆိုရင်တွော error တက်ပါလိမ့်မယ်။

```
1 def greet(name,msg):
2     """This function greets to the person with the provided message"""
3     print("Hello",name + ', ' + msg)
4
5 greet("Monica")
```

C:\Python27\python.exe
 Traceback (most recent call last):
 File "C:\Users\ekhaphy\Documents\atom\Python\testpython.py", line 5
 e> greet("Monica")
 TypeError: greet() missing 1 required positional argument: 'msg'

အောက်မှာ နောက်ထပ် example တစ်ခုပါ run ဌာည့်ပါပါ။ function ကို ခေါ်လိုက်တဲ့ အချိန်မှာ ရလာတဲ့ data ကို တွေက်လိုက်ပြီး return ပြန်ပေးတဲ့ ပုံစံပါ။



```

1 def fahr_to_celsius(temp):
2     return ((temp - 32) * (5/9))
3
4 print(fahr_to_celsius(92))
5
  
```

A screenshot of a terminal window titled "C:\Python27\python.exe". The window displays the output of the Python code, which is the result of the function call `fahr_to_celsius(92)`, showing the value `33.33333333333336`.

9.2.2. Keyword Argument

Keyword Argument တွေသည် function call ခါးတဲ့နေရာမှာ argument လွှာတာ အစိအစဉ် မရှုဖို့တောင် မရှိအောင် လုပ်ဆောင်ပေးပါတယ်။ ဒီနေရာ မှာ တစ်ခု မှတ်ထားရမှာကတွာ **function call** လုပ်တဲ့အခါမှာ သုံးတဲ့ argument နာမည်နဲ့ function မှာ define လုပ်ခဲ့တဲ့ parameter နာမည်ကတွေ တူရပါမယ်။

```

1 def intro(name,age):
2     """This function intro to the person with name and age"""
3     print("My name is ",name + ' and I am ' + str(age) + " year old")
4
5 intro(age = 27,name = "MgMg")
  
```

အပေါ်က ကုတ်ကို execute လုပ်လိုက်တဲ့အချိန်မှာ.. အောက်ပါအတိုင်း result ကို ရပါလိမ့်မယ်။

```

My name is MgMg and I am 27 year old
Process returned 0 <0x0>      execution time : 0.234 s
Press any key to continue . . .
  
```

9.2.3. Default Argument

Default argument ဆိုတာကရာတွေ function call လုပ်လိုက်တဲ့အချိန်မှာ data value မပါလာ ခဲ့ရင် default အနေနဲ့ ထည့်သုံးမယ့် ကောင်ကို ခေါ်တာပါ။ ခုအောက်က example မှာဆို Line no.9 မှာက `age` ကော့ `name` ကော့ မှာ data ထည့်ပေးထားတဲ့ အတွက် `Name: miki Age 50` ဆိုပြီး ထွက်လာမှာပါ။ Line no.10 မှာရာတွေ `name` တစ်ခုပဲ ပါတာမို့.. Function ကို `define` လုပ်စဉ်က သတ်မှတ်ထားတဲ့ `default age` ကို ယူသုံးသွားပါတယ်။ အဲတော့ရှောင့် result သည် `Name: miki Age 35` ဆိုတဲ့ အဖြေကို ရပါတယ်။



```

1 # Function definition is here
2 def printinfo( name, age = 35 ):
3     "This prints a passed info into this function"
4     print "Name: ", name
5     print "Age ", age
6     return;
7
8 # Now you can call printinfo function
9 printinfo( age=50, name="miki" )
10 printinfo( name="miki" )
11

```

9.2.4. Python Arbitrary Arguments

ခုထိရေးလာတဲ့ function တွေမှာပါတဲ့ argument တွေသည် အရေအတွက် အတိအကျ ပါပါတယ်။ တစ်ခါတစ်လေဂျာ function မှာ ရှိနိုင်မယ့် argument တွေကို ကြိုတွက်လို့ မရတာမျိုးမှာဆိုရင် Python မှာ ဘာလုပ်လို့မျေားလိုပေးပေးတာကို parameter name ရှိ၏မှာ * လေးခံပြီးရေးပေးပေါ်တယ်။ အဲလိုရေးပေးတာကို arbitrary argument လိုခေါ်ပါတယ်။

အရေအတွက် ကြိုမသိနိုင်တဲ့ argument တွေကို handle လုပ်မယ် Python function လေး တစ်ခု ကို arbitrary argument တွေသုံးပြီးတော့ define လုပ်ကြော်ရှာ ရအောင်များ။

```

1 def greet(*names):
2     """This function greets all the person in the names tuple."""
3
4     # names is a tuple with arguments
5     for name in names:
6         print("Hello",name)
7
8 greet("Monica","Luke","Steve","John")
9

```

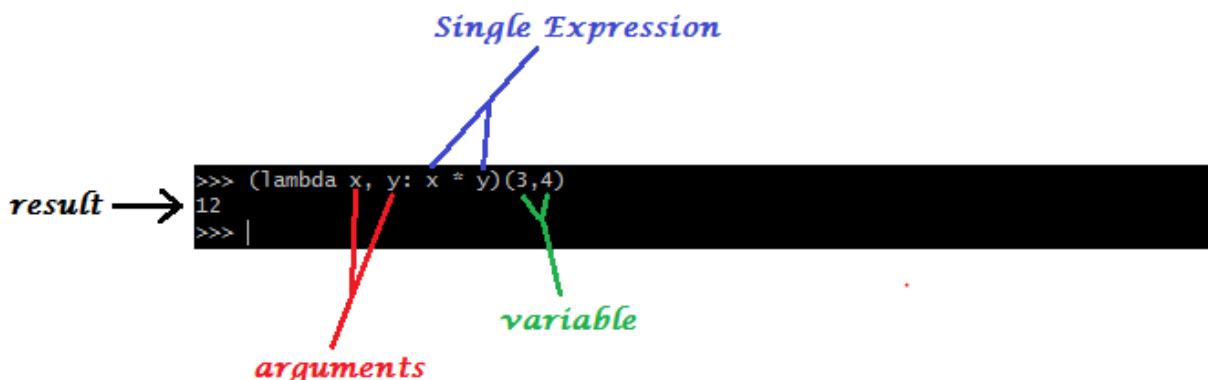
Def ကို သုံးပြီးတော့ greet function ကို တည်ဆောက်ထားပါတယ်။ () ထဲမှာတဲ့ *names ဆိုပြီးတော့ arbitrary argument ကို သုံးထားပါတယ်။ line no.8 greet("Monica","Luke","Steve","John") ဆိုပြီးက မေးမြန်မှု function ကို လုမ်းချေလိုက်တဲ့အနီးမှာ အနောက်က argument တွေသည် function ထဲ ကို tuples အနေဖြင့် ပေးပေးပါတယ်။ အဲနောက် for loop ပါတ်ပြီးတော့ tuple names ထဲကောင်တွေ မကုန်မချင်း print ထုတ်ပေးသွားတာပါ။

9.2.5. lambda functions in Python

ပုံမှန် function မျိုးမဟုတ်ပဲ.. တစ်ခါသုံး function မျိုး ပြီးတော့ expression တစ်ခု လောက်ပဲ ရှိနိုင် ။
function မျိုးဆို lambda function အနေနဲ့ သုံးလို့ ရပါတယ်။ ပုံမှန် function သုံးရင်လဲ ဖြစ်တယ်နော..
ကိုယ်စိတ်တိုင်းရဲ code ကို ချယ်လှယ်ပါ။ ရေးရင်နဲ့ အသုံးပေါင်းပုံကို မြင်လာပါလိမ့်မယ်။ စစ်ချင်း အများငြား ပြော
ရင် လည်သွားပါလိမ့်မယ်။

Lambda function ကို anonymous function လိုလဲ ခေါ်ရှာပါတယ်။ ဘာလိုလဲ ဆိုတော့ သူ့မှာက
function name မပါတဲ့ အတွက် ငြောင့်ပါ။ ပုံမှန် function ကို define လုပ်ဖို့ def [FunctionName] ကို
သုံးခဲ့ရှာပါတယ်။ anonymous function မှာရှာတော့ def ကို မသုံးပဲ lambda ဆိုတဲ့ keyword ကိုပဲ သုံးပြီး
defined လုပ်တာပါ။ Lambda မှာ argument မထည့်လဲ ရပါတယ်။ တစ်ခုထက်မက ထည့်လဲ ရပါတယ်။ ဒါပေါ်
မယ့် expression အနေနဲ့ ကတော့ တစ်ခုပဲ ရှိလို့ရပါတယ်။ lambda function မှာ variable ကို assign လု
ပ်ပေးစရာမလိုပဲ တန်းသုံးလိုလဲ ရပါတယ်။ Lambda ခဲ့ syntax ကတော့ အောက်ပါ အတိုင်းပဲ ဖြစ်ပါတယ်။

```
lambda [arg1 [,arg2,....argn]]:expression
```



ခုအပေါ်က ပုံမှာဆို lambda function ကို တန်းခေါ်သုံးလိုက်တာပါ။ အဲလိုမဟုတ်ပဲ variable
object တစ်ခု lambda function အတွက် တည်ဆောက်ပြီး သုံးလဲရပါတယ်။ traditional def function မှာ
ဆင်ပါတယ်။

```

>>> r = lambda x, y: x * y
>>> r(12, 3)    # call the lambda function
36
>>>

```

R ဆိုတဲ့ function object တစ်ခု တည်ဆောက်လိုက်ပါတယ်။ အဲနောက် r ဆိုတဲ့ object ထဲကို
argument တွေထည့်လိုက်တဲ့ အတွက် x*y ဆိုတဲ့ expression အရ result သည် 36 ရပါတယ်။ အောက်
မှာ traditional function နဲ့ anonymous function ကို နှိမ်းယူဉ်ပေးထားပါတယ်။ အရမ်း ကို ကွဲကွဲ ပြား၏

ဟာ: သိမ့်ကတွေ python ထဲကို ထဲထဲငင်ယင် ရေးထားဖြောမှ ပိုမြေး သိမြင်လာပါလိမ့်မယ်။ လေ့လာခါစမှာ အရမ်းပြီး pressure ထားပြီး မလောပါနဲ့။ ဒုတွော စမ်းကြာည့်လိုက်ပါပြီး.. ;-)

Traditional Function with 'def' keyword

```
1 def multiply(x, y):
2     return x * y
3
4 print(multiply(12,3))
5
```

It have function name

Vs

Anonymous Function with 'Lambda' keyword

```
1 r = lambda x, y: x * y
2
3 print(r(12,3))
```

It don't have function name

9.3 Python recursive function

Python ရဲ function တွေသည် function ထဲမှာ တစ်ခြား function ထပ်ခံလို့ ရပါတယ်။ အဲလို င်လိုရတယ် ဆိုတဲ့ နေရာမှာ .. သူ ကိုယ် သူလဲ ပြန်ခံနိုင်ပါတယ်။ အဲလို ကိုယ့်ကိုကိုယ် ပြန်ခံတာကို function recursive လုပ်တယ်လို့ ခံပါတယ်။

ချေအောက်က example မှာဆို သုတေသနတဲ့ factorial ရှာတဲ့ ပုံသေနည်းကို recursive function သုံးပြီးတွော ရေးထားပါတယ်။ factorial မှာ.. ရှာမယ် ကကန်းတစ်ခု ရှိတယ်.. အဲက ကိုန်းကကန်းကို တစ် နှုတ်လိုက် မြောက်လိုက်... တစ်နှုတ်လိုက် မြောက်လိုက်နဲ့.. ၁ ကို ရောက်တဲ့ အတိလျပ် သွားရမှာပါ။ ၁ကို ရောပြီး နှုတ်စရာ မရှာနိတွောဘူးဆိုမှ .. final result ကို ရပါတယ်။ ကိုန်း သည် 4 ဆိုပါစို့။ 4 * 3 * 2 * 1 ဆိုတဲ့ အတွေ့ ၂၄ ရပါတယ်။ အောက်ကုတ်တို့ run ကြေည့်ပါ။

```
1 def calc_factorial(x):
2     """This is a recursive function to find the factorial of an integer"""
3
4     if x == 1:
5         return 1
6     else:
7         return (x * calc_factorial(x-1))
8
9 num = 4
10 print("The factorial of", num, "is", calc_factorial(num))
```

Line no.9 မှာ num = 4 ဆိုပြီး ကြောသာထားပါတယ်။ line no.10 မှာ calc_factorial(num) ဆိုပြီး function ကို ခေါ်လိုက်ပါတယ်။ သူမှာ condition စစ်တဲ့ နေရာရှိပါတယ်။ if x == 1 ဖြစ်လား တစ် နဲ့ ညီ။

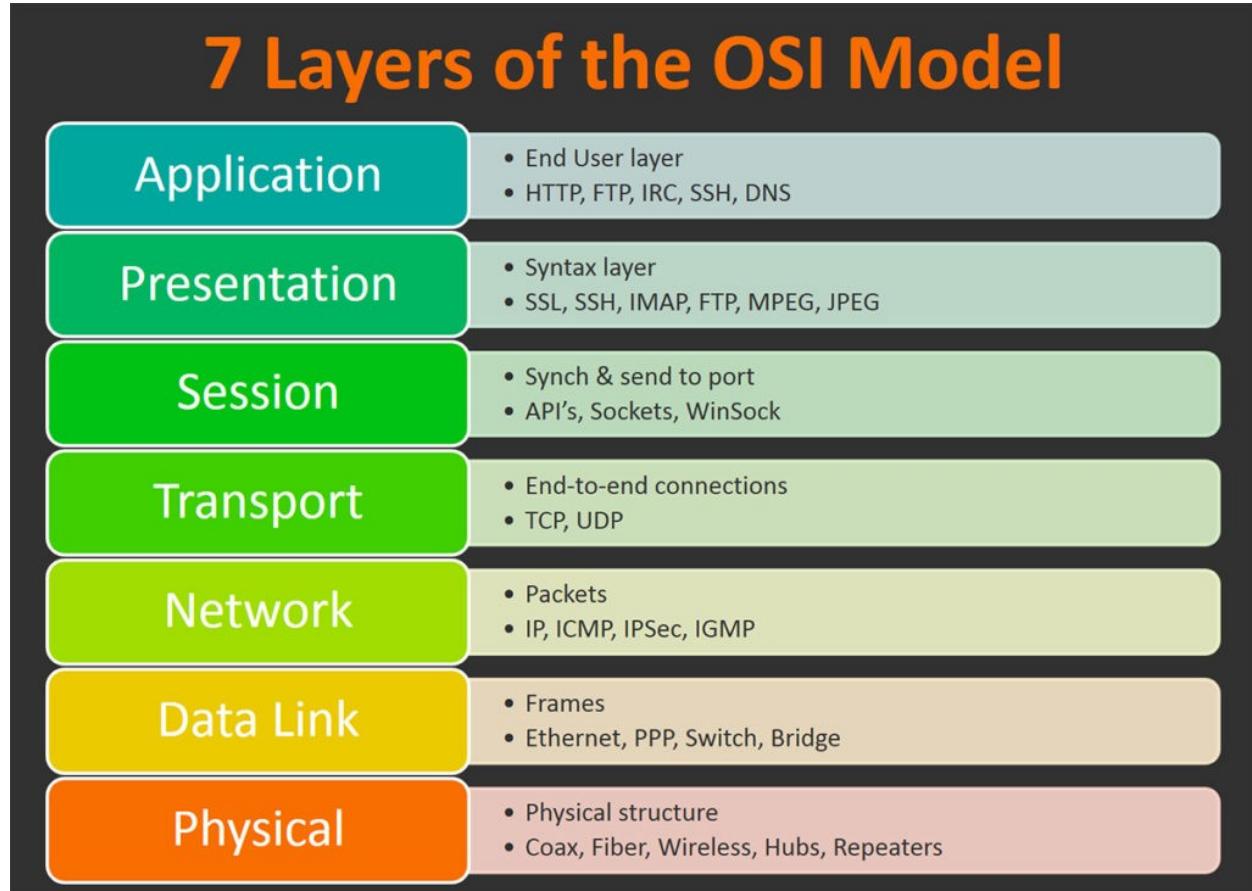
ပိုလား ဆိတာကို စစ်ပါတယ်။ ညီခဲ့ရင် return 1 ဆိုပြီး ထွက်သွားမှာပါ ဆက်မလုပ်တော့ပါဘူး.. 1 နဲ့ မည့် မချုပ်။.. else: ။ x * calc_factorial(x-1) ဆိုပြီး

```
calc_factorial(4)      # ပထမအကြိမ်မှာက 4ပါ
4 * calc_factorial(3)  # ဒုတိယ အကြိမ် ရာတွေ 1 နှင့်တဲ့အတွက် 3ပါ
4 * 3 * calc_factorial(2)  # သုံးကြိမ်မြောက်ရာလဲ 1 ထပ်နှင့်တဲ့အတွက် 2ပါ
4 * 3 * 2 * calc_factorial(1) # လေးကြိမ်မြောက်မှာတွေ ၁ ဖြစ်သွာပါပြီး
4 * 3 * 2 * 1          #ခုလိုရလာတဲ့ ကောင်တွေအကုန်မြောက်ပြီး return ပြန်ပါတယ်။
```

ရမယ်လို့ ဖြောလင့်ပါတယ်။ တစ်ခါလျပ်ကြေည့်လို့ နားမလည်ရင် step by step function call ကို ချေရေးပြီး ၆ လုလာကြေည့်ပါ။ စာလို ရေးရတာဆိုတော့ အတတ်နိုင်ဆုံး နားလည်အောင်တွော ရေးပေးထားပါတယ်။ စာသင်ခန်းလို့မဟုတ်တော့ ကောင်းကောင်းနားမလည်မှာ ဆိုးလိုပါ။

ဒီနေရာမှာ Python basic ကို တစ်ခန်း ရပ်ပြီး Network script ဘက်နဲ့ Network lab setup ဘက်လုညွှေ့ ဖူးလို့။

10. Intro OSI 7 Layer



Open Systems Interconnection model (OSI model) ကို Computer system တွေကြားမှာ error-free

communication ဖြစ်ပေါ် International Organization for Standardization (ISO) က သတ်မှတ်ခဲ့တာပါ.

ဘယ်လိုကြောင့်လဲ ဆိုတော့.. ကွန်ပူးတာ ထုတ်ထဲ company တွေ အများကြီးရှိသလို ကွန်ပူးတာတွေ Network

တွေ အပြန်အလှန်ရှိတ်ဆက်ဖို့ device ထုတ်ထဲ company တွေ ကလဲ အများကြီးပါ. မတူညီတဲ့ company တွေက

ထုတ်ထဲ နည်းပညာ မတူတဲ့ device တွေ ကွန်ပူးတာတွေ, Network တွေ တစ်ခုနဲ့ တစ်ခု အပြန်အလှန် ရွှေ့ချောမွေ့မွေ့

နဲ့ အချိတ်အဆက်မိတော့ စံသတ်မှတ်ပေးထားတဲ့ standard model တစ်ခု ရှုမှတ်ခဲ့တာပါ. ပြောရရင်တော့ မှာ..

Computer တွေ Node တွေ တစ်ခုနဲ့ တစ်ခြကြားမှာ သုံးမထုံး physical media တွေ Encrypting Type တွေ rule တွေ

application တွေ protocols တွေ encoding/decoding စတာတွေကို သတ်မှတ်ပေးထားတဲ့ Standard model

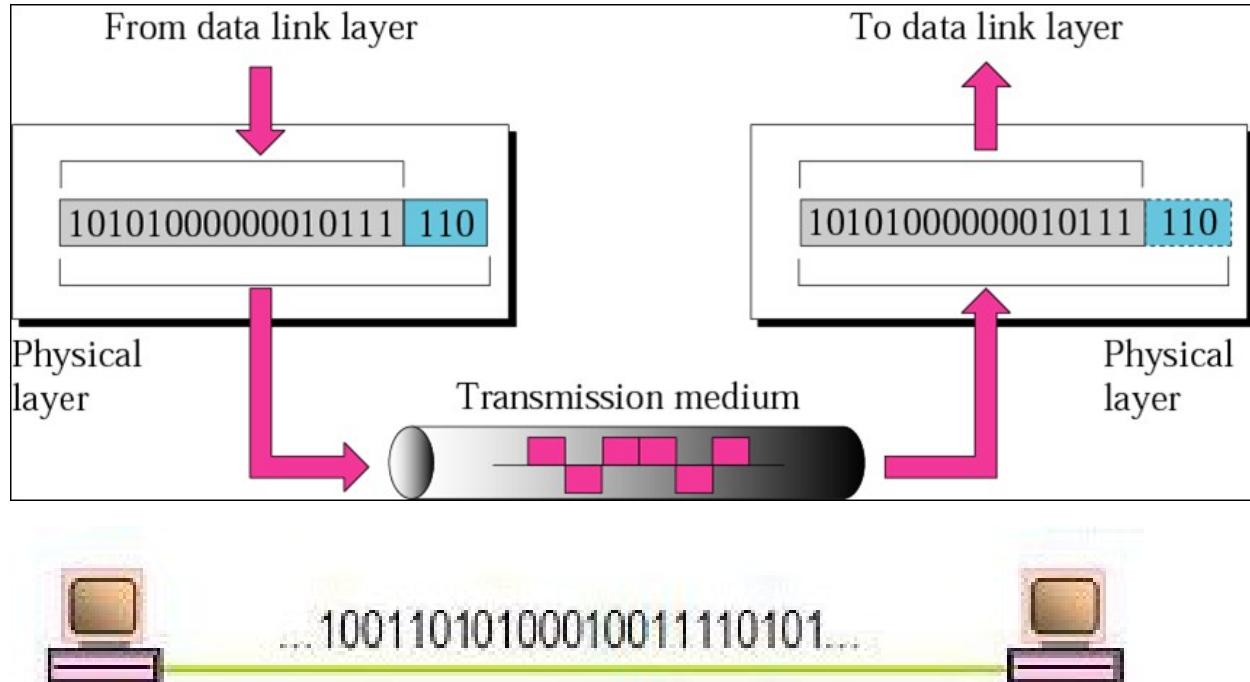
တစ်ခုဖြစ်ပါတယ်. OSI model မှာ Layer 7 ခုရှိပါတယ်.. အဲတာတွေကတွော...

10.1. Physical Layer

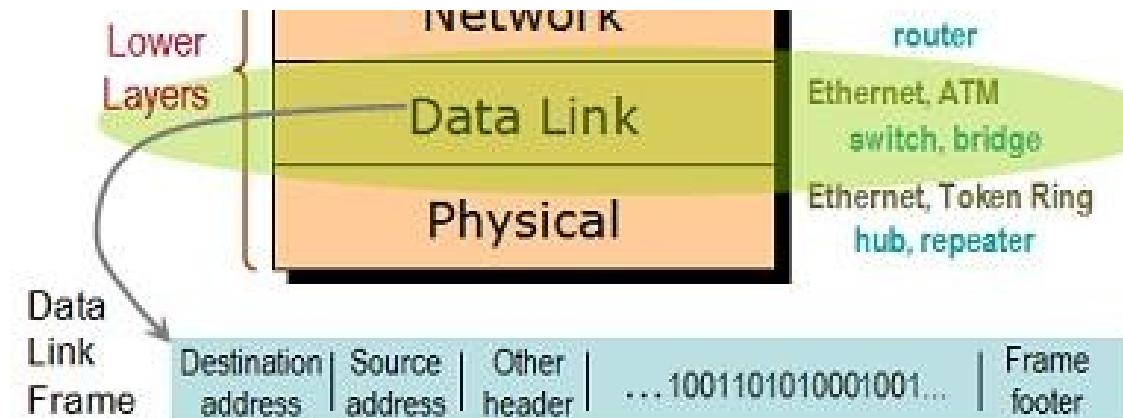
Physical Layer ဆိုတွော OSI model ရဲ့ ပထမဆုံး layer ဖြစ်ပါတယ်. သူက data တွေကို Bit အနေနဲ့ ပိုပါတယ်. သူတာကန်က pin no, volt, dB(အလင်းပြင်းအား), cable (Twisted Pair, Coaxial Cable, Optical Fiber), electrical/optical signal(on, off, 0, 1), frequency စတာတွေကို standard တစ်ခု သတ်မှတ်ပေးထားပါတယ်. ပြီးတွော signal timing ပြောရရင် Data singal တစ်ခု 0 ပို့ပြီးရင် နောက် Data Signal တစ်ခု 1 ကို ချက်ချင်း ပိုလိုက်လို့မရပါဘူး.. အချိန်အပိုင်းအမြား တစ်ခု Nano second ပိုင်းလောက်ကြာသည်အထိ စောင့်ပြီးမှတိမှ ရပါမယ်. အဲသည် Data signal တွေကြားကြာဖို့နဲ့တွေ့ကိုလဲသက်မှတ်ပေးထားရပါတယ်. Transmitter နဲ့ Receiver ကြား Signal Timing မတူရင် Timing လွှဲရင် error တက်နိုင်ပါတယ်.

ပြီးတွော wireless device တွေ အတွက်ဆိုရင်လဲ.. သုံးရမထုံး Frequency(2.4Ghz, 5Ghz,.etc) တွေ ကို တူညီမှုရပါမယ် မတူညီခဲ့ရင် communicate လုပ်လို့မရပါဘူး.. တစ်ခါ transmission mode တွေဖြစ်တဲ့ simplex, half duplex, full duple mode တွေကိုလဲ Physical Layer က define လုပ်ပါတယ် လုပ်ဆောင်တာ ကတွော Session Layer ကလုပ်ဆောင်ပေးပါတယ်. နောက်ပြီး Physical topology တွေ ဖြစ်တဲ့ Bus, Mesh တို့ ring တို့ကာလဲ Physical Layer နဲ့ သက်ဆိုင်ပါတယ်. Physical Layer ရဲ့ အဓိက တာကန်ကတွော သူအထူး Layer ဖြစ်တဲ့ DataLink Layer ကနေ ပို့သမှု data frame တွေကို bit signal(0,1) တွေ အနေနဲ့ပြောင်းလဲပြီး ပို့ဆောင်ပေးရပြီး ပင်လာသမှုကိုလဲ လက်ခံယူပြီး datalink layer ကို frame တွေအနေနဲ့ပြန်လည်ပို့ဆောင်ပေးပါတယ်.

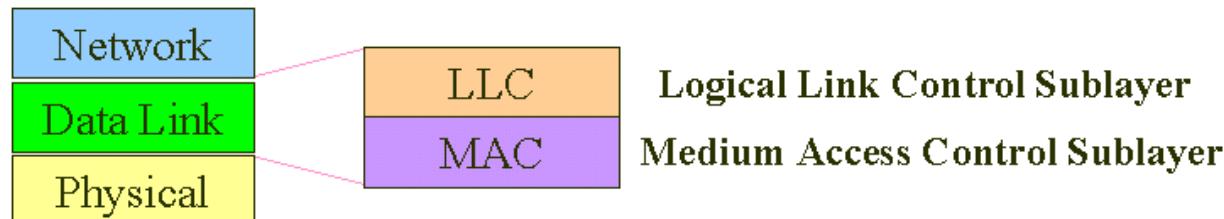
10.2. Data Link Layer



DataLink Layer ကတော် OSI မှာ ဒုတိယမြောက် Layer ဖြစ်ပြီးတော် Physical Layer နဲ့ Network Layer တို့ အကြားမှာ ရှိပါတယ်. သူက Data တွေကို Frame တစ်ခု အနေနဲ့ ပြောင်းပြီး ပို့ဆောင်ပေးပါတယ်. အမိက တာဝန်ကတော် sender နဲ့ receiver ကြားက Network မှာ ရှိတဲ့ device တွေကြား data ပိုလိုက်ရင် collision ဖြစ်မဖြစ် အရင်စစ်ပါတယ် ပြီးမှ ok ပြီ့ဆိုတော့မှ data စတင်ပိုပါတယ်. ဥပမာ Sender A နဲ့ receiver D ကြားမှာ Device B နဲ့ C ရှိတယ် ဆိုရာပါဆို. ပထမဆုံး sender A and devive B ကြားမှာ collision ဖြစ်မဖြစ် တစ်ခါ စစ်ပါတယ် clear ဖြစ်မှ data ပိုပါတယ်. တစ်ခါအဲလိုပဲ device B နဲ့ C ကြားမှာလဲ စစ်ပါတယ် clear ဖြစ်မှ data ပိုပါတယ် နောက် ဆက်စစ်ပါတယ် device C နဲ့ receiver D ကြားကို အဆင်ပြေမှ data ပိုပါတယ် ပြောရရင် နောက်ဆုံး လက်ခံမယ့်သူလဲ မရောက်မချင်းလမ်းမှာ ရှိသမှ point to point နှင့်ထားတဲ့ segment တိုင်း အပိုင်းတိုင်းကိုစစ်ပါတယ် collision ဖြစ်နိုင်မဖြစ်နိုင်ကို ပြီးမှ data ကိုပိုပါတယ်.

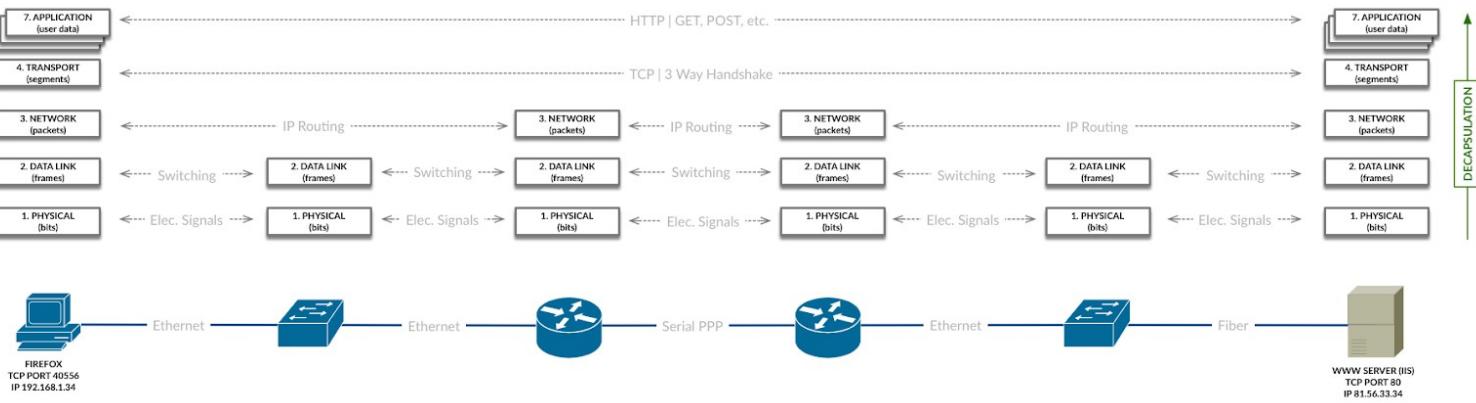


သူမှာ SubLayer 2 ခုရှိပါတယ် Media Access Control (MAC) Sublayer နဲ့ Logical Link Control (LLC) Sublayer တို့ ဖြစ်ပါတယ်. Logical Link Control(LLC) sublayer ကတော့ point to point Node က လေးတွေကြားမှာ logical connection တစ်ခု အရင် ဖန်တီးပေးပါတယ် ပြီတော့ data တွေ အဲ connection Link က လေးပေါ်ကသွားနိုင်အောင် အသင့်ပြင်ပေးပါတယ်. Media Access Control Sublayer မှာ.. MAC address(Layer 2 address, Physical address လို့လဲ ခေါ်ပါတယ်) ကို သုံးပြီးတော့ .. Frame တွေကို Point to Point Directly connected node တွေ ကြား ပို့ဆောင်ပေးပါတယ်. Data Link Layer မှာ synchronization, bit error detection/correction error control, and flow control စောတွေ လုပ်င်ဆောင်ပါတယ်. switch သည် Layer 2 device ဖြစ်ပါတယ်..

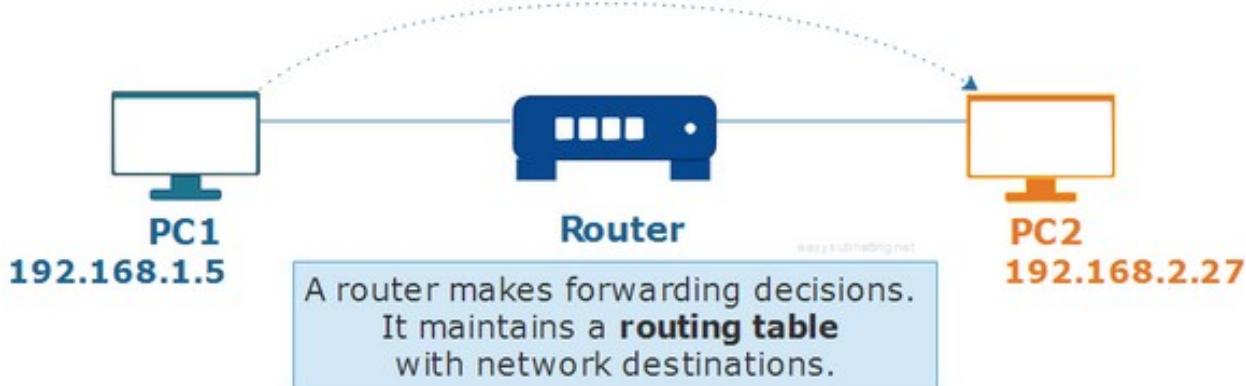


10.3. Network Layer

OSI Layer ၂ ဒဲ ၃ မြောက် Layer ကတော့ **Network Layer** ဖြစ်ပါတယ် Network layer က data packet တွေရဲ့ logical addressing(ဥမှာ IP address) အနုက်အလက်တွေကို တာယန်ယူ ဆောင်ရွက်ပေးပီး၊ တဲ့ အဲဒီ packet တွေကို သွားချင်တဲ့ (destination address) လို့ရာခရီးရောက် အောင်ပို့ဆောင်ပေးပါတယ်.



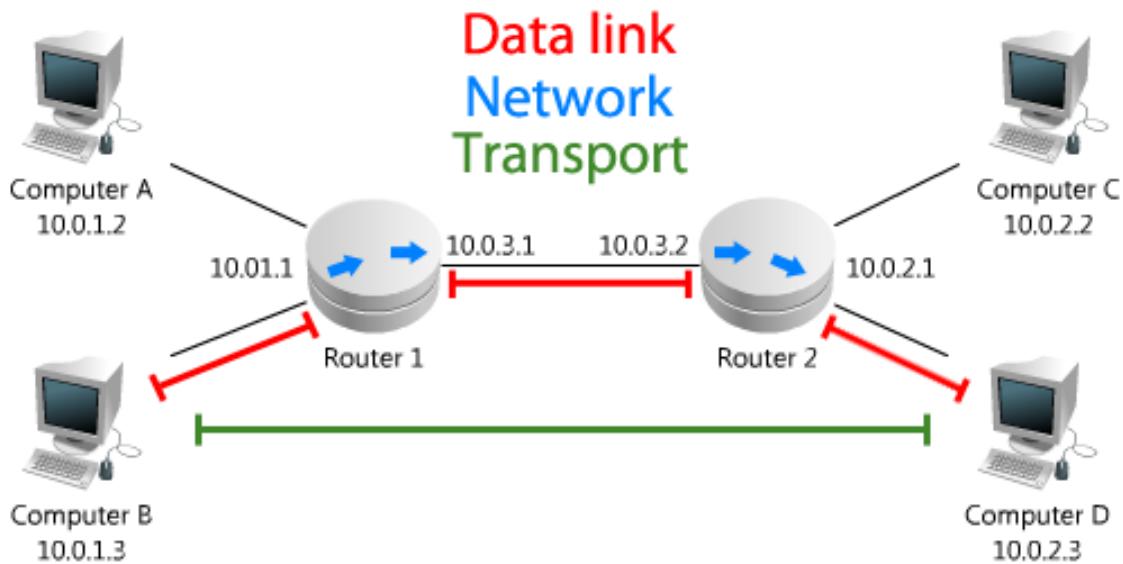
သူရဲ့ Function တွေကတော် routing and Forwarding, addressing, internetworking, traffic problems management (flow control), error handling, congestion control and packet sequencing တို့ပဲ ဖြစ်ပါတယ်. သူရဲ့အလုပ်လုပ်ပုံကတော် packet တရာ်ပင်လာပါဆိတာနဲ့ memory ထဲမှာ မှတ်သားသိမ်းဆည်း ထားတဲ့ routing table ကိုကျဉ်းပြီးတော် ဝင်လာတဲ့ Packet သွားရမယ့် destination ရဲ့ logical address ကို ကျဉ်းပြီး routing table ထဲမှာ ရှိခဲ့ရင် အဲ packet နဲ့ သက်ဆိုင်တဲ့ Network ကို ပို့ဆောင်ပေးပါတယ်. ပို့ဆောင်ပေးတဲ့နေရာမှာ လဲ destination ကိုသွားနိုင်မယ့်လမ်းက တစ်ခုမှကရှိနေရင် Data Packet တွေကို ဖြန့်ပြီးပို့ဆောင်ပေးပါတယ်. Network Layer မှာ သုံးတဲ့ protocol ပေါ်မှတ်ည့်ပြီး packet ရဲ့ destination ကို သွားနိုင်မယ် လမ်းကြောင်း(Route) ကို ရွှေ့ချယ်ပုံချိန်းမတူညီရှာပါဘူး. Logical address တွေက same network မှာ computer တစ်လုံးနဲ့ တစ်လုံးမတူရပါဘူး.. Network Layer မှာ သုံးတဲ့ logical address ကို TCP/IP သုံးထားတဲ့ network မှာဆို IP address လို့ခေါ်ပါတယ်.



10.4. Transport Layer

နောက်တစ်ခုကတော် **Transport Layer** ပါ. သူကတော် OSI model မှာ 4th layer မှာ တည်ရှုပါတယ်. Transport Layer ကဘာတွေလုပ်ဆောင်ပေးလဲဆိုတော် sender နဲ့ receiver Node (logical end to end Point) တွေကြား အပြန်အလှန် ပိုလိုက်တဲ့ Dataတွေ ကို error-free ဖြစ်စေဖို့ Sequence အတိုင်းအစဉ်လို

ကုမ္ပဏီအောင် ဆုံးရှုံးမှု(data loss) မဖြစ်အောင် data တွေကို အကြိမ်ပိုမိုတဲ့မျိုး(Data Duplication)မဖြစ်အောင် လုပ်ဆောင်ပေးပါတယ်.



ဘယ်လိုမျိုးလဲဆိုတော့ သူရဲ့ အပေါ် Layer ဖြစ်တဲ့ session layer က ပိုလိုက်တဲ့ data packet တွေကို သေးဇသေးလေ့ဖြစ်အောင် ပိုင်းလိုက်ပါတယ်(data units ရဲ့ size ကဲ့ဒေါ်မယ်ဆိုရင်ပဲ) ပြီးတော့ တစ်ခြားလိုအပ်တဲ့ additional information လေးလေ့လိုက်ထည့်ပေးပါတယ်. ဘာ information လေးလေ့လိုတော့ packet တစ်ခုရွင်းဆီမှာ ရှိတဲ့ date ရဲ့ byte အရေအတွက်တွေ့, တစ်ခြား function information တစ်ခုဆီရဲ့ byte အင်ရအတွက်တွေ့, အပြင်ပြီးတော့ error control လုပ်ဖို့လဲ ထည့်ပေးထားပါတယ်. ဘယ်လို Error Control လုပ်တာ လဲဆိုတော့ sender နဲ့ receiver ကြားမှာ data transfer လုပ်တဲ့ အချိန်မှာ connection ကြောင့် ဒါမူမဟုတ်ဘူး။ media တစ်ခုကြောင့် data တစ်ခုရဲ့ ဖုက်စီးပြောက်ဆုံးတာမျိုး, data သွားနေတဲ့ လမ်းကြောင်းမှာ တစ်ခုတစ်ခု ယာက်က င်ပြင်လိုက်တာမျိုးတွေ ကို သိနိုင်တဲ့ Method တွေ သုံးခဲ့ အခြေခံ တွက်နည်းပုံစံ တွေနဲ့ တွက်ချက်ပါး ပါး sender က ပိုလိုက်တဲ့ data တွေဟာ original အတိုင်းရှိမရှိကို စစ်ဆေးပါတယ်. Error Control ကို ကြည့်ပြီး စစ်ဆေးလိုက်လို့ original အတိုင်း မဟုတ်တဲ့ data packet ဆုံး အဲ ဒေါ် အဲ data packet တစ်ခုတည်းကိုပဲ ပြန်ပိုမို request လုပ်ပါတယ်. အဲအခါဂျာရင် sender ဘက်က Transport Layer ကနေ ပြန်လည်ပို့ဆောင်ပေးပါတယ်.

Data တွေကို ပိုတဲ့ နေရာမှာ တစ်ဘက်နဲ့ တစ်ဘက် end-to-end reliable ဖြစ်ဖို့ အတွက် data လက်ခံရရှိပြီးတိုင် receiver ကနေ sender ကို data ရရှိကြောင်း (acknowledge ပြန်) ပြောပါတယ်. အကယ်လို့ receiver ဘက်က နဲ့ data လက်ခံ ရရှိကြောင်း ack မပြန်ဘူးဆုံး sender ကအချိန်အတိုင်းအတာ တစ်ခုထိတောင့်ပြီး နောက်တစ်ကိုမဲ့ ထပ်ပို ပါတယ်(Retransmitting Data). နောက်တစ်ခုက Flow control ပါ ကျွန်ုတော်တို့ data packet တွေ network အပေါ်သွားတဲ့ အခါမှာ ကြားမှာရှိတဲ့ router ကွဲ switch ကွဲ Gate way တွေကို အဆင့်ဆင့်ဖြတ်ပြီး သယ်သွားရပါတယ်. အဲလိုသွားတဲ့အချိန်မှာ routerတွေ Gate-way တွေကြား သွားနိုင်တဲ့ speed တွေ



bandwidth တွေကို Learning လုပ် လေ့လာမှတ်သားပြီးတော့ ဘယ်router နဲ့ ဘယ်ကြားဆို ဘယ်လောက်နှင့် နဲ့သွား ဘယ်နဲ့ဘယ်ကြားဆို ဘယ်လောက်သွားဆိုတာ မျိုးကို Transport Layer ကနေထိန်းချုပ်လုပ်ဆောင်ပါတယ်. Transport Layer မှာ connection service မျိုးသုံးပါတယ် Connection-Oriented (TCP) နဲ့ Connectionless service (UDP) ပါ ဒီလောက်ဆိုရင် Transport layer အလုပ်လုပ်ပုံကို သဘောပေါက်လောက်ပါ။ ;-)

TCP

Use for Data Sensitive

Reliable

Connection-oriented

**Segment retransmission
and flow control through
windowing**

Segment sequencing

Acknowledge segments

UDP

use for Real time (like streaming video)

Unreliable

Connectionless

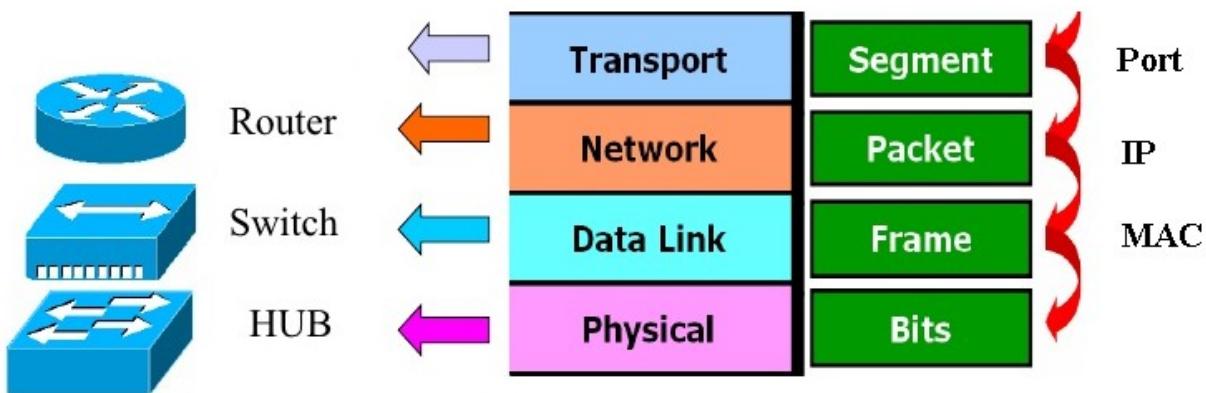
**No windowing or
retransmission**

No sequencing

No acknowledgement

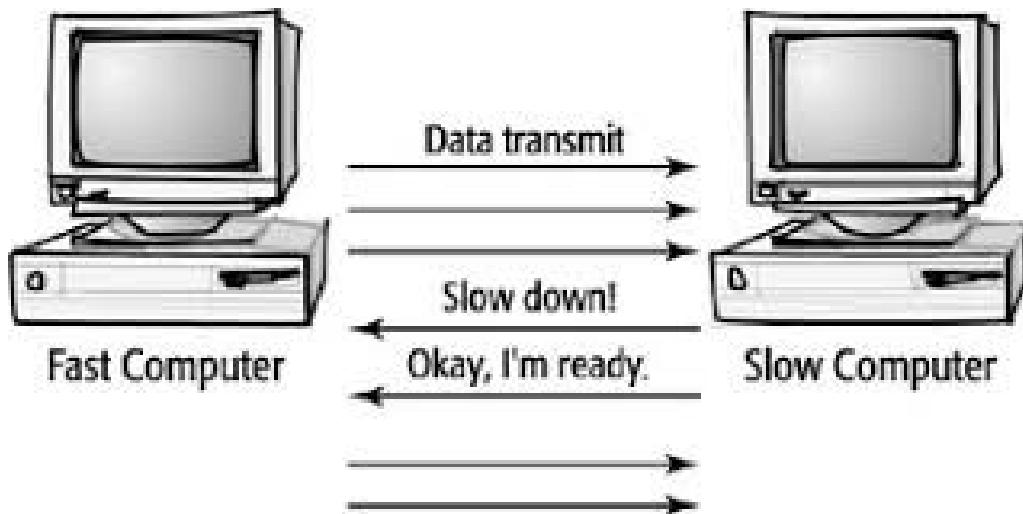
CCNA (Routing & Switch) ခုမှ စလုပ်မယ့် သူတွေအနေနဲ့ ခုပြာခဲ့တဲ့ layer 4 ခုကို ကောင်းကောင်း သဘောပါက်ဖို့

လိပ်ပါတယ်။



10.5. Session Layer

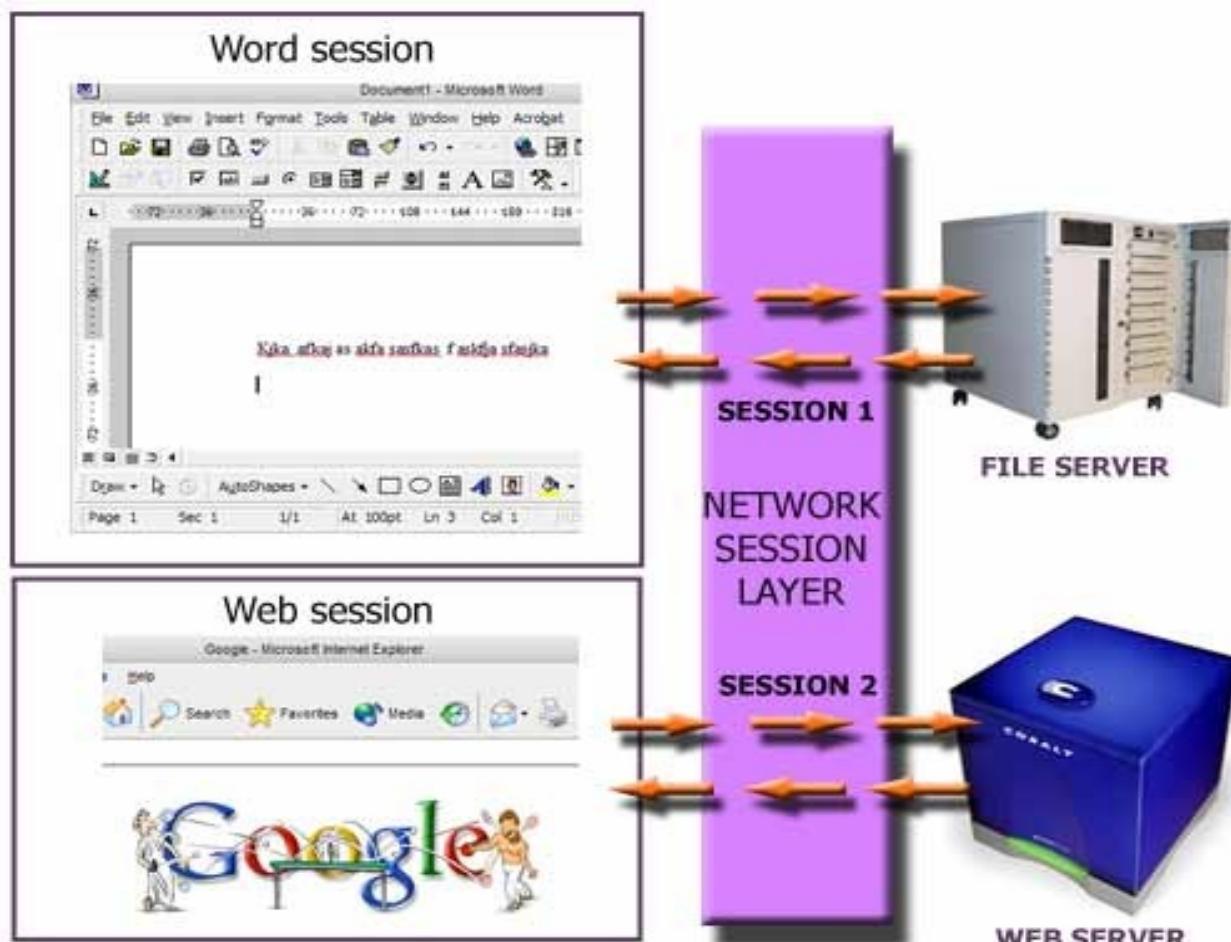
Session Layer က ကွန်ဖူတာ တစ်လုံးနဲ့ တစ်လုံးကြောင် Logical connection ကို control လုပ်ပေးပါတယ်. တစ်နည်းပြောရရင် machine တစ်ခုပေါ်က application process တစ်ခု နဲ့ တစ်ခြား machine ပေါ်က application process တစ်ခုကြေား connection တစ်ခု(session တစ်ခု) စတင်ထိုး(establish), ထိန်းသိမ်းထိုး(Maintain) နဲ့ data တွေ ပို့ဆိုပြီးသွားရင် ဖုက်သိမ်းထိုး(terminate) လုပ်ဆောင်ပါတယ်. ပြီးတော့ အဲ connection ကို simplex လား Half Duplex လား Full Duplex လား ဘယ် mode နဲ့ လုပ်ဆောင်မယ်ဆိုတာကို session layer ကဆုံးဖြတ်ပါတယ်. အဲသည့် sessions တွေ ကို dialogs လိုလဲ ခေါ်ပါတယ်.



ဘာလိုလဲ ဆိုတော့ ကျွန်တော့တို့ လူတွေ ဖုန်းပြောပုံနဲ့ session အလုပ် လုပ်ပုံတူလိုပါ. ဥပမာ.. လူတစ်ယောက်(A) ကနေ ဖုန်းခေါ်လိုက်လို့ တစ်ခြားတစ်ယောက်(B) ကဖုန်းကိုင်ပြီးရင် ဖုန်းကိုင်လိုက်ကြောင်း စကားစပောလို့ရကြောင်း အချက်ပြတဲ့ အနေနဲ့ hello ဆိုပြီး လုပ်ပါတယ် တစ်နည်းပြောရရင် session establish လုပ်တယ်ပေါ့မှာ.. Hello လုပ်ပြီး establish လုပ်လိုက်တာလဲ တစ်ဘက်နဲ့တစ်ဘက် စကားစပောရာတွေတာပဲ(Data transfer လုပ်တယ်)



၃၆) အဲတော့ စကားလဲ ပြောစရာမရန်တော့ဘူးဆိုတော့မှ ဒါပံ့နော ဆိုပြီး ဖုန်းချုပ်လေ. အဲတာ session terminate လုပ်လိုက်တာပေါ့မှာ.. Follow ကတော့ အတူတူပါပဲ



Source: www.teach-ict.com



အဲလောက်ဆို session layer အလုပ် လုပ်ပုံကို မြင်လောက်ပြီလို ယုံကြည်ပါတယ်..

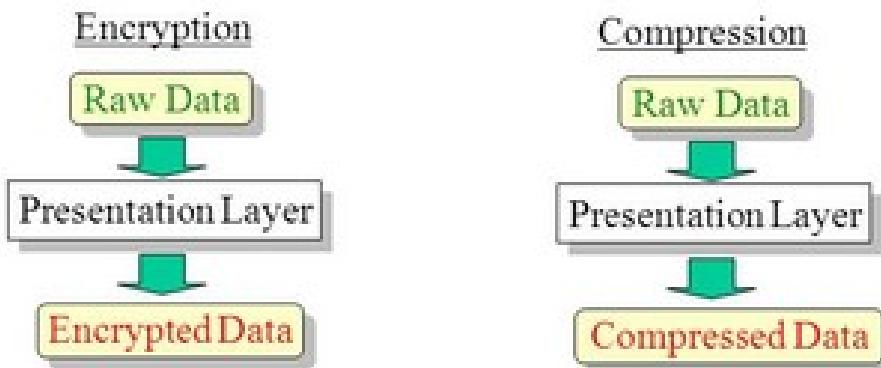
10.6. Presentation Layer

Presentation Layer- ကဘာတွေလုပ်နိုင်လဲဆိုတော့.. application Layer ကနေ ပိုလိုက်တဲ့ data က တွေကို sender ဘက်ခြေးကနေ မပိုခင်မှာ Network ထဲမှာ သွားလာနိုင်တဲ့ Format ပြောင်းပေးပါတယ် Encode လုပ်တယ်လိုလဲပေါ်ပါတယ် (Character code translation>> from..ASCII to EBCDIC) ဥပမာပြောရင် ဒီဘက်က sender မှာ သုံးတဲ့ Character code(Data Representation) က ASCII (American Standard Code for Information Interchange) ဖြစ်ပြီး ဟိုဘက်တစ်ချားဘက်ခြေးက receiver မှာ သုံးတဲ့ Character Code က EBCDIC (Extended Binary Coded Decimal Interchange Code) ဖြစ်နေနိုင်ပါတယ်..

Presentation Layer သာက်က ASCII code တွကို Network ထဲမှာသွားလိုရမယ် format ပြောင်းပြီး dataတွကို ပိုလိုက်ပါတယ်(encoding လုပ်တယ်လိုပေါ်ပါတယ်) အဲပိုလိုက်တဲ့ dataတွကို လက်ခံတဲ့ဘက်ပူစ်မှာ ရှိတဲ့ Presentation Layer ကနေပြီးတွေ EBCDIC code ကို ပြောင်းလဲပေးပါတယ် (decoding လုပ်တယ်လိုပေါ်ပါတယ်). Encoding တစ်ဘက်နဲ့ တစ်ဘက် သုံးတဲ့ Mechanism တွေကတူဖွင့်မှတူပါလိမ့်မယ်..

Layer 6: Presentation Layer

Major Functions: Encryption, compression, etc



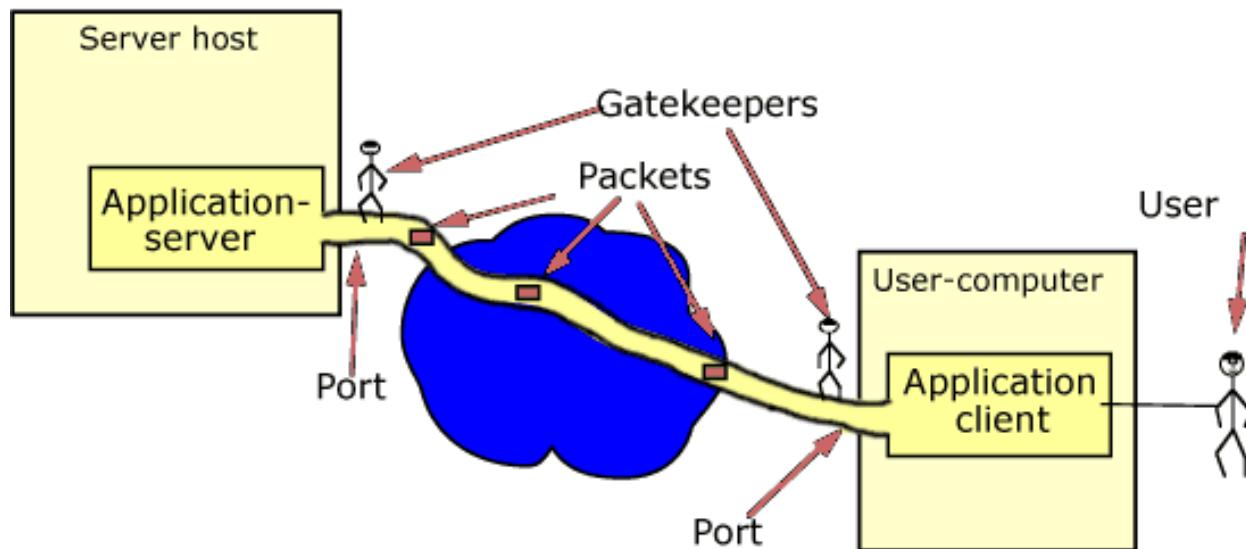
နောက်တစ်ခုကတွေ့ ကျွန်တော်တို့ data တွေ ဟိုဘက်ကနေ ဒီဘက်ကို transfer လုပ်တဲ့ အဆိုန်မှာ secure ဖို့စေနိုင်တယ် Encryption လုပ်တာကလဲ Presentation Layer ကနေလုပ်ဆောင်ပေးတာဖြစ်ပါတယ်. ဟိုဘက် ဒီဘက် Encryption Decryption လုပ်တဲ့ စနစ်က တူညီဖိုလိုပါတယ် ပြောရရင် Encode/Decode လို ကွဲပြားလိုမပါဘူး. နောက် Presentation Layer ကနေလုပ်ပေးနိုင်တဲ့ တစ်ခုကတွေ့ Compression/Decompression ပါပဲ.. ကျွန်တော်တို့တွေ data ကို network တစ်ရွောက်မပိုစ်မှာ Data ကို compress လုပ်ပါတယ် compress လုပ်လိုက်တော့ Size သေးသွားတာပေါ့ size သေးတွေ့ မြန်မြန် transfer လုပ်လိုရတာပေါ့မှာ.. ဟိုဘက်တစ်ခြားဘက်ရောက်တွေ့ရာမှ ရလာတဲ့ data ကို decompress ပြန်လုပ်လိုက်မှသာ original အတိုင်းပြန်ရမှာပါ.. အရာဉ်းခြီး ပြောရရင်တွေ့မှာ.. data ကိုမပိုစ်မှာ ပထမဆုံး encode လုပ်လိုက်ပါတယ် ပြီးတွေ့ security အတွက် encode လုပ်ထားတဲ့ data ကို encrypt လုပ်ပါတယ်. ပြီးတွေ့ network ပေါ်မှာ ပိုရတာမြန်အောင် data size ကြိုးမနောင် အောင် compress လုပ်လိုက်ပါတယ်. အဲလိုလုပ်ပြီးတွေ့မှ ပိုလိုက်ပါတယ်. လက်ခံတဲ့ဘက်ခြိမ်းမှာရွေတွေ့ ရလာတဲ့ data ကို ပထမဆုံး decompress အရင်လုပ်ပါတယ်. Original အတိုင်းရပြီးဆုံး decryption ပြန်လုပ်ပါတယ်. ပြီး မှ မှန်ကန်တဲ့ format နဲ့ data ကို decode ပြန်လုပ်ပါတယ်.. ကဲအဲတာကတွေ့ presentation layer ရဲ့ အလုပ်လုပ်ပုံဖြစ်ပါတယ်.

Functions Provided by Presentation Layer



10.7. Application Layer

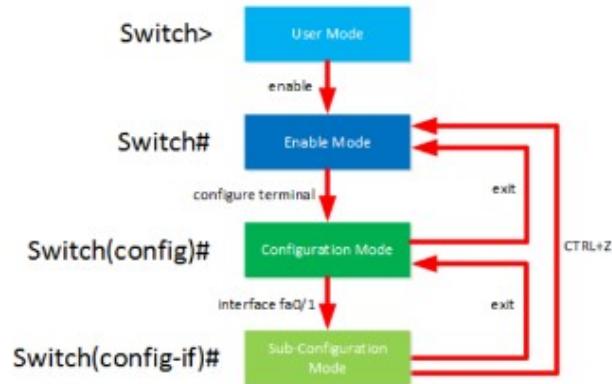
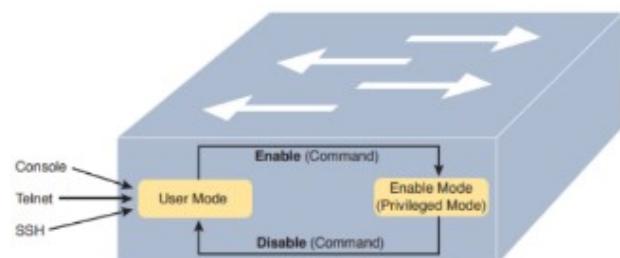
Application Layer ကတွေ့ ကျွန်တော်ထို user တွေနဲ့ အနီးကပ်ဆုံး Layer ဖြစ်ပြီး User တွေနဲ့ ဆိတ်ဆက်မှုများ၊ interface တစ်ခုအနေနဲ့ လုပ်ဆောင်ပေးပါတယ် ..သူက ဘာတွေဘယ်တွေ မှာလုပ်ဆောင် ပေးလဲ ဆိတ်အဓိကအားဖြင့် end-user process တွေနဲ့ software application တွေကြား မှာလုပ်ဆောင်ပေးပါတယ်.



ဒီနေရာမှာ တစ်ခုပြုချင်တာက application Layer ဆိတ် ကျွန်တော်ထိုတွေ သုံးနေတဲ့ software application တွေကိုဆိုလိုတာမဟုတ်ပါဘူး.. ကျွန်တော်ထိုတွေ network တွေကိုဖြတ်ပြီး data တွေပဲ မယ် ဆိတ်ဘာရာမှ application layer protocol တွေကတစ်ဆင့်လုပ်ဆောင်ပေးတာပါ.. ပိုပြီးရင်းသွားအောင် ရုပ် မာဇားရရင် ကျွန်တော်ထို မေးလုပ်တစ်စိမယ်ဆိုရာပါတွေ့ အဲတော့ဆို Gmail(E-mail) စုတဲ့ software application တစ်ခုသုံးပြုလိုက်မယ် ဆိုရင် application layer protocol ဖြစ်တယ်ထဲ့ SMTP(simple mail transfer protocol) ကိုသုံးရပါတယ်. ကျွန်တော်ထို Web Page တွေ ကြည့်မယ် web site တွေသွားမယ်ဆို ရင်တွာ Application Layer Protocol ဖြစ်ထဲ့ HTTP(Hyper Text Transfer Protocol) နဲ့ သွားပါတယ်။ Application Layer ရဲ့ အလုပ်လုပ်ပုပါ 😊

11. Cisco Lab basic

11.1 Cisco CLI Level



Lab စလုပ်လုပ်ချင်း ဒါမှမဟုတ် ခုမှ စလေ့လာတဲ့ သူတွေ မှားတက်ရာတာက ကိုယ်ဘယ် mode ကို င်ရာက်နေလဲ သတိမထားမိတာနဲ့.. Configuration command တွေ mode မှားရှိက်တက်ရာတာပဲ..

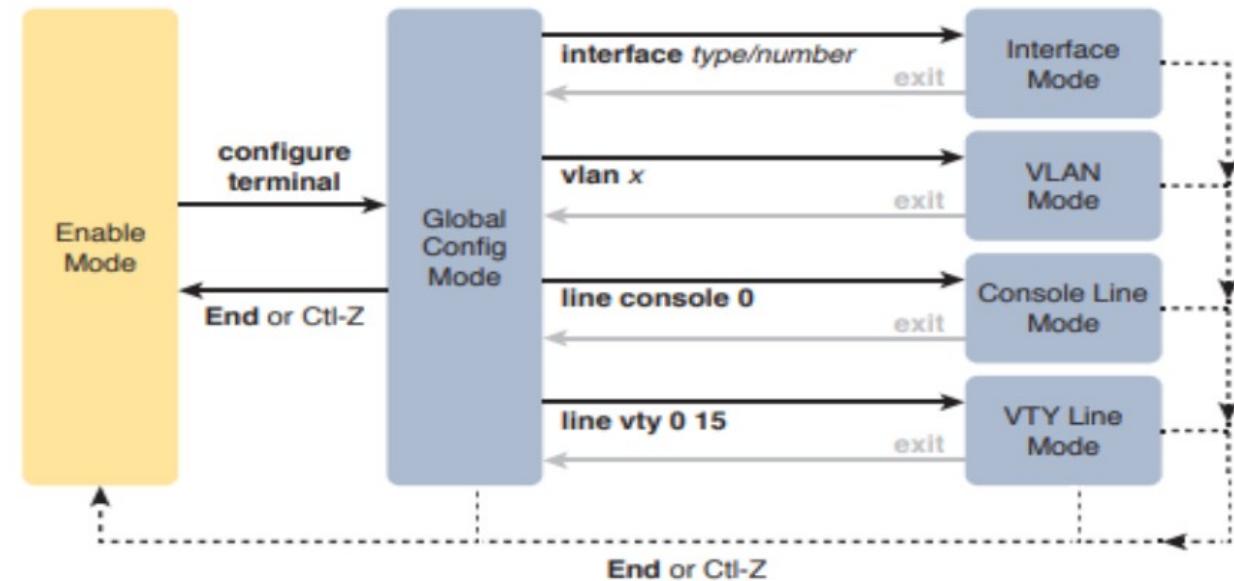
အဲတော့ မှတ်ထားရမှာက

switch> ဆို usermode သူက > sign လေးနဲ့

switch# ဆို enable mode လို့ ခေါ်တဲ့ privilege mode သူက **#** sign လေးနဲ့ ..

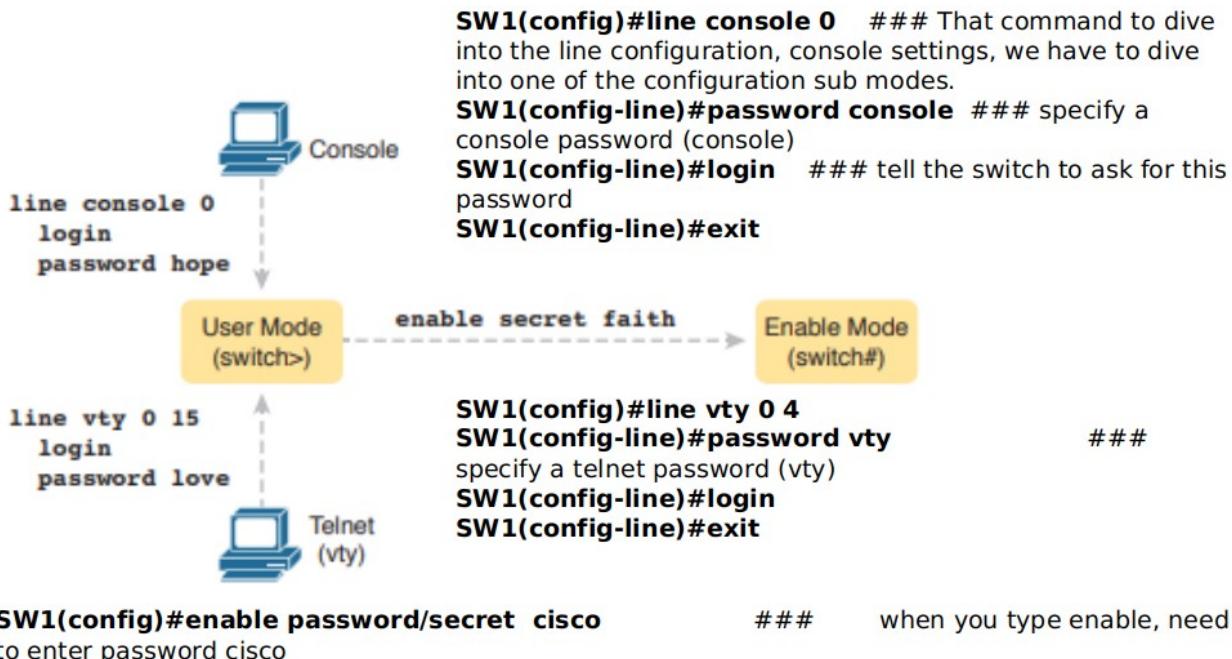
switch(config)# ဒါကတွေ global configuration mode ပါ သူကတွေ **#** sign လေးအပြင် **(config)** ဆိုတာလေးပါ ပါဝါတယ်။ နောက်တစ်ခါ interface တိ_lineတိ_vlan တို့ထဲ ဝင်ရင် sub configuration mode ထဲကိုရောက်ပါတယ်။ အဲရှာမှ သူတို့နဲ့ သက်ဆိုင်တာကို ရှိက်လို့ရတာပါ။

ခုမှ စြိုး lab လုပ်တာဆိုတော့ ကိုယ်တစ်ခုရှိ ရှိက်လိုက်ရင် ဘာ ဖြစ်သွားမလဲ တွေးပါ သတိထားရာ ညွှပ်ပါ။ command တွေက သူ mode နဲ့ သူ ရှိက်ရတာမှားပါတယ်။ အဲတော့.. Mode မှားပြီး command ရှိက်မိရင် ဘယ်လို့ error တက်မလဲ ဆိုမိုး ကို သတိထား ရှာညွှပ်ပြီး lab လုပ်သွားရင် နောက်ပိုင်းရာ Lab လုပ်ရတာ လွယ်သွားမှာပါ။



ပုံမှာ ပြထားတဲ့ အတိုင်း exit နဲ့ဆို တစ်ဆင့်ချင်း ထွက်သွားမှ ဖြစ်ပြီး end/ ctrl+z ကိုနိပ်ရင်တော့ enable mode ကို တန်းရောက်သွားမှာပဲ ဖြစ်ပါတယ်။

11.2 Login Lab



ဒီနေရာမှာ တစ်ခုမှတ်ထားရမှာက **login** ဆိုတဲ့ keyword ရှိကိုဖို့ မေ့တက်ရာတယ်။ ကိုယ်က **Password {your password}** ရှိက်ကြပြီး **Login** ဆိုတဲ့ command ရှိကိုဖို့ မေ့ခဲ့ရင် ကိုယ်

device ကို ဝင်တဲ့ အဆိုန်မှာ **password** တောင်းမှာမဟုတ်ပါဘူး.. အဲဒေါာ **authentication** လုပ်ခ စချင်ရင် **Login** ဆိုတာလေးကို မမေ့မရော့၊ ရိုက်ပေးဖို့ လိပါတယ်။ **login** ဆိုတာလေးရိုက်မှ **Login** ဝင်တဲ့အဆိုန်မှာ **Password** တောင်းမှာပါ။

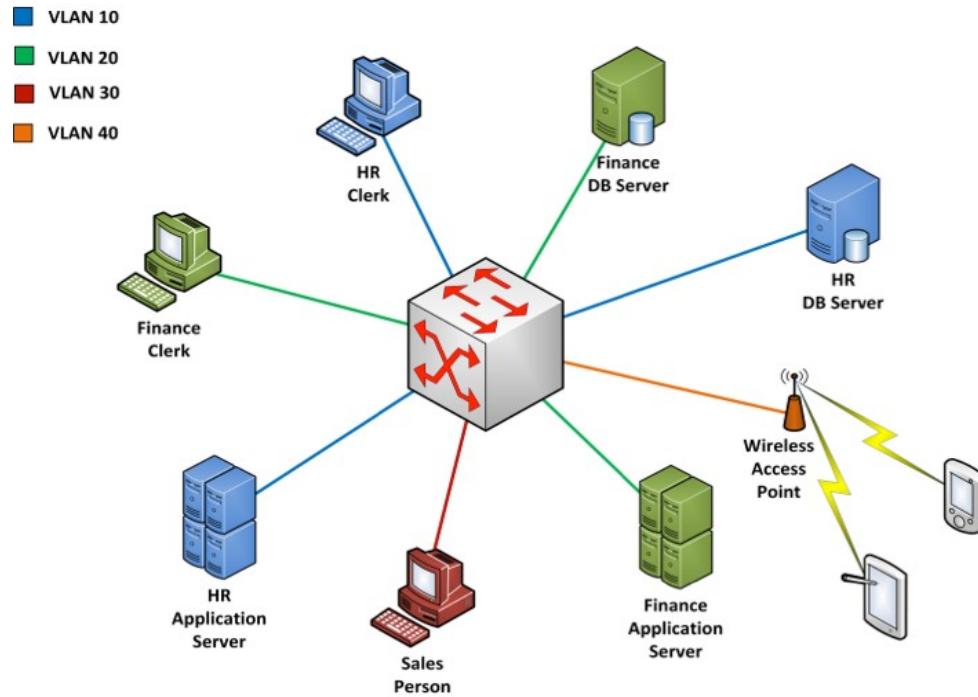
11.3 Intro to VLAN

VLAN ဆိုတဲ့အတိုင်းပဲ Virtual LAN network တွေ တည်ဆောက်မထုံး နေရာမှာ သုံးမထုံး feature တစ်ခါ။ ပြောရရင်တော့ physical switch တစ်ခုမှာ logical switch (different network) တွေ အများကြီးကို isolateဖြစ်အောင် ခွဲပေးလိုက်တဲ့ သဘောပါပဲ..

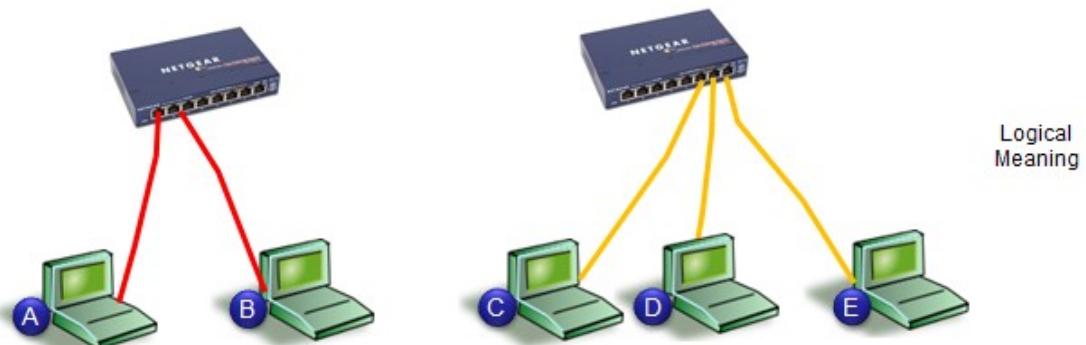
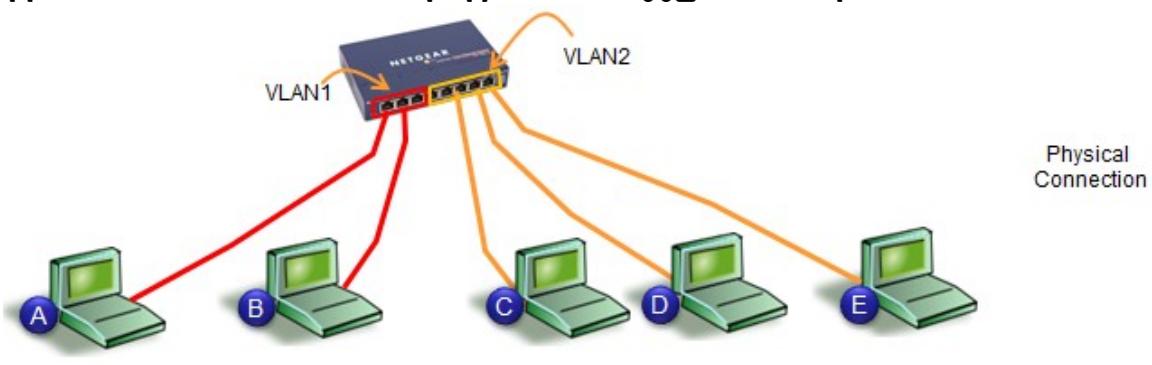
အဲလို ခွဲပြီးသုံးတော့ ဘာတွေကောင်းလဲ ဆိုတော့ VLAN တစ်ခုလီတိုင်းသည် different subnet (different network) ဖြစ်တဲ့အတွက် broadcast Domain က သတ်သတ်ဆီ ဖြစ်သွားပြီ။ အဲအတွက်ပြော မြို့VLAN "A" မှာ ဖြစ်နေတဲ့ Broadcast တွေသည် different network မှာ ရှိနေတဲ့ တစ်ခြား VLAN "B" တို့ VLAN "C" တို့ကို ရောက်မှာမဟုတ်ပါဘူး။ အဲ အတွက်ပြောင့် **performace** အနေနဲ့ ပိုကောင်းလာပါတယ်။

Different Network connectလုပ်မထုံးဆို**Layer 3 device** ဖြစ်တဲ့ Routerတို့ Layer 3 switch တို့လိုပါတယ်။ အဲကောင်တွေသည်လဲပဲ broadcast packet တွေကို forward လုပ်မပေးပါဘူး. သူ ဆီရောက်လာတဲ့ broadcast packet တွေကို **drop** လုပ်ပစ်လိုက်တာပါ။

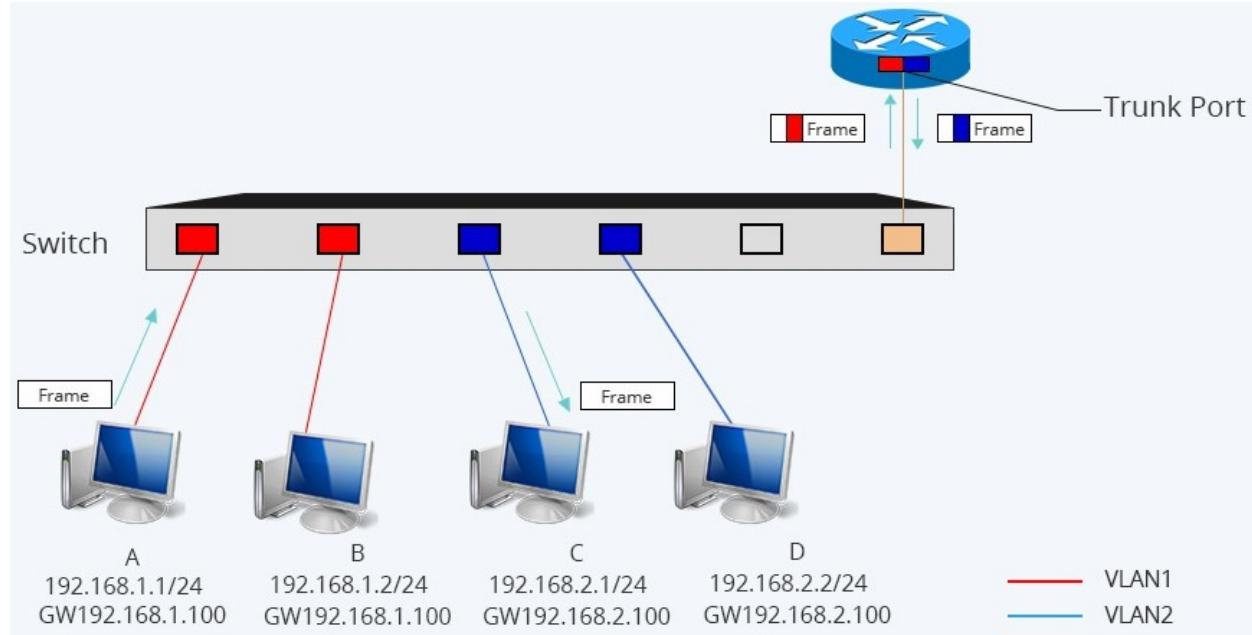
ရုံးတစ်ရုံးမှာ Department တွေရှိမယ် ဆိုရှပါစိုး.. HR dep, Finance dep, Sales dep, Engr dep, IT dep .. စသည်ဖြင့် အဲလို department တွေရှိမယ်ပေါ့.. မြို့departmentတွေက computerတွေကို same network ထဲမှာ ထားတာထက် vlan တွေ ခွဲပြီး သူဆိုင်ရာ ဆိုင်ရာ domain(Dep) တွေကို different network အနေနဲ့ထားတာက ပိုကောင်းပါတယ်။ **Management** လုပ်ရတာလဲလွယ်တယ်။ **TShoot** လုပ်ရတာလဲ ပိုကောင်းတာပေါ့. ပြီးတော့ same network တစ်ခုထဲမှာ အကုန်စုပြုထားတာထက်တာရင် သူသားဆိုင်ရာ သက်ဆိုင်ရာ လေးတွေကို different network အနေနဲ့ ထားတာက **security** အရလဲ level တစ်ခုထိ ကောင်းပါတယ်။



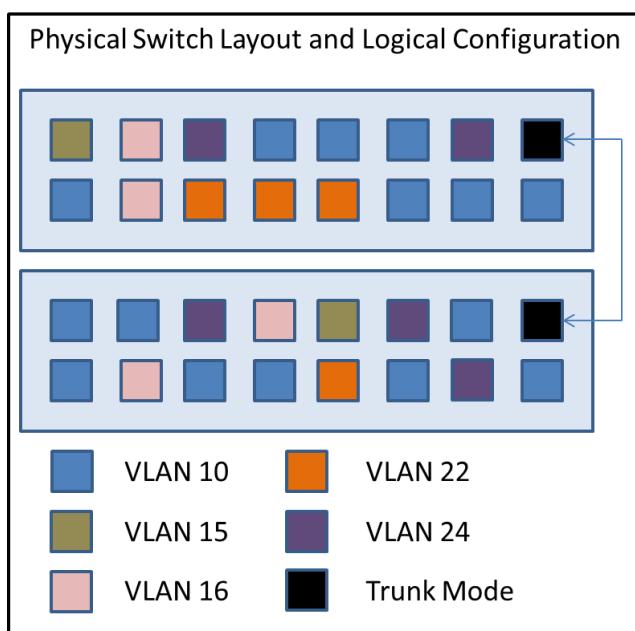
ခုပံ့ကတော့ Physical switch တစ်လုံးကိုမှာ VLAN တွေခဲ့ပြီး ထား ထားပဲ။



VLAN (different Network) အော် connect လုပ်မယ်ဆို Layer 3 (IP) ကို router လုပ်ပေးနိုင်တဲ့ Router လိုပါတယ်။ Router On a Stick Lab ရာရင် လုပ်ရင်းနဲ့ မှတ်စီထဲ ပိုမြင်လာပါလိမ့်မယ်။



Physical switch to switch ချိတ်တဲ့ interface တွေသည် Trunk ဖြစ်ဖိုလိုပါတယ်။ Trunk interface ကမှ vlan တွေကို tag လုပ်ပြီးသယ်နိုင်တာပါ။ tag မလုပ်ပဲ ဒီတိုင်းသယ်မယ်ဆို.. ဘယ်vlan ကလာလို့.. ဘယ် ကိုသွားလဲ



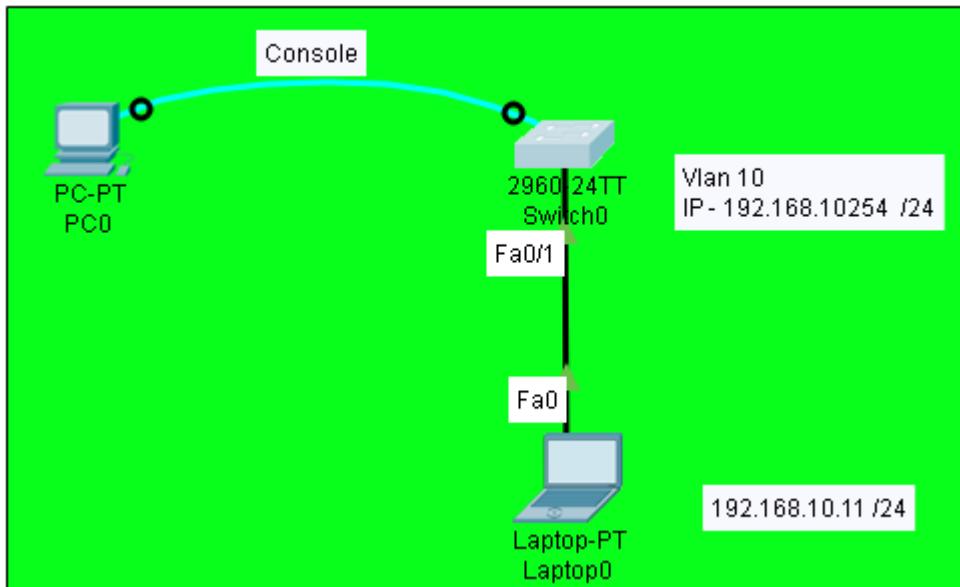
သိတော့မှာ မဟုတ်ပါဘူး ဥပမာ vlan10(Blue) အတွက် ပေးထားတဲ့ port ကနေလာတဲ့ packet ကို Trunk link (black) မေးမြေ switch to switch သယ်မယ်ဆို အဲ packet ကို vlan 10 မေးမြေသိတော်သိတော်ဆိုတယ် ဆိုတာသိမ့် encapsulation လုပ်ပေးမယ့် protocol တစ်ခု လိုအပ်ပါတယ်။

မေးမြေ protocol သည်..
ISL ဒါမ္မမဟုတ် do1q ကိုသုံးရပါတယ်။ ISL သည် cisco ရဲ့ ကိုယ်ပိုင် protocol ဖြစ်ပြီး 802.1q မေးမြေ IEEE ကပါ။ ခုနောက်ပိုမြော်မြော် Dot1q ပဲ ပါတွောတာကို တွောရပါမယ်။

Dot1q မှာ native vlan တို့ tag လုပ်ပြီးမသယ်ပါဘူး.. Native vlan ဆုံး Untag ပါ။ default Native VLAN မှာ VLAN 1ပါ။

11.4 VLAN lab 1

ဦးLAB မှာ ပထားမဆုံး console မှုံး login ဝင်ပြီး VLAN 10 ကို create လုပ်ပါ။ VLAN 10 interface ကို management အတွက် ip assign ပေးရမယ်။ interface fa0/1 ကို vlan 10 ထဲကို ထည့်ပေးရမယ်။ ဦးဇွဲ့ာ laptop ကနေပြီး telnet မှုံး Login ဝင်ရမယ်။ အဲဇား host name, console password, ssh/telnet password, enable password စောင့် config လုပ်ပေးရမယ်။ config တွက် save ခဲ့ပါ။

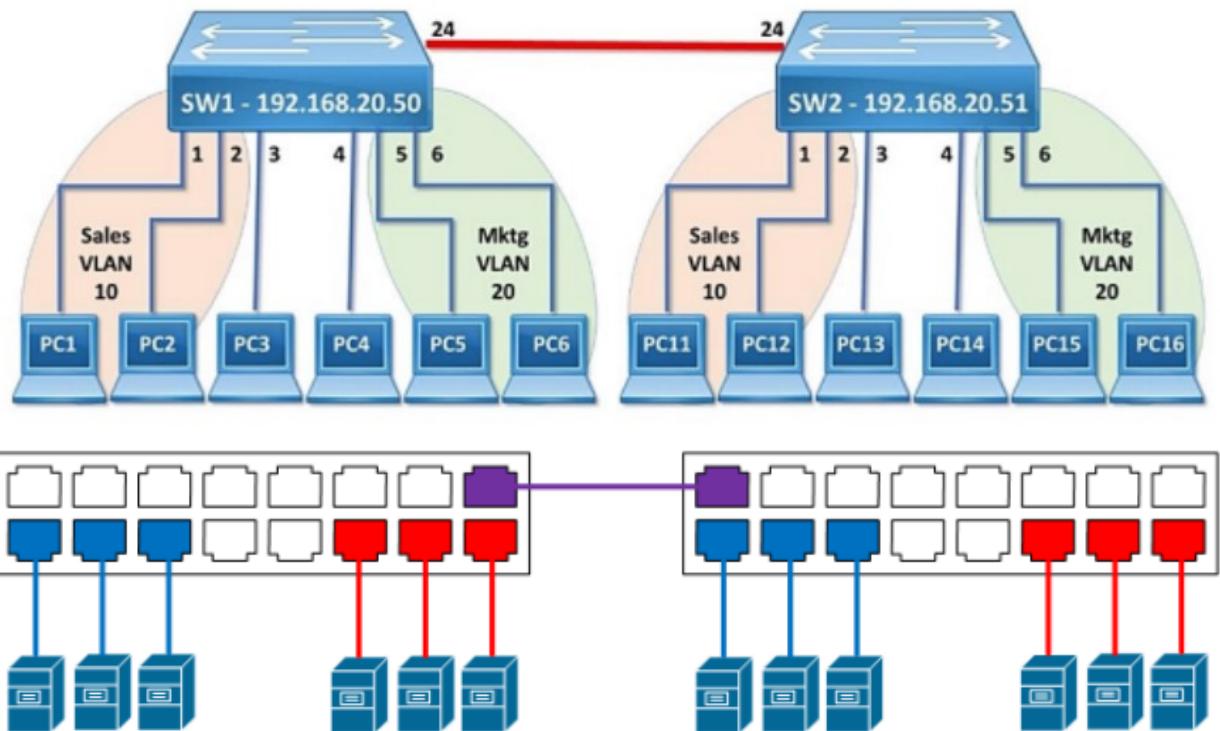


VLan Lab 1		
To change the Host Name of Switch	To configure telnet password	Configure Privilege Mode / Enable Password
Switch # configure terminal	SW1(config)# line vty 0 15	Configure privilege password
Switch (config) # hostname SW1	SW1(config-line) # password zoom	SW1 (config) #enable password ccna
SW1(config) #	SW1(config-line) #login	SW1(config) #enable secret zoom
To configure IP address on Interface VLAN 10	SW1(config-line) exit	
SW1(config)#vlan 10	To configure console password	



SW1(config-vlan)#name Mgmt	SW1 (config) #line console 0	
SW1(config-vlan)#exit	SW1 (config-line) # password zoom	
SW1 (config) # interface vlan 10	SW1 (config-line) #login	
SW1 (config-if) # ip address 192.168.10.254 255.255.255.0	SW1 (config-line) #exit	
SW1 (config-if) # no shutdown		To save configuration on switch
SW1 (config-if) #int fa0/1		SW1 # copy running-config startup-config
SW1 (config-if) #switchport access vlan 10		
SW1 (config-if) #end		

11.5 VLAN lab 2



VLAN Lab 2

Configure and implement VLAN

SW - Configuration	SW2 – Configuration	
SW1 #configure terminal	SW2 # configure terminal	
SW1 (config) # vlan 10	SW2 (config) # vlan 10	Vlan create လုပ်ပါ
SW1 (config-vlan) # name SALES	SW2 (config-vlan) # name SALES	Create လုပ်ထားတဲ့ vlan ကိနာမည့်ငါးပါ
SW1 (config-vlan) #exit	SW2 (config-vlan) #exit	
SW (config) # vlan 20	SW2 (config) # vlan 20	
SW1 (config-vlan) # name MKTG	SW2 (config-vlan) # name MKTG	
SW1 (config-vlan) #exit	SW2 (config-vlan) #exit	
SW1 (config) #	SW2 (config) #	
SW1 (config) # interface range fastethernet 0/1 -2	SW2 (config) # interface range fastethernet 0/1 -2	
SW1 (config-if-range) # switchport mode access	SW2 (config-if-range) # switchport mode access	Interface mode ထဲကို ဝင်ပါ။ access mode ခြောင်းပါ။
SW1 (config-if-range) # switchport access vlan 10	SW2 (config-if-range) # switchport access vlan 10	
SW1 (config-if-range) # exit	SW2 (config-if-range) # exit	Vlan ထဲကို interface ထည့်ပါ။
SW1(config) #	SW2(config) #	
SW (config)# Interface range fastethernet 0/5 -6	SW2 (config) # interface range fastethernet 0/5 -6	
SW1 (config-if-range) # switchport mode access	SW2 (config-if-range) # switchport mode access	
SW1 (config-if-range) # switchport access vlan 20	SW2 (config-if-range) # switchport access vlan 20	
SW1 (config-if-range) # exit	SW2 (config-if-range) # exit	

Switch1 ရဲ့ vlan 10/20 မှာရှိတဲ့ PC တွေကနေ တစ်ခြားသက်မှာရှိတဲ့ switch2 ရဲ့ VLAN10/20 ဗြိ PC
တွေကို ping ရှိပါ။ ping လို့ ရှိုးမှာ မဟုတ်ပါဘူး..

#####Verification#####

#show interface fastethernet0/x switchport
#show vlan brief

Switch#show vlan brief

VLAN	Name	Status	Ports
1	default	active	Fa0/3, Fa0/4, Fa0/7, Fa0/8 Fa0/9, Fa0/10, Fa0/11, Fa0/12 Fa0/13, Fa0/14, Fa0/15, Fa0/16 Fa0/17, Fa0/18, Fa0/19, Fa0/20 Fa0/21, Fa0/22, Fa0/23, Gig0/1 Gig0/2
10	MKTG	active	Fa0/1, Fa0/2
20	SALES	active	Fa0/5, Fa0/6

Switch1 ရဲ VLAN 10 ကနေ Switch2 မှာရှိတဲ့ VLAN 10 ကိုသွားမယ် ဒါမှာမဟုတ် Switch1 ရဲ VLAN 20 ကနေ Switch2 မှာရှိတဲ့ VLAN 20 ကိုသွားမယ်ဆို switch-to-switch နှင့် ထားတဲ့ interface သည် မတူညီတဲ့ VLAN တွေ ကို tag(encapsulation)လုပ်ပြီး သယ်သွားဖို့ အတွက် Trunk interface ဖြစ်နေဖို့လိုအပ်ပါတယ်။

Configure Trunking

SW1 – Configuration

```
SW1 # configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW1 (config)# interface fastethernet 0/24
SW1 (config-if)# switchport mode trunk
SW1 (config-if)# switchport trunk allowed vlan all
SW1 (config-if)# ^Z
SW1 #
```

#ping PC to PC

#show interface fastethernet0/24 switchport

Switch#show interfaces fastEthernet 0/24 switchport

Name: Fa0/24

Switchport: Enabled

Administrative Mode: trunk

Operational Mode: trunk

Administrative Trunking Encapsulation: dot1q

Operational Trunking Encapsulation: dot1q

Negotiation of Trunking: On

Access Mode VLAN: 1 (default)

Trunking Native Mode VLAN: 1 (default)

Voice VLAN: none

Administrative private-vlan host-association: none

SW2 – Configuration

```
SW2 # configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
SW2 (config)# interface fastethernet 0/24
SW2 (config-if)# switchport mode trunk
SW2 (config-if)# switchport trunk allowed vlan all
SW2 (config-if)# ^Z
SW2 #
```

<http://crossnetmm.com/>

Administrative private-vlan mapping: none
 Administrative private-vlan trunk native VLAN: none
 Administrative private-vlan trunk encapsulation: dot1q
 Administrative private-vlan trunk normal VLANs: none
 Administrative private-vlan trunk private VLANs: none
 Operational private-vlan: none
 Trunking VLANs Enabled: All
 Pruning VLANs Enabled: 2-1001
 Capture Mode Disabled
 Capture VLANs Allowed: ALL
 Protected: false
 Unknown unicast blocked: disabled
 Unknown multicast blocked: disabled
 Appliance trust: none

Switch#

#show interfaces trunk

```

Switch#show interfaces trunk
Port      Mode          Encapsulation  Status       Native vlan
Fa0/24    on           802.1q        trunking    1

Port      Vlans allowed on trunk
Fa0/24    1-1005

Port      Vlans allowed and active in management domain
Fa0/24    1,10,20

Port      Vlans in spanning tree forwarding state and not pruned
Fa0/24    1,10,20
  
```

Switch#|

Ping PC to PC from different VLAN..

12 Network Lab Setup

12.1 History of GNS3

Graphical Network Simulator-3

	
Developer(s)	Jeremy Grossmann
Initial release	2008; 10 years ago
Stable release	2.1.8 / 14 June 2018; 4 months ago
Repository	github.com/GNS3/gns3-server
Written in	Python
Type	Network simulator
License	GNU GPL
Website	gns3.com



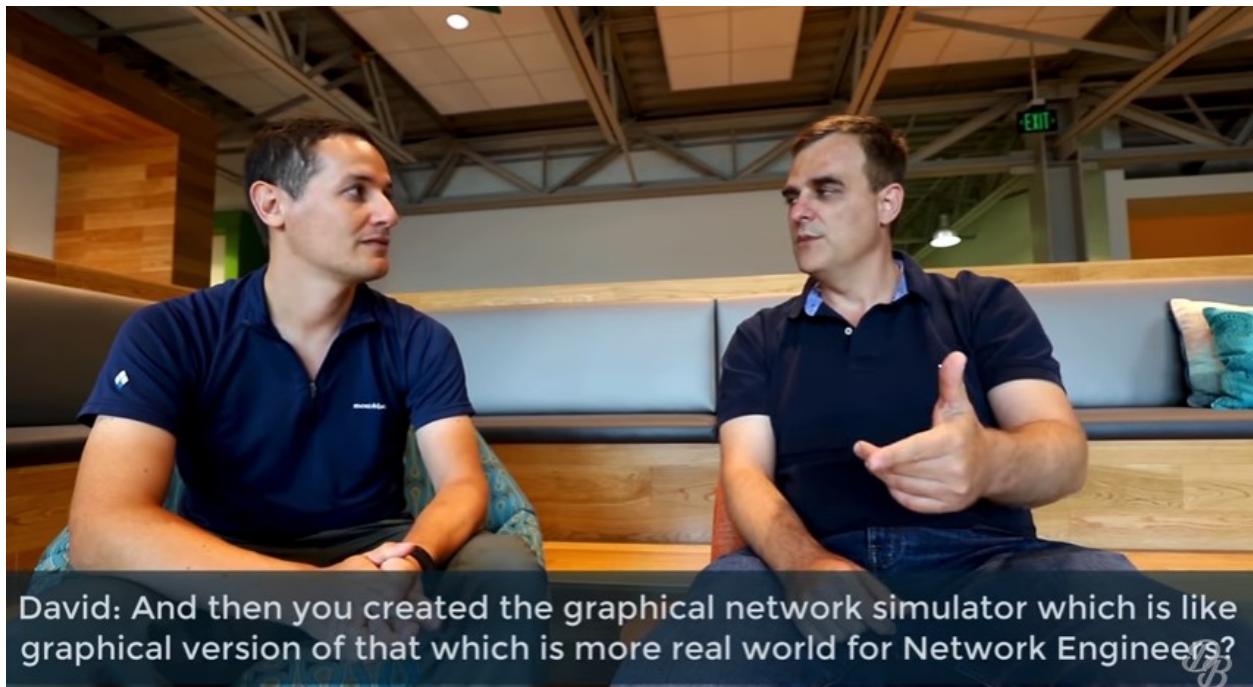
Jeremy Grossmann

ဟိုးအရင် GNS3 မပေါ်ခင်တုန်းက ဆို တို့တွေ ဘာနဲ့ cisco router တွေကို emulate လုပ်ရာလဲ ဆိုတော့ dynamips ဆိုတဲ့ emulator ကို သုံးပြီး run ရတယ်။ real device တွေနဲ့ မဟုတ်ပဲ သူက PC တွေ မှာ Router တွေIOS software Image ကို runပြီး lab စမ်းနိုင်အောင် emulation လုပ်ပေးတဲ့ computer program တစ်ခုပေါ့။ Cisco platforms 1700, 2600, 2691, 3600, 3725, 3745, and 7200 series cisco router တွေကို support ပေးပါတယ်။ ဒါမှာဟုတ် NS3 တို့ ဘာတို့လဲ ရှိသေးတယ်.. ကိုယ်တွေ ခေါ် ရာ အဲတာတွေနဲ့ စမ်းစရာမလိုဖော့တော့ဘူး။

ခု ဒီ Jeremy ဆိုတဲ့ ပုဂ္ဂိုလ်က သူ France မှာ မာစတာတက်တုန်း final year project အဖောက်၏ စတင်ခဲ့တာပါ။ hardware device တွေနဲ့ထက် .. အနုန်မရွေး real device တွေနဲ့ lab လုပ်သလိုမျိုး လုပ်လို့ငွောင် စိတ်ကူးပြီး စတင်အကောင်အတည်း ဖော့ခဲ့တာ ခုဆို .. Network Engineerတွေအတွက် အတော်လေး ကို အဆင်ပြေတဲ့ software တစ်ခုပေါ့။ GNS3 ကို python language ကိုအသုံးပြု ရေးသားထားပြီး GNU General Public License အောက်မှာပါရှိတယ်။ သူသည် opensource တစ်ခုဖြစ်တာဖို့ သူ source code ငဲ တွေအကုန်လုံးကို [github](https://github.com/GNS3/gns3-gui/releases) (<https://github.com/GNS3/gns3-gui/releases>)မှာ တင်ထားပြီး public မှာ ရှုတားပါတယ်။ contribute လုပ်ချင်တဲ့ မည်သူမဆို ပါဝင်လိုရပါတယ်။

ဒီနေရာ တစ်ခု သတိထားရမှာက cisco packet tracer သည် simulation လုပ်ပေးတဲ့ tools တစ်ခုပါ။ တစ်ကယ့် device တွေနဲ့ lab ရသလိုခိုး၊ ရအောင် အဆင်ပြေအောင် code ရေးထားတဲ့ software ပေါ့။ GNS 3 မှာ မြတ် Graphical Network Emulator ဆိုပေးမယ်，emulator (dynamips, QEMU, VM, etc) တွေသုံးပြီး IOS imag တွေကို တစ်ကယ် boot တက်ပြီး run ပေးတဲ့ emulator နော့။ အဲဒြောင့် သူက ram/cpu စားတယ်။

ကိုယ်သုံးရင် သုံးသလောက် RAM တွေ CPU တွေကုန်မယ်။ ပြောချင်တာက Device တွေ များကြီးနဲ့ lab လုပ်မယ်ဆို RAM/CPU မြင့်တဲ့ များတဲ့ စက်မှ အဆင်ပြေမယ်။ Packet tracer မှာ code တွေနဲ့ simulate လုပ်ပေးထားတာ ဖြစ်လို့ စက်အနိမ့်လေးတွေ Phone တွေမှာပါ အဆင်ပြေတယ်။



12.2 Why window user Need Gns3VM

ကဲ ခဲ GNS3 ကို install လုပ်တာရယ် သူ့ရဲ့ concept လေးကို အရင်ပြောပြုမယ်။ Cisco Devices က တွေကို run မယ်ဆို GNS3 တစ်ခုထဲသုံးမလား။ ဒါမှမဟုတ် GNS3 VM နဲ့ တွေသုံးမလား။ GNS3 တစ်ခုထဲဆို IOS တွေကို သုံးရမယ်။ GNS3 VM(linux base) ဆိုရင်တော့ IOU (IOS On Unix) ကို သုံးလို့ရတဲ့အတွေး IOS ထပ် CPU/RAM သုံးတာနည်းပါတယ်။ ပြီးတော့ IOU မှာ က Layer 2 ကိုပါ စမ်းလို့ရပါတယ်။ ပြီးခဲ့ရတဲ့ GNS3 vm သည် အကုန် လိုသမှု IOU support, docker, qemu စတာတွေ အဆင် ပြောပြုသုံးနိုင် ခဲ့ရတဲ့ pre config လုပ်ပေးထားတာပါ။ ubuntu 16.04 မှာ gns3VM လို့ pre config ရအောင် script များ run လို့ ရပေါ်မယ့် window သမား Mac သမားတွေ အနေနဲ့ ကတော့ remote server ဒါမှ မဟုတ် Gns3vm သည်သာ အကောင်းဆုံး option တစ်ခုပါ။

window user တွေအနေနဲ့ဆိုရင်တွေ GNS3 VM ကိုထွေပြီးသုံးစေချင်တယ်။ window သမားတွေက နှဲနဲ့က GNS3 VM မပါရင် IOU မရတဲ့အတွက် layer 2 စမ်းလို့ အဆင်ပြောမဟုတ်ပါဘူး။ ပြီးတွေ GNS3 VM ကို run နဲ့ virtual emulator တစ်ခုလိုပါတယ်။ အဲအတွက် VMware သုံးမလား Virtual box သုံးမလားပေါ့။ သူ့ recommend အရတွေ GNS3 ကိုသုံးသင့်ပါတယ်။

ဘာလိုလဲဆိုတွေ.. VMware မှာက nested virtualization ကို support ပေးတဲ့အတွက်ကြောင့်ပါ။ GNS3 VM မှာက ubridge ကိုအသုံးပြုပြီးတွေ compute server (Dynamips, QEMU,,etc) တွေ တစ်ခုခုနဲ့တစ်ခုခု UTP tunnel လုပ်ပြီး နိုတ်ဆက်တဲ့အခါဂ္ဂ nested virtualization ရှိမှာ အဆင်ပြောပါ။ virtual box သုံးမယ်ဆိုရင်တွေ နေးကွုးနေ့ပါလိမ့်မယ်။ vbox version6 မှာတွေ Nested virtualization ရသွားပါပြီ။

Linux user တွေအတွက်ကတွေ IOU support ကိုလို local machine မှာ လိုချင်ရင် GNS3 ရဲ့ official doc မှာ ပါတဲ့အတိုင်း config လုပ်ပေးလိုက်ယုံပါဝဲ။ Docker support လိုချင်ရင် additional config လေးထည့်လိုက်ရင်ရပါတယ်။ အဲလို တစ်ခုခုနဲ့ config ချုပ်တာအလုပ် ရှုပ်တယ်ထင်ရင် GNS3 ကိုသွင်းပြီး GNS3 VM နဲ့ နိုတ်ထားလိုက်ရင် ရပါတယ်။ GNS3 VM မှာက အဲတာတွေကို pre-config အကုန်လုပ်ပေးထားပါတယ်။

အဲတွေ ခုလုပ်မလှ့ task ကို ပြန်ပြောမယ်..

1. Install gns3
2. Install VMware workstation/Player or Virtual Box
3. Install GNS3 VM on Virtual Machine

12.3 GNS3 install (with Several ways)

တစ်ခုမှတ်ထားရမှာက GNS3 ။ version 2.1.11 ဆုံး GNS3 VM ကလဲ version 2.1.11 ဖြစ်မှ ရမှာပါ။ တစ်ခြား version ဖြစ်လိုပေါ်ဘူး။

ပထားဆုံး GNS3 ကို install လုပ်ပါ။

For Window > <https://www.gns3.com/software/download>

For linux >

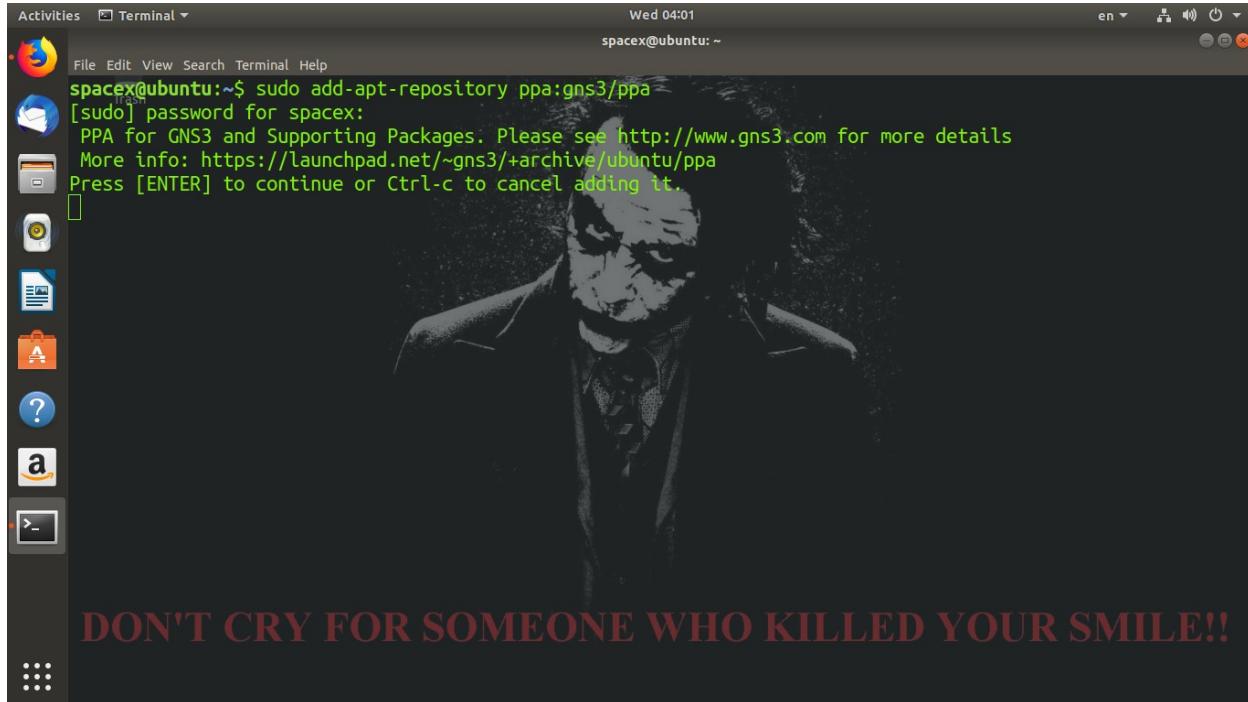
<https://docs.gns3.com/1QXVIihk7dsOL7Xr7Bmz4zRzTsJ02wklfImGuHwTlaA4/index.html>

ခုအမိက ထားပြီး ပြောသွားမှာကတွေ Linux ဘက်ကို အမိကထားပြောသွားမှာပါ။ ပြီးတွေ တစ်ခြားလိုအပ်တဲ့ concept တွေ

<http://crossnetmm.com/>

။။။။။ linux ကော် window မှာပါ အတူထူပါပဲ။ ဒု ubuntu မှာဆို ppa(Personal Package Archive) ။
ရင်ဆုံး addလိုက်ပါ။

sudo add-apt-repository ppa:gns3/ppa



အဲလို PPA channel ကို add ပြီး install လုပ်လိုရသလို .. **add-apt-repository** မရရင်လဲ..

/etc/apt/source.list မှာ သွားပြီး repo ကို သွားထည့်လိုရပါတယ်။ GNS3 ရဲ့ Document မှာလဲ ရေးဇာတ်ပါတယ်။ အဲလိုမှာဟုတ် ppa ထည့်ချင်တယ် add-apt-repository လဲ မရရင် ဘာထပ်လုပ်လို ရမလဲ ဆိုတော့ကာ..

12.3.1 linux user to get ppa support

sudo apt-get install software-properties-common

And/OR

sudo apt-get install python3-software-properties

And/OR

sudo apt-get install python-software-properties

တို့ကို သုံးပြီးတော့ PPA channel ကို ထည့်လိုရအောင်လဲ လုပ်လိုရပါတယ်။



<http://crossnetmm.com/>

A screenshot of an Ubuntu desktop environment. A terminal window is open in the foreground, showing the command `sudo add-apt-repository ppa:gns3/ppa` being run. The output indicates that a PPA for GNS3 and Supporting Packages is being added, with instructions to press [ENTER] to continue or Ctrl-c to cancel. The terminal also shows the process of updating package lists, with various package details like URLs and file sizes listed. In the background, the Unity desktop interface is visible with its characteristic dock and icons.

```
Activities Terminal ▾ Wed 04:02
spacex@ubuntu:~$ sudo add-apt-repository ppa:gns3/ppa
[sudo] password for spacex:
PPA for GNS3 and Supporting Packages. Please see http://www.gns3.com for more details
More info: https://launchpad.net/~gns3/+archive/ubuntu/ppa
Press [ENTER] to continue or Ctrl-c to cancel adding it.

Ign:1 http://dl.google.com/linux/chrome/deb stable InRelease
Get:2 http://dl.google.com/linux/chrome/deb stable Release [943 B]
Get:3 http://dl.google.com/linux/chrome/deb stable Release.gpg [819 B]
Get:4 http://ppa.launchpad.net/gns3/ppa/ubuntu bionic InRelease [15.3 kB]
Hit:5 http://security.ubuntu.com/ubuntu bionic security InRelease
Hit:6 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Hit:7 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:8 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:9 http://ppa.launchpad.net/kokoye2007/ppa/ubuntu bionic InRelease
Get:10 http://ppa.launchpad.net/gns3/ppa/ubuntu bionic/main i386 Packages [1,236 B]
Get:11 http://ppa.launchpad.net/gns3/ppa/ubuntu bionic/main amd64 Packages [1,736 B]
Get:12 http://ppa.launchpad.net/gns3/ppa/ubuntu bionic/main Translation-en [836 B]
Reading package lists... 9%
```

DON'T CRY FOR SOMEONE WHO KILLED YOUR SMILE!!

sudo apt-get install gns3-gui

ဆိုပြီး GNS3 ကို install လုပ်ပေးလိုက်ပါ။

A screenshot of a terminal window on an Ubuntu system. The user has run the command `sudo apt-get install gns3-gui`. The terminal shows the progress of the package installation, including reading package lists, building dependency trees, and determining state information. It then lists the additional packages that will be installed, which include various networking and graphical components required for GNS3. The terminal interface is dark-themed, and the output text is white.

```
es Terminal ▾ Wed 04:04
spacex@ubuntu:~$ sudo apt-get install gns3-gui
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  augeas-lenses bridge-utils cpu-checker cpulimit dmeventd dynamips ebtables gns3-server
  ibverbs-providers ipxe-qemu ipxe-qemu-256k-compat-efl-roms libaio1 libaugeas0 libavahi-gobject0
  libe-ares2 libcacard0 libdevmapper-event1.02.1 libdouble-conversion1 libfdt1 libgtk-vnc-2.0-0
  libgvnc-1.0-0 libibverbs1 libiscsi7 liblua5.2-0 liblvm2app2.2 liblvm2cmd2.02 libnetcf1
  libnl-route-3-200 libphodav-2.0-0 libphodav-2.0-common libqgsttools-p1 libqt5core5a libqt5dbus5
  libqt5designer5 libqt5equipe5 libqt5help5 libqt5multimedia5 libqt5multimedia5-plugins
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  augeas-lenses bridge-utils cpu-checker cpulimit dmeventd dynamips ebtables gns3-server
  ibverbs-providers ipxe-qemu ipxe-qemu-256k-compat-efl-roms libaio1 libaugeas0 libavahi-gobject0
  libe-ares2 libcacard0 libdevmapper-event1.02.1 libdouble-conversion1 libfdt1 libgtk-vnc-2.0-0
  libgvnc-1.0-0 libibverbs1 libiscsi7 liblua5.2-0 liblvm2app2.2 liblvm2cmd2.02 libnetcf1
  libnl-route-3-200 libphodav-2.0-0 libphodav-2.0-common libqgsttools-p1 libqt5core5a libqt5dbus5
  libqt5designer5 libqt5equipe5 libqt5help5 libqt5multimedia5 libqt5multimedia5-plugins
```



Configuring wireshark-common

Dumpcap can be installed in a way that allows members of the "wireshark" system group to capture packets. This is recommended over the alternative of running Wireshark/Tshark directly as root, because less of the code will run with elevated privileges.

For more detailed information please see /usr/share/doc/wireshark-common/README.Debian.

Enabling this feature may be a security risk, so it is disabled by default. If in doubt, it is suggested to leave it disabled.

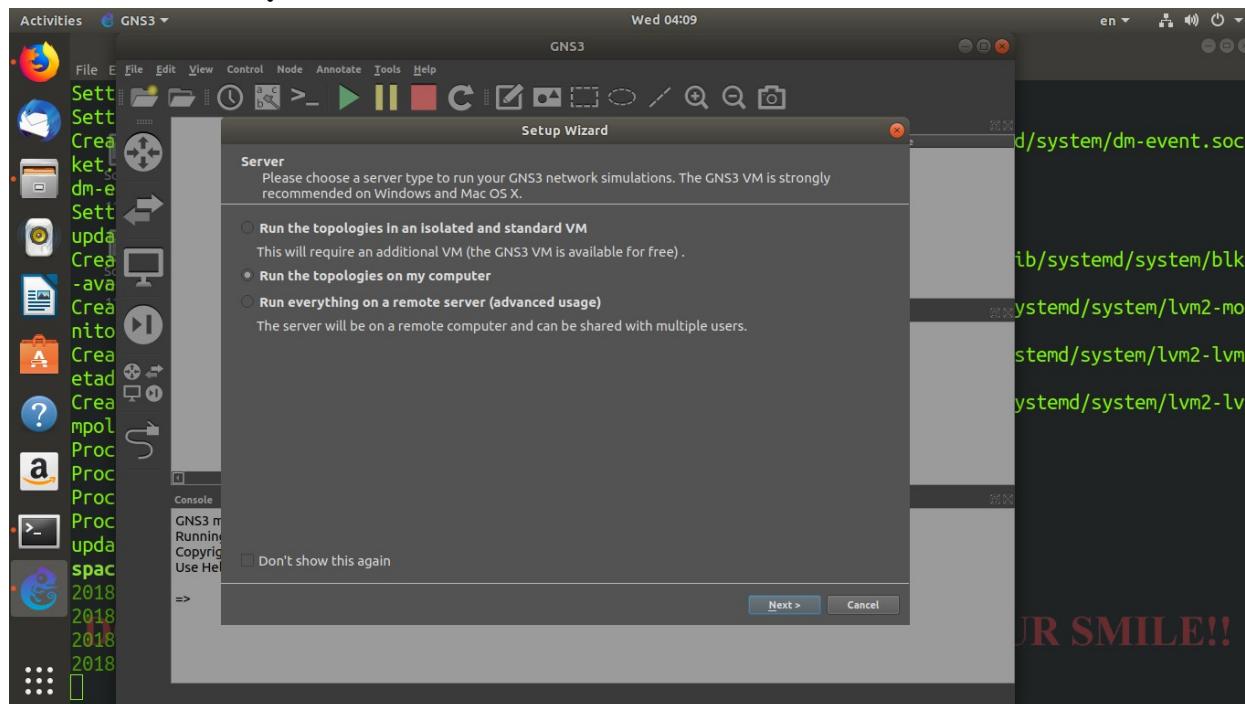
Should non-superusers be able to capture packets?

<Yes> <No>

Root မဟုတ်တဲ့ non-superuser အတွက် Yes ကို ရွေးပေးပါ။ အဲဒေါက်မှာတွေ့။ Terminal ကနေပြီးတွေ့ \$**gns3** ဆိုပြီး run လိုက်ရင်

```
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Processing triggers for systemd (237-3ubuntu10.9) ...
Processing triggers for ureadahead (0.100.0-20) ...
Processing triggers for initramfs-tools (0.130ubuntu3) ...
update-initramfs: Generating /boot/initrd.img-4.15.0-20-generic
spacex@ubuntu:~$ gns3
2018-11-21 04:09:19 INFO root:126 Log level: INFO
2018-11-21 04:09:19 INFO main:259 GNS3 GUI version 2.1.11
2018-11-21 04:09:19 INFO main:260 Copyright (c) 2007-2018 GNS3 Technologies Inc.
2018-11-21 04:09:19 INFO main:262 Application started with /usr/bin/gns3
```

GNS3 တက်လာပါလိမ့်မယ်။



12.3.2 install GNS3 with pip3

ဟုတ်ပြီး အဲလို PPA add ။။ source list ထည့်တော့ မလုပ်ပဲ **pip** ကိုသုံးပြီး **install** လုပ်တဲ့ နည်းကို ပြုမယ်။ Pip ကို သုံးမယ်ဆို သူက python/python3 run လို့ရတဲ့ စက်တွေမှာ ဆို ထည့်လို ရတယ်။ ဂျွန်တို့ကတော့ ခုပြောတဲ့ နည်းကိုသုံးဖြစ်တာ ပိုများတယ်။ တစ်ခုမှတ်ထားရမှာက အပေါ်မှာ ပြောခဲ့တာသည် ppa/source တွေ ထည့်ပြီး GNS3 ကို သွင်းတာ။

ခုထပ်ပြောမှာက pip3 ကိုသုံးပြီးတော့ GNS3 ကို install လုပ်မှာ။ ဘယ်နည်းနဲ့ ရရ GNS3 ဖွင့်လို့ရရင် ရပြီး။ ရနေတာကိုမှ အကုန်လုံးစမ်းစရာ မလိုဘူးနော်။ apt-get ။။ install လုပ်ထားတဲ့ GNS3 ကို ပြန်ဖြတ်ချင်ရင် **apt-get remove gns3-gui** နော်..

ကဲ စမယ်.. Pip3 ကို သုံးပြီး gns3 ကို install လုပ်ရမယ့်။ **sudo apt-get install python3 python3-pip** ဆိုပြီး pip3 ကို install လုပ်လိုက်ပါ။

```
spacex@ubuntu:~$ sudo apt-get install python3
[sudo] password for spacex:
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3 is already the newest version (3.6.5-3).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
spacex@ubuntu:~$ sudo apt-get install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

ဒီနေရာမှ တစ်ခုသိထားရမှာက pip သည် python (2.7) ။။ run ပြီး pip3 ဆိုတဲ့ အတိုင်း Python (3.4 နဲ့ အထက်) လို အပ်ပါတယ်။ အဲနောက် pip3 ကို သွင်းပြီးသွားပြီဖြစ်တဲ့ အတွက် pip ကိုသုံးပြီးတော့ gns3-gui ကို install လုပ်ပါ။ အဲနောက် gns3-server ကို install လုပ်ပါ။

သုံးရမယ် command ။။ ။။။ .. >

GNS3 GUI (client)

sudo pip3 install gns3-gui (အဲလိုရှိက်ရင် နောက်ဆုံး ထွက်တဲ့ version ကိုသွင်းပေးသွားမှာပါ)
sudo pip3 install gns3-gui==2.1.5 (ဒီလိုရှိက်ရင်တော့ ကိုယ်သွင်းစေချင်တဲ့ version ကို သွင်းပေးသွားမှာပါ) **ကြိုက်တဲ့** တစ်ခုကို သုံးပါ။

```
Setting up python3-dev (3.6.5-3) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
spacex@ubuntu:~$ sudo pip3 install gns3-gui==2.1.5
The directory '/home/spacex/.cache/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
The directory '/home/spacex/.cache/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. Check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting gns3-gui==2.1.5
  Downloading https://files.pythonhosted.org/packages/55/6d/154ccfa1090dded022cd7b5105d1b96af92eb3198170cc38b220d76ae76e/gns3-gui-2.1.5.tar.gz (4.7MB)
    30% |██████████| 1.4MB 9.4MB/s eta 0:00:01
```

GNS3 Server (background)

`sudo pip3 install gns3-server` (အဲလိုရှိရင် နောက်ဆုံး ထွက်ထဲ version ကိုသွင်းပေးသွားမှာပါ)
`sudo pip3 install gns3-server==2.1.5` (ဒီလိုရှိရင်တော့ ကိုယ်သွင်းစေချင်ထဲ version ကို သွင်းပေးသွားမှာပါ) **ကြိုက်** ထဲ တစ်ခုကို သုံးပါ။

ပြီးတော့ GNS3 VM မသုံးပဲ ကိုယ်လိုတာတွေပဲ (qemu, iouyap, ubridge, dynamips...). ထည့်သုံးမယ်ဆိုလဲ သုံးလို့ရ တယ်။

pip/pip3 ဆိတာက python programming သမားတွေဆိုရင်တော့နားလည်တယ်။ python packageတော့ install လုပ်လိုက်တာပဲ။ GNS3 တစ်ခု လုံးကလဲ pythonနဲ့ ရေးထားပဲလေ။



```
kB)          100% |██████████| 286kB 3.1MB/s
Installing collected packages: jsonschema, psutil, raven, gns3-gui
  Running setup.py install for psutil ... done
  Running setup.py install for gns3-gui ... done
Successfully installed gns3-gui-2.1.5 jsonschema-2.6.0 psutil-5.4.5 raven-6.8.0
spacex@ubuntu:~$ sudo pip3 install gns3-server==2.1.5
The directory '/home/spacex/.cache/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
The directory '/home/spacex/.cache/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting gns3-server==2.1.5
  Downloading https://files.pythonhosted.org/packages/7e/bf/7d3f1f62f223e64f4396e53ec05b6a5b805ded130a755e53563e7717b1b1/gns3-server-2.1.5.tar.gz (1.5MB)
    100% |██████████| 1.5MB 738kB/s
Collecting Jinja2>=2.7.3 (from gns3-server==2.1.5)
  
```

သူက repository ထည့်တာ/source.list ထည့်တာနဲ့တော့ မတူဘူး.. ထည့်ပြီး run လိုက်ရင် လိုနေထဲ python package လေးတွေ ရှိနေတက်တယ်။ အဲလိုနေတာလေးတွေကိုလဲ pip3 ကိုသုံးပြီး ထည့်လိုက်ရှုံးပဲ။ ;-)

12.3.3 Common GNS3 error with pip

```
spacex@ubuntu:~$ gns3
Fail update installation: No module named 'sip'
Can't import Qt modules: Qt and/or PyQt is probably not installed correctly...
spacex@ubuntu:~$ sudo apt-get install sip
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package sip
spacex@ubuntu:~$ sudo pip3 install sip
The directory '/home/spacex/.cache/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
The directory '/home/spacex/.cache/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting sip
  Downloading https://files.pythonhosted.org/packages/8a/ea/d317ce5696dda4df7c156cd60447cda22833b38106c98250eae1451f03ec/sip-4.19.8-cp36-cp36m-manylinux1_x86_64.whl (66kB)
    100% |██████████| 71kB 408kB/s
Installing collected packages: sip
Successfully installed sip-4.19.8
  
```

```

Successfully installed sip-4.19.8
spacex@ubuntu:~$ gns3
Fail update installation: No module named 'PyQt5'
Can't import Qt modules: Qt and/or PyQt is probably not installed correctly...
spacex@ubuntu:~$ sudo pip3 install pyqt5
The directory '/home/spacex/.cache/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
The directory '/home/spacex/.cache/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting pyqt5
  Downloading https://files.pythonhosted.org/packages/e4/15/4e2e49f64884edbab6f833c6fd3add24d7938f2429aec1f2883e645d4d8f/PyQt5-5.10.1-5.10.1-cp35.cp36.cp37.cp38-abi3-manylinux1_x86_64.whl (107.8MB)
    67% |██████████| 73.0MB 9.4MB/s eta 0:00:04

```

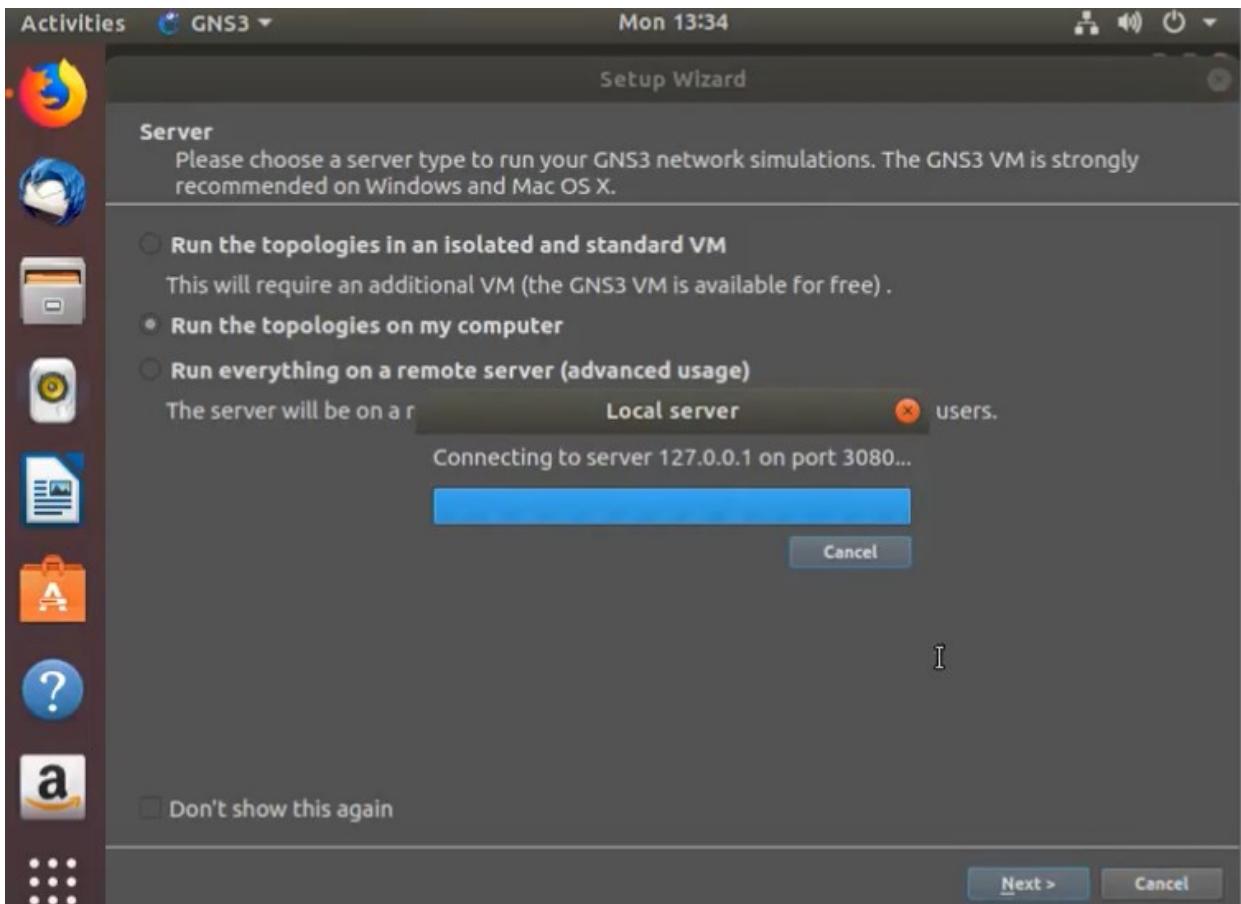
လိုနေတဲ့ package တွေဖြစ်တဲ့ sip & PyQt5 တွေ၊ ထည့်ပြီးသွားပြီ ဆိတော့ ပြန် run ကည့်ပါ။ GNS3 ကို.....

```

spacex@ubuntu:~$ gns3
2018-05-21 13:34:24 INFO root:126 Log level: INFO
2018-05-21 13:34:24 INFO main:259 GNS3 GUI version 2.1.5
2018-05-21 13:34:24 INFO main:260 Copyright (c) 2007-2018 GNS3 Technologies Inc
.
2018-05-21 13:34:24 INFO main:262 Application started with /usr/local/bin/gns3

```

ကဲ လိုနေတာမရှိတော့ဘူးဆိုရင်တော့ ရွောရွော မွှေ့ မွှေ့ပြီးသွားပြီပေါ့။ တစ်ခါတစ်လေ ကိုယ်သုံးတဲ့ linux desto မှာ တစ်ခြား သူမှားနဲ့ မတူတဲ့ error မျိုး တတ်တက်တယ်။ အဲတော့ရာရွောကြပါ။ လုပ်ပြီး သူ ဆိုင်ရာ ဆိုင်ရာ community တွေ site တွေမှာ မွေးနှုန်းဖော်ရတာပေါ့။ အတွေအကြံ့ ရပါတယ်။ စိတ်မပျက်ပဲ စမ်းကြာည့်ရာပါ။ မရရင်လဲ လာမေးလို့ရပါတယ်။



က တစ်ခုတွော တွောပြီးသွားပြီး.. GNS3 VM နဲ့တွဲပြီး lab လုပ်ဖိအတွက် VMware workstation/player လိပ်တ ။..VMware Player ။။။။ commercial use မဟုတ်ရင် free သုံးလို့ရတွော အဆင်ပြေမှာပါ။ VMware Workstation ကို crack ရှုရင်လဲ သုံးချင်သုံးဖော့။

12.4. Get virtual machine

ခုခံမှာတွော... VMware player ကို သွင်းပြထားပါတယ်။ vmplay သည် commercial use မဟုတ်ရင် free သုံးလို့ ရပါတ ယ်။

>>> Download Link <<

VMware Workstation Pro >> <https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html>

VM Player >> <https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html>



vmware-
player-14.1.2-8
497320.exe



VMware-
Player-14.1.2-8
49732...undle

Download ရလာမည့် linux အတွက် install လုပ်ရမည့် file သည် *.bundle နဲ့ ဆုံးပါတယ်။ သူကို executable permission ပေးပြီး execute လုပ်လိုက်ပါ။ အောက်မှာ လုပ်ပြထားပါတယ်။

Sudo chmod a+x VMware-Player-14.1.2-8497320.x86_64.bundle

Sudo ./VMware-Player-14.1.2-8497320.x86_64.bundle

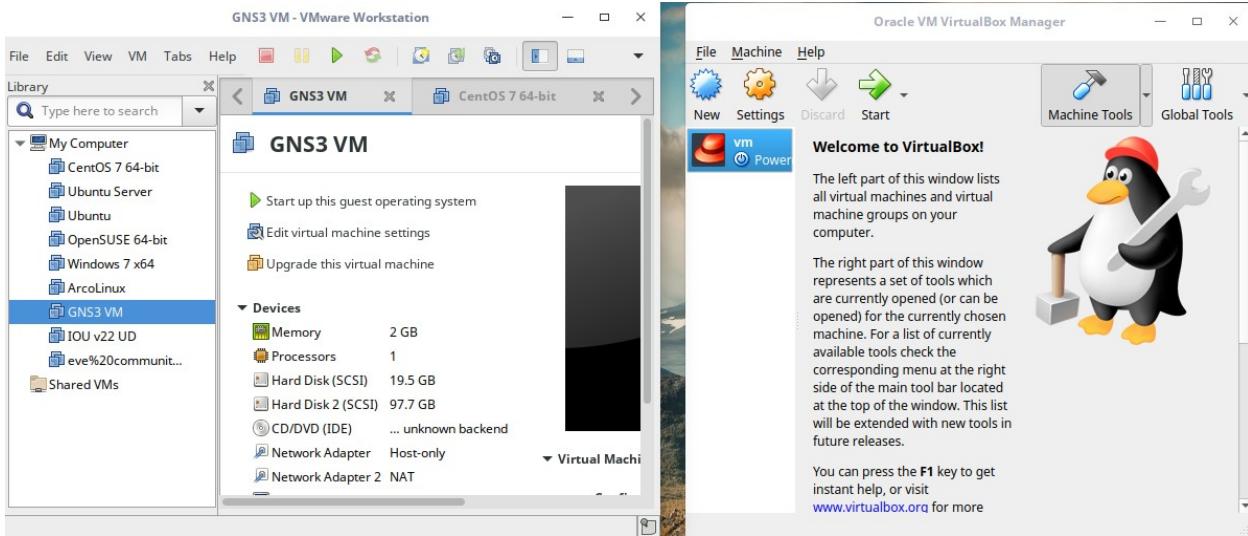
Or

Sudo bash VMware-Player-14.1.2-8497320.x86_64.bundle

တစ်ခုမှတ်ထားရမှာက ကိုယ် VMware version 12 သွင်းလို့ error တက်ရင် version 14 ထို့ 15 ထိုနဲ့ စမ်းပြောသွားပါ။ အဲတာမ မရရင်တော့ တက်တဲ့ error ကို ရှာပြီး FIX လုပ်ပါ။ ဖြစ်တက်တဲ့ error အများစုက ရှာလိုက်ရင် အဖြေဖော်ပါတယ်။

ခုအောက်က ပုံကတော့ virtualbox ကို သွင်းပြထားတာပါ။

```
spacex@ubuntu: ~
File Edit View Search Terminal Help
spacex@ubuntu:~$ sudo apt-get install virtualbox
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
dkms libdouble-conversion1 libgsoap-2.8.60 libqt5core5a libqt5dbus5
libqt5gui5 libqt5network5 libqt5opengl5 libqt5printsupport5 libqt5svg5
libqt5widgets5 libqt5x11extras5 libsdl1.2debian libvncserver1
libxcb-xinerama0 qt5-gtk-platformtheme qttranslations5-l10n virtualbox-dkms
virtualbox-qt
Suggested packages:
menu qt5-image-formats-plugins qtwayland5 vde2
virtualbox-guest-additions-iso
The following NEW packages will be installed:
dkms libdouble-conversion1 libgsoap-2.8.60 libqt5core5a libqt5dbus5
libqt5gui5 libqt5network5 libqt5opengl5 libqt5printsupport5 libqt5svg5
libqt5widgets5 libqt5x11extras5 libsdl1.2debian libvncserver1
libxcb-xinerama0 qt5-gtk-platformtheme qttranslations5-l10n virtualbox
virtualbox-dkms virtualbox-qt
0 upgraded, 20 newly installed, 0 to remove and 0 not upgraded.
Need to get 36.6 MB of archives.
After this operation, 158 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu bionic/main amd64 libdouble-conversio
n1 amd64 2.0.1-4ubuntu1 [33.0 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu bionic/main amd64 libqt5core5a amd64
5.9.5+dfsg-0ubuntu1 [2,035 kB]
1% [2 libqt5core5a 82.3 kB/2,035 kB 4%]
```



12.5 Get Gns3vm

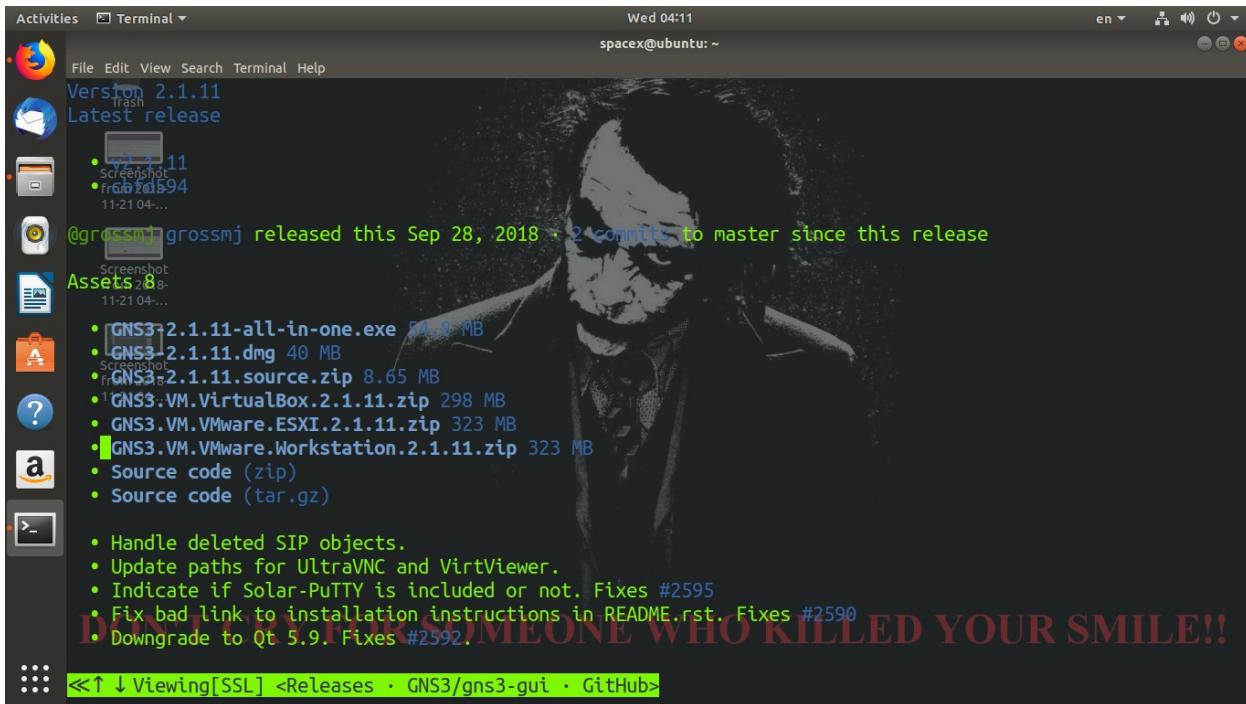
အဲနောက်မှာတော့ နောက်ထပ် လိုနေတာဖြစ်တဲ့ GNS3 VM ကို ဒေါင်းပါ။ ကိုယ်က VMware သုံးမှာလား.. Virtual box သုံးမှာလား..

ESXI သုံးမှာလား.. သုံးမထုတ်ကောင်အတွက် လိုအပ်တဲ့ GNS3 VM တစ်ခုကို ပဲ ဖောင်းပါ။။ တစ်ခုမှတ်ထားပြီးနော်.. GNS3 VM version သည် GNS3 version နဲ့ တူမှ အဆင်ပြေမှာပါ။

Download link >> <https://github.com/GNS3/gns3-gui/releases>

See by Terminal Browser ..

<http://crossnetmm.com/>



GNS3 VM (virtualbox / VMware) download link >> <https://github.com/GNS3/gns3-gui/releases>

ခုလုပ်မထဲ labတွေ CCNA တန်းမှာ သင်တာတွေကတွော Virtual Box နဲ့ ဆိုလဲ ပြောတယ် ပြသနာမရှိဘူး။

Seen by Web Browser ..

Latest release

↳ v2.1.11
↳ cbfd594

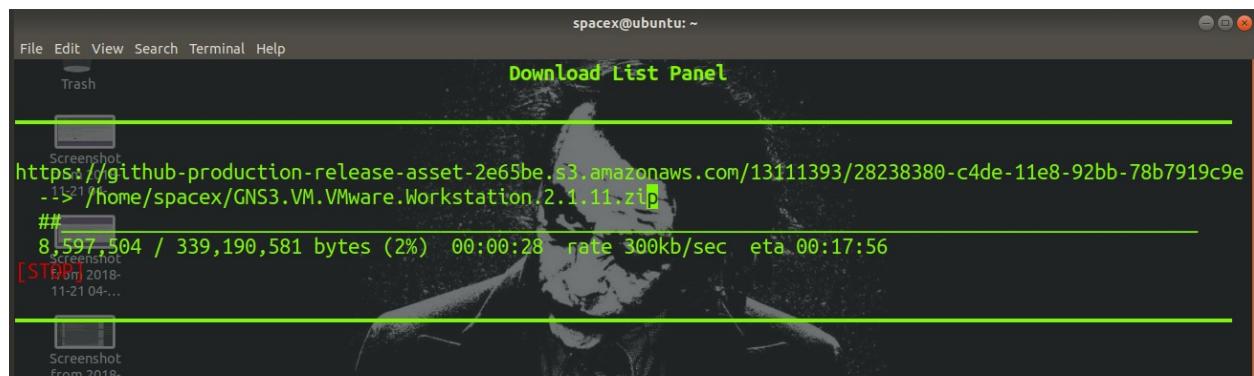
Version 2.1.11

 grossmj released this on Sep 29 · 2 commits to master since this release

Assets 8

 GNS3-2.1.11-all-in-one.exe	54.8 MB
 GNS3-2.1.11.dmg	40 MB
 GNS3-2.1.11.source.zip	8.65 MB
 GNS3.VM.VirtualBox.2.1.11.zip	298 MB
 GNS3.VM.VMware.ESXI.2.1.11.zip	323 MB
 GNS3.VM.VMware.Workstation.2.1.11.zip	323 MB
 Source code (zip)	
 Source code (tar.gz)	

VMware လား Virtualbox လား ရွေးပါ။ ပြီးတွေ့GNS3 Version တူတော် အောင်းပါ။

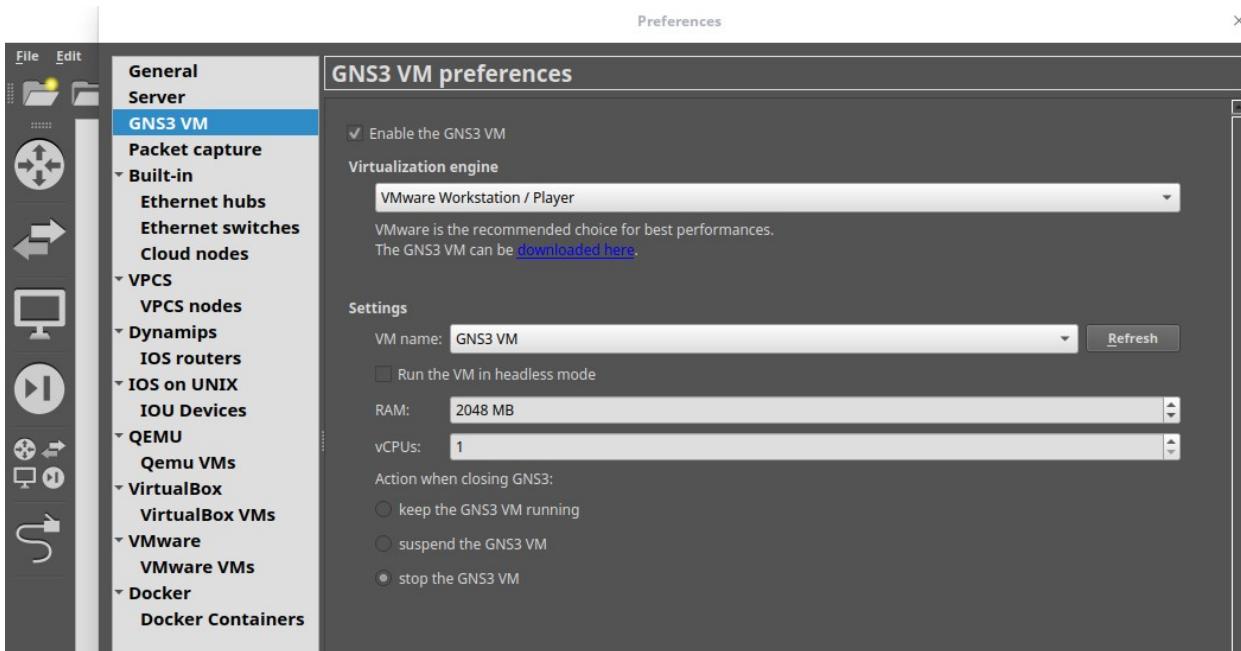


12.6 Connect Gns3 with Gns3vm

GNS3 မှုGNS3 VM နိုင်တွေ့ဖို့ VM ကို အရင်ဖွင့်ထားတာ အကောင်းဆုံးပါပဲ။



ဤမှ GNS3 ကို run ပါ။ Setup Wizard မှာ တစ်ခါတည်းရွေးပေးလို့ရသလို ...GNS3 menu ထဲက Edit >> Preferences မှာ လဲ သွားပြီး ရွေးပေးလို့ရပါတယ်။



ឧប ទៅ install លើកបង្កើតនៃ setup wizard ការណែនាំ លើកបង្កើតនៃកម្មវិធី។ Preferences ការណែនាំ នឹងបង្កើតឡាយការណែនាំ នៃកម្មវិធី។ និងការណែនាំ នឹងបង្កើតឡាយការណែនាំ នៃកម្មវិធី។

Setup Wizard

GNS3 VM
In order to run the GNS3 VM you must first have VMware or VirtualBox installed and the GNS3 VM.ova imported with one of these software.

Virtualization software:

- VMware (recommended)
- VirtualBox



If you don't have the GNS3 Virtual Machine you can [download it here](#). And import the VM in the virtualization software and hit refresh.

VM name: Refresh

vCPU cores:

RAM size:

[« Back](#) [Next »](#) [Cancel](#)

VMware လား VirtualBox လားရွေးပေးလိုက်ရင် auto GNS3 VM ကို select ဖိတယ်။

12.7 Get cisco Device image file (IOS/IOU)

ထိုတွေကို နောက်ထပ် မေးလာမှာ လုပ်ရမှာက IOSတွေIOUတွေ စတာတွေကို ဘယ်လို run မလဲ ပေါ့။ GNS VM သုံးရင် သူပေါ်မှာ run တာ အကောင်းဆုံးပါပဲ။

IOSတွေIOUတွေ ဒေါင်းစုံ လင့်ပါ။

IOU (IOS On Unix) >> <http://www.networklab.in/downloads/>

နောက်တစ်ခက်တွာ SourceForge ကနေ တင်ပေးထားတဲ့ GNS3 နဲ့ ပါတာသက်တဲ့ Appliance တွေ Fileတွေ ပါ။

<https://sourceforge.net/projects/gns-3/files/>



Hands On Lab Environment Fundamentals – Tips and Tricks

GNS3 »

HP | HCL »

HP | HNS »

Huawei | ENSP »

VMware »

VirtualBox »

CISCO | IOU »

Checkpoint | GAIA »

Download >> <https://www.gns3.com/software/download>
Support >> <https://www.gns3.com/support/docs/>
Appliances >> 1. Cisco IOS Images 2. IOU VirtualBox VM 3. L2 & L3 IOU Image 4. [IOU Official Download Page](#)

ခု စစ်ဆေး အမိက လိုနေတာကတွေ့wa IOU (IOS On Unix) ပါ။ အဲတာကို အရင်ဖောင်းလိုက်ပါ။ အောက်မှာ ပြထားသလို import လုပ် လိုက်ပါ။ nexte next okay ပါပဲ.. အရမ်းမဆင်ပါဘူး...

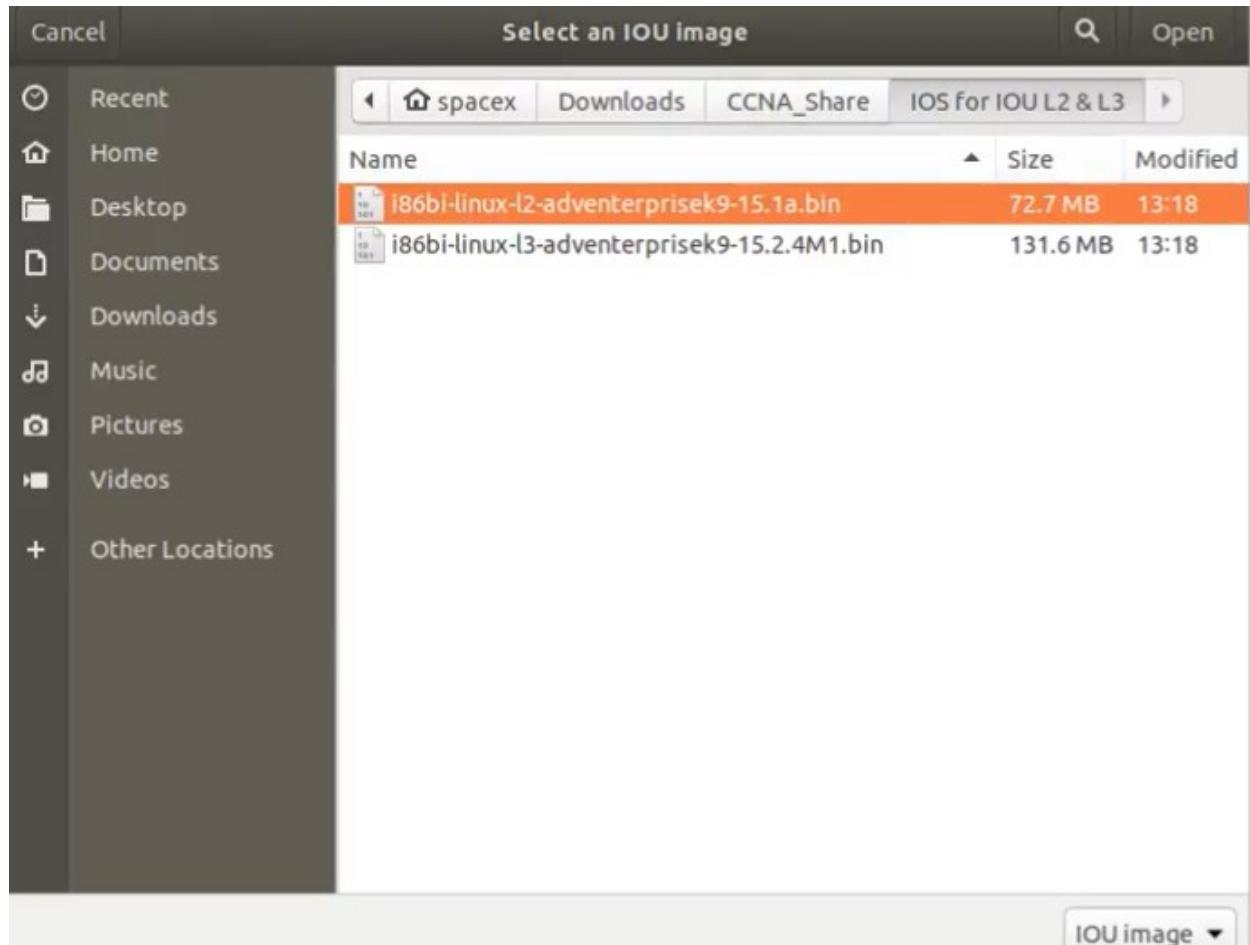


GNS3 VM ပေါ်မှာ run သို့ ဖြောလိုက်ပါ။ အဲလိုမှ မဟုတ် ကိုယ့် စက် local OS မှာ မှ run ချင်ရင်တွော တစ်ခြားလိုဘာက တွေ ဖြစ်ထဲIOU support ရအောင် သွင်းပေးဖိုလိုအပ်ပါတယ်။

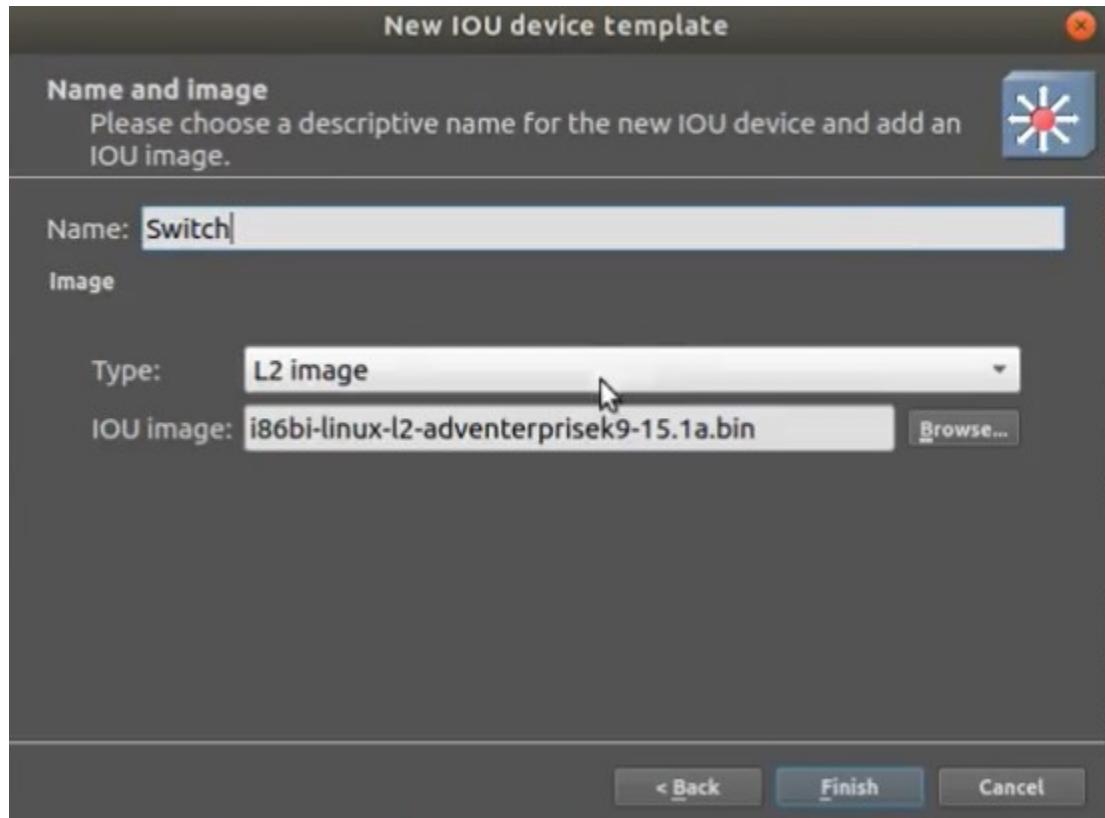
If_you_want_IOU_support

```
sudo dpkg --add-architecture i386
sudo apt-get update
sudo apt-get install gns3-iou
```

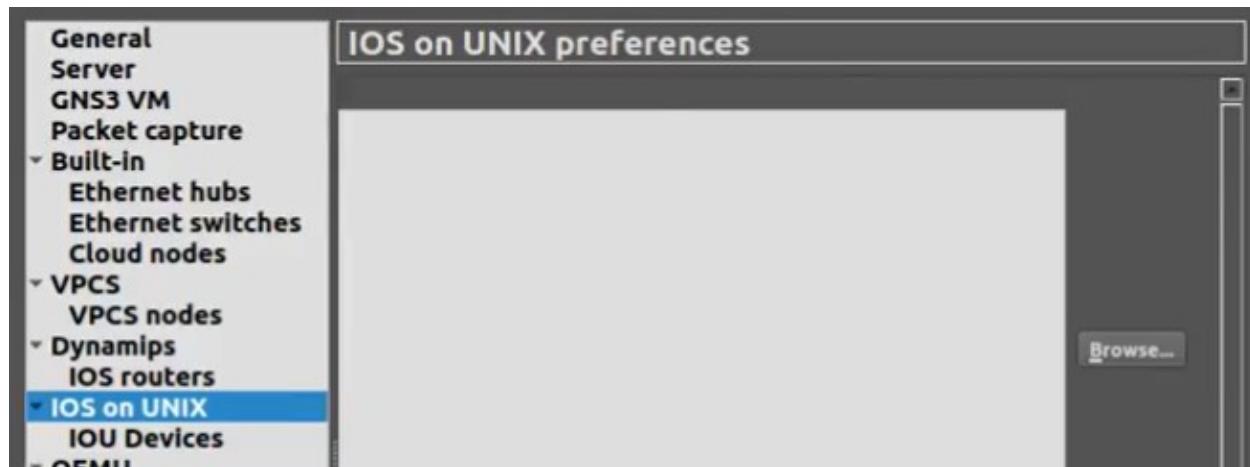
<http://crossnetmm.com/>



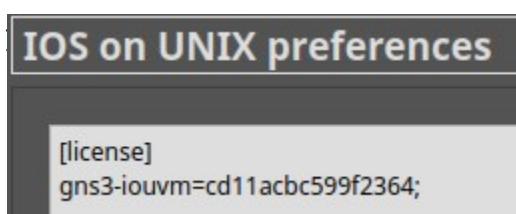
IOU binary ဖိုင် ကို ရွေးပေးပါ။ L2 ဆိတာသည့် layer 3 image ဖြစ်ပြီး L3 မှာ မှာ Layer 3 image ပါ။



နာမည်ပေးပါ။ L2/L3 ရွေးပေးပြီး finished နိုင်ပါ။



12.7.1 IOU license



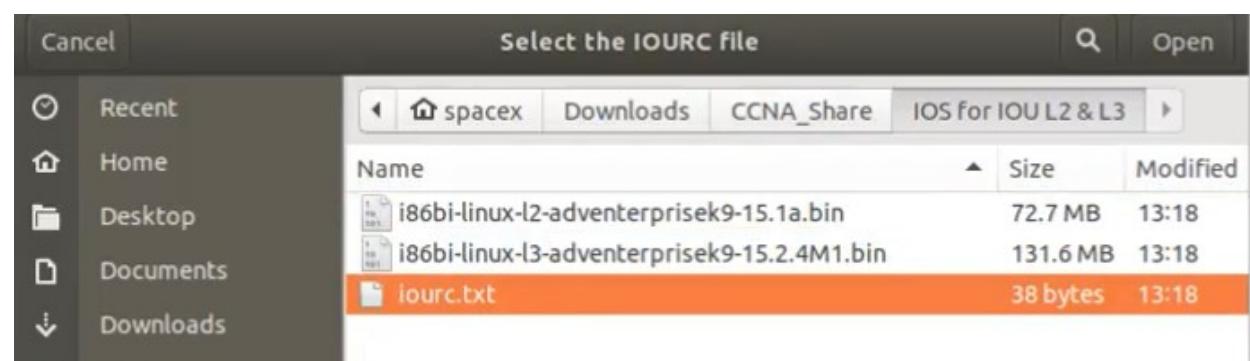
ထွေ ထည့်ပြီးသွားရင် နောက်ထပ်လိုတာက IOU အတွက် license ရယ် အဲ license မှာ ရှိတဲ့ နာမည် အတိုင်း GNS3 VM ကို ပြောင်းပေး

ဖို့ရင်လိုပါတယ်။ ပေးထားတဲ့ ထဲမှာ iourc.txt ဆိတာလေး ပါပါတယ်။ အဲကောင်ကို Browse လုပ်လိုက်ပါတယ်။ ခုဘေးမှာ ပြတဲ့ ပုံအတိုင်းရ ပါလိမ့်မယ်။

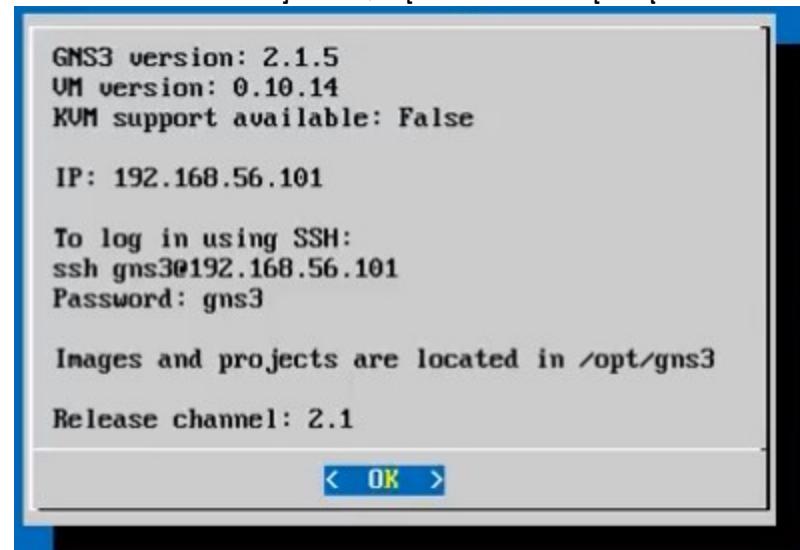
License error တက်ရင် python license generator လေးကိုသုံးပြီးတော့ ထုတ်လိုခဲ့ပါတယ်။

<https://gist.githubusercontent.com/paalte/8edd82f780c650ae2b4a/raw/bd7b6b8a81c338359e6de4ff0ed0def9f7dc9146/CiscoKeyGen.py>

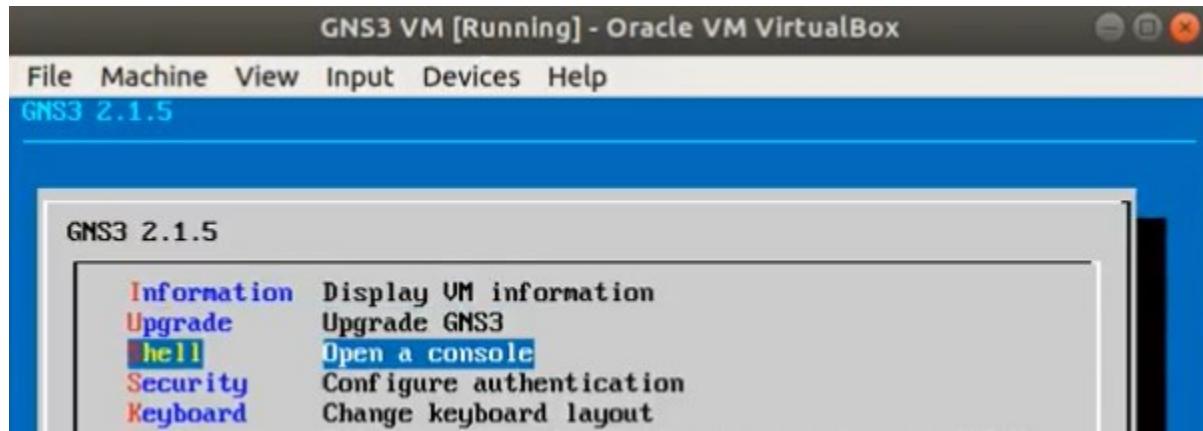
IOU နဲ့ script ကို directory အတူတူထားပြီး script ကို execute လုပ်ယုံပါပဲ။



အောင်အောင် GNS3 VM မှာ okay ကို enterခဲ့ကိုလိုက်ရင် Menuလေးရှာ လာပါလိမ့်မယ်။



Shell ကို ရွေးလိုက်ပါ။ Terminal တစ်ခု ရှာလာပါလိမ့်မယ်။

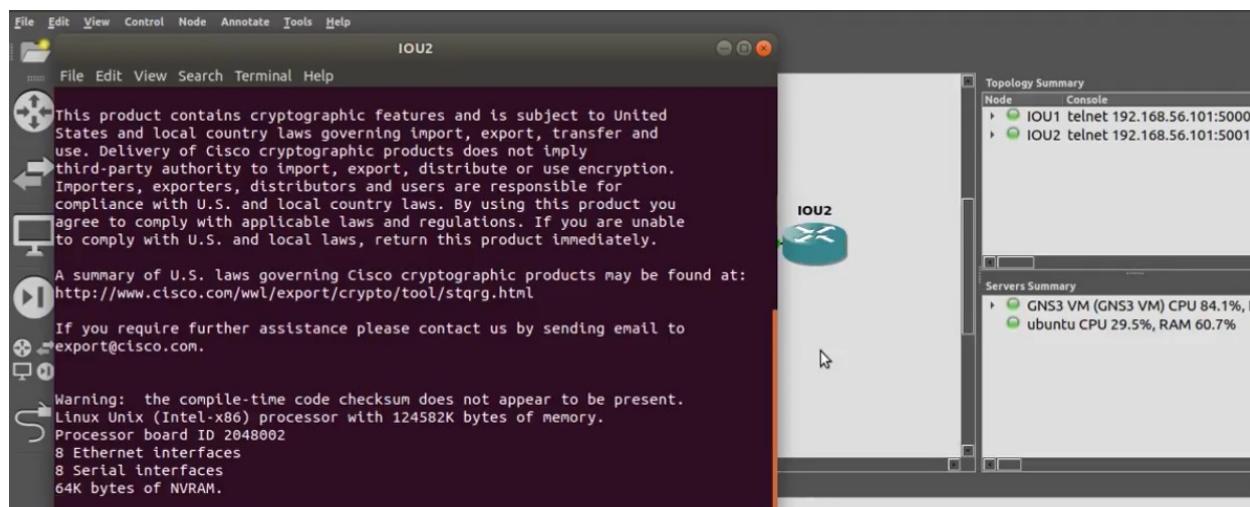


>> sudo hostnamectl set-hostname gns3-iouvm

လို့ နာမည်ပြောင်းပေးလိုက်ပြီး Bash ကို ရှိလိုက်ပါ။ အဲတာဆို ရပါပြီ။

```
gns3@gns3vm:~$  
gns3@gns3vm:~$  
gns3@gns3vm:~$  
gns3@gns3vm:~$ sudo hostnamectl set-hostname gns3-iouvm  
gns3@gns3vm:~$ bash  
gns3@gns3-iouvm:~$  
gns3@gns3-iouvm:~$  
gns3@gns3-iouvm:~$
```

ခုခွဲရင်တွေ့ GNS3 ကို GNS3 VM နဲ့ တွေ့သုံးပြီး IOUတွေနဲ့ ကောင်းကောင်း LAB လုပ်နိုင်ပါပြီး



12.8. Get Python Container & Ubuntu Docker image (GNS3 Appliance)

နောက်ထပ် Additional အနေနဲ့ လိုအပ်တာတွေကတွေ Ubuntu Docker image မှာ Python ပါ။ ဖွဲ့ကောင်တွေကို ခေါင်းမယ်ဆိုရင်တွေ့ဘာ။

>>> <https://gns3.com/marketplace/appliances>

အဲကနေ ရှာလိုက်လိုရပါတယ်။ GNS3 စမ်းပို Contribute လုပ် ပေးထားတဲ့ သူတွေ အများပြီးပါ။

GNS3 VM မသုံးဘဲ local OS မှာ Docker support ရ ရှင်ရင်တွေ့ GNS3 official doc ထဲက အတိုင်းပေါ့။

If you want Docker support

Remove any old versions:

```
sudo apt-get remove docker docker-engine docker.io
```

Install these packages, if not already present:

```
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common
```

Import the official Docker GPG key:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
add -
```

Setup the Docker repo:

```
sudo add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) \
    stable"
```

<http://crossnetmm.com/>

NOTE There currently isn't a package for Ubuntu 18.04 in the stable release channel, but there is in the edge channel.

Update the metadata:

sudo apt-get update

Install Docker:

sudo apt-get install docker-ce

Finally, add your user to the docker group, and restart your user session

```
sudo usermod -a -G docker username
```

NOTE_1 If you encounter any permission errors when trying to run GNS3, DynaMIPS devices, Wireshark, or Qemu VMs, ensure your user belongs to the following groups:

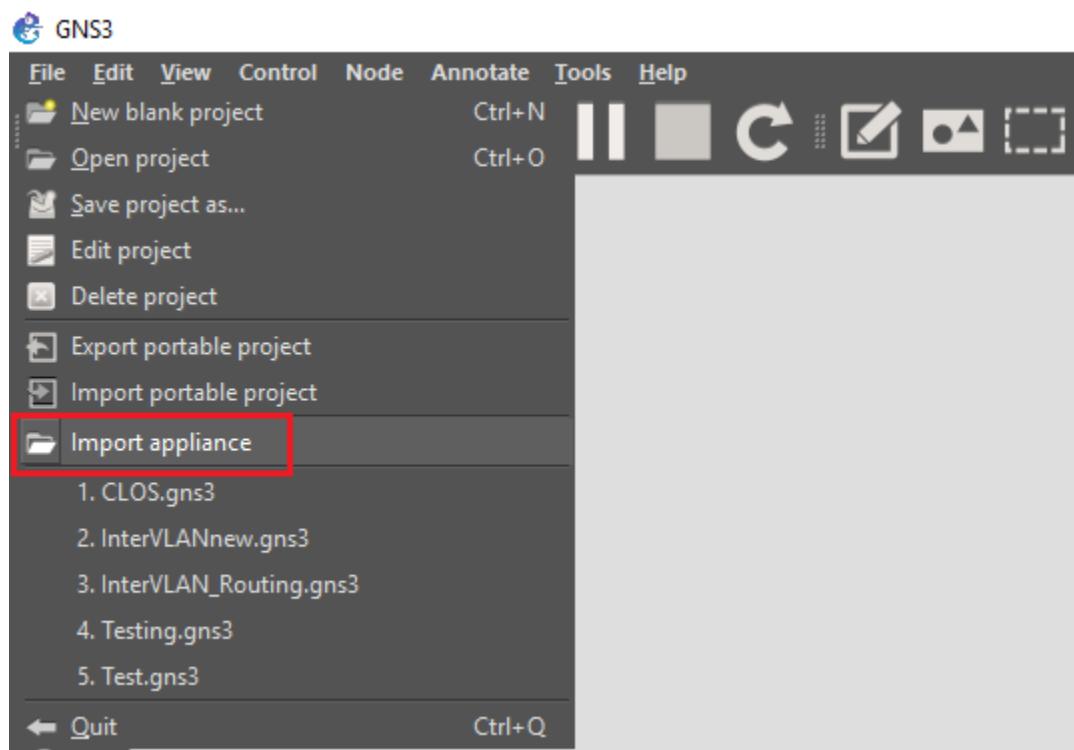
ubridge libvirt kvm wireshark

GNS3 VM နဲ့ သုံးရွာမယ်သူတွေကတော့ အပေါ်က ပြဿနာမရှိဘူးပေါ့.. ကဲ ဒဲတာဆို gns3 appliance လေးတွေကို အောင်လိုက်ပါ။

*.gns3a නේ පාඨම්පත් වලදී පිළිගුණයි॥

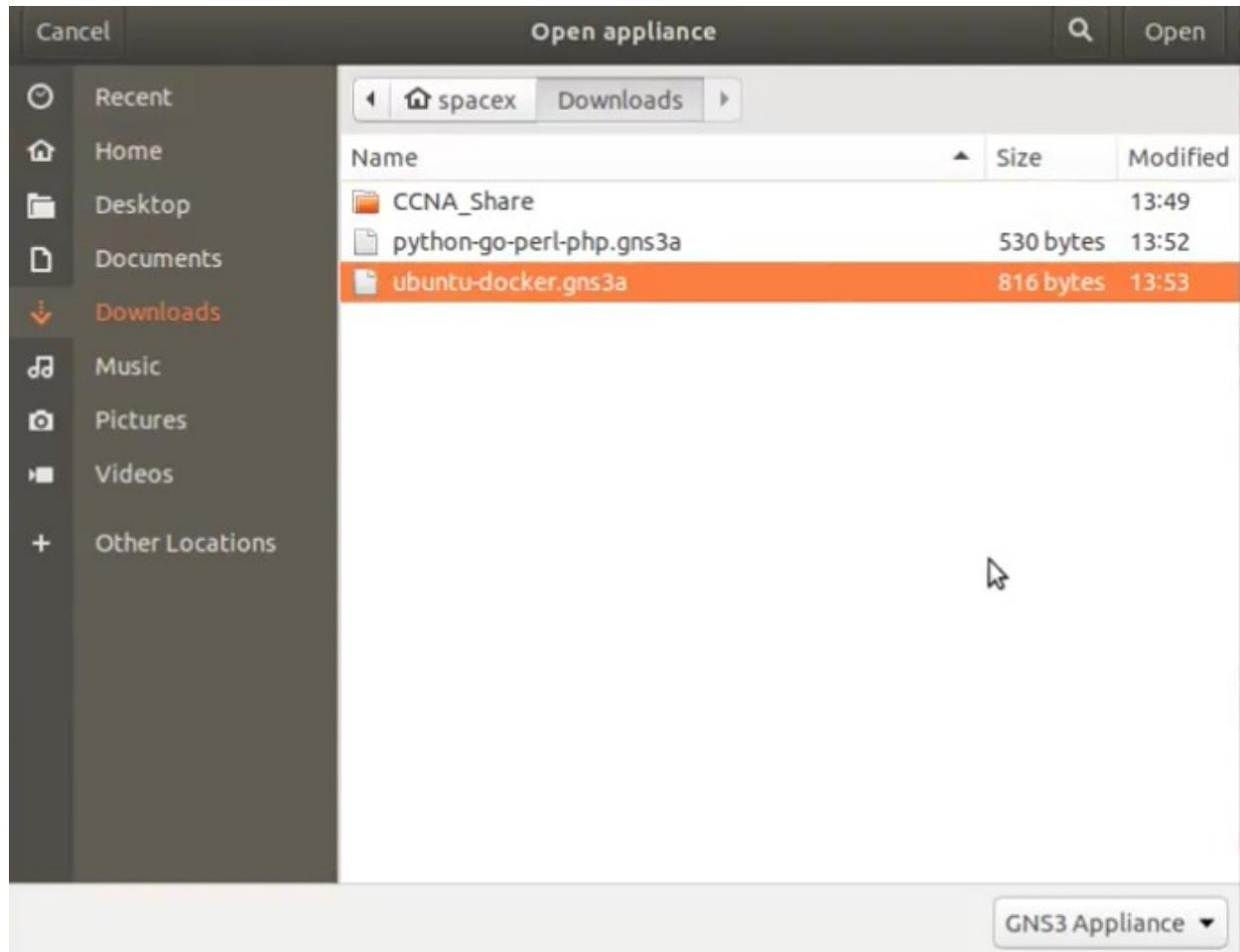
A screenshot of a web browser window titled "Marketplace - Ubuntu Docker". The URL in the address bar is https://gns3.com/marketplace/appliance/ubuntu. The page content shows the "Ubuntu Docker Guest" appliance by Canonical. It features an orange icon with the white Ubuntu logo. Below the icon, the text "Ubuntu Docker Guest" and "APPLIANCE BY CANONICAL" is displayed. At the bottom, there is a blue "DOWNLOAD" button.

GNS3 menu ခဲ့ File > Import Appliances ကို ရွေးလိုက်ပါ။

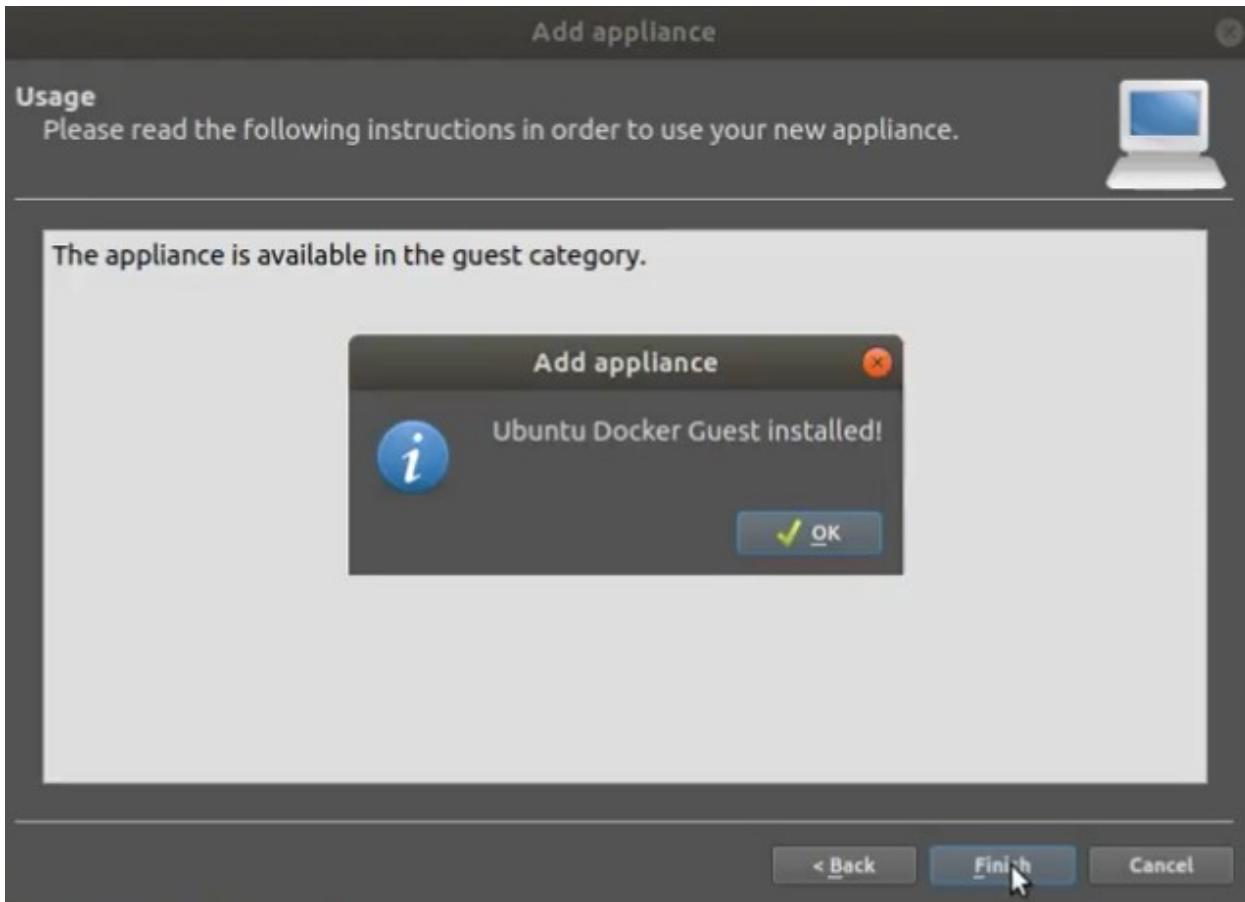


ကိုယ်သွင်းမထုံးကောင်ကို ယူလိုက်ပါ။

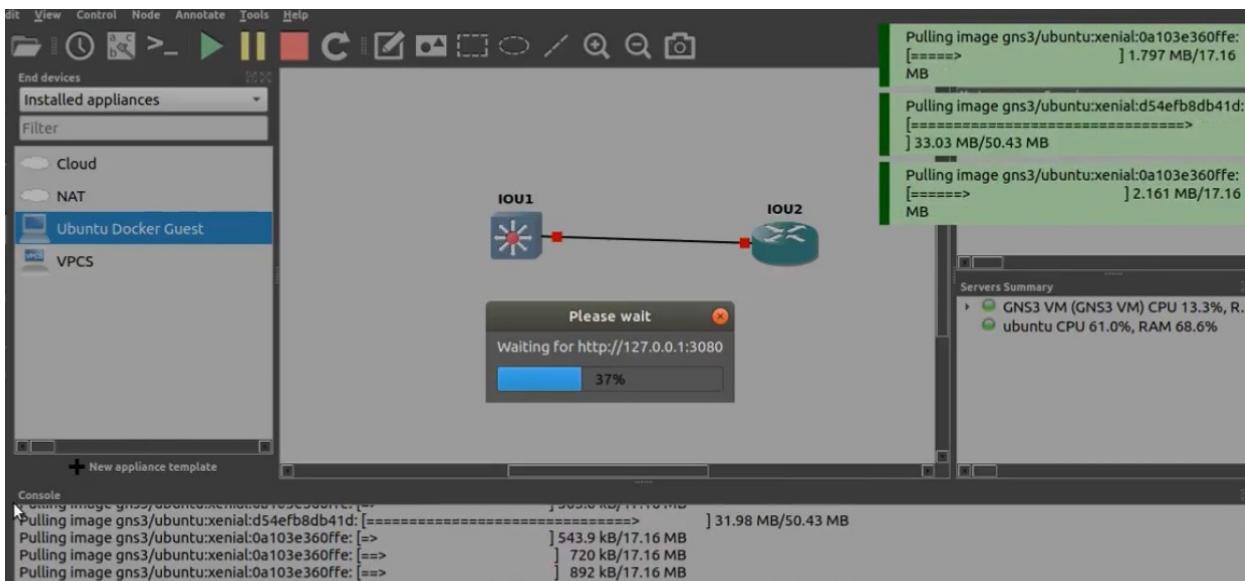
<http://crossnetmm.com/>



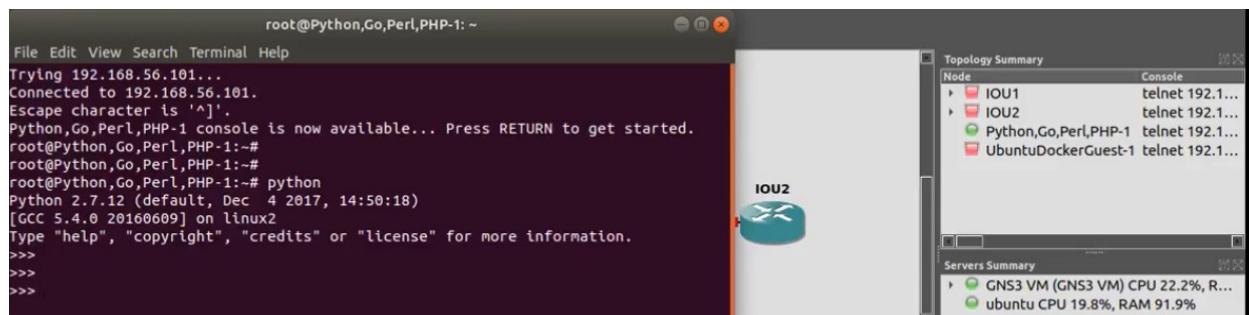
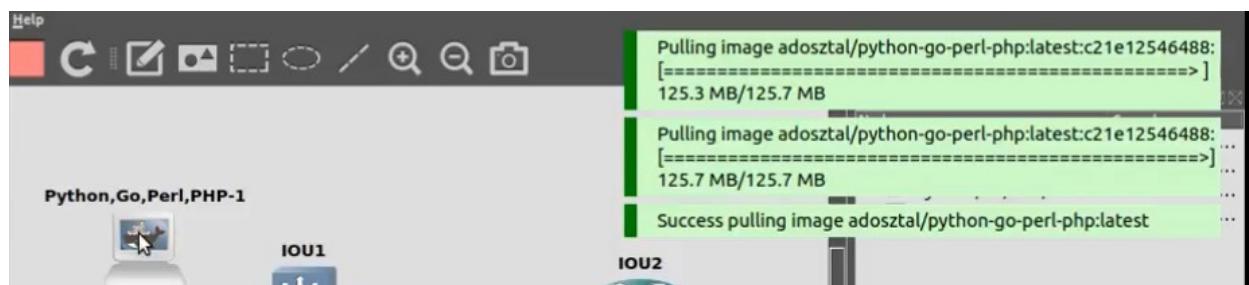
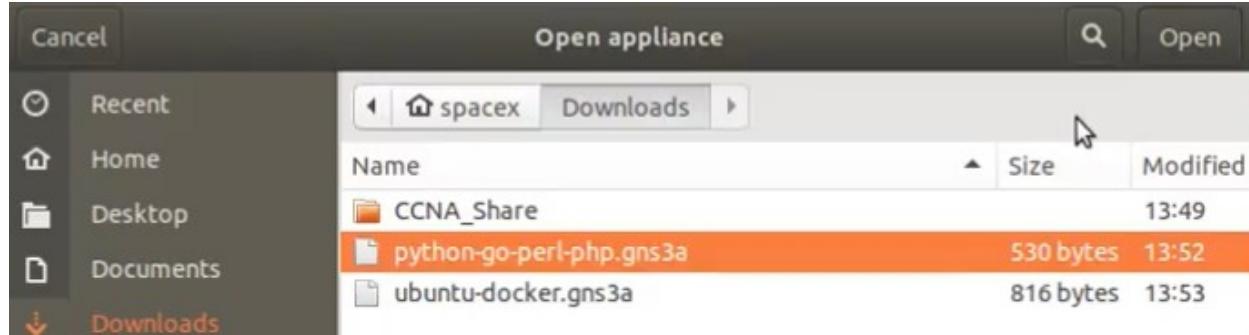
Next Next okay finished 001



Docker သွင်းလိုက်တာပဲ ရှိပါသေးတယ်.. Container တို့ pull မလုပ်ရသေးပါဘူး.. ကိုယ်သွင်းထားတဲ့ ကောင်ကို topology workspace မှာ Drag&Drop လုပ်လိုက်မှာ သွင်းသွားတာပါ။



Python အတွက်လဲ Ubuntu နဲ့ အတူတူပါပဲ။



အကုန်သွင်းပြီးသွားရင် ကိုယ်စမ်းချင်တဲ့ device ကို right click ဆောက်ပြီး run လိုက်ပါ အဲနောက် right click ပြန်ဆောက်ပြီး console ဣဣဣ ကြည့်လုပ်ပါ။ အဲတာဆို ရပါပြီး။

ခုအောက်က ကောင်က video link ပါ။

<https://www.youtube.com/watch?v=KraZsFiZzMQ&t=87s>

13. Python Telnet Script

ကဲ python network program သေးသေးလေး တစ်ခု ရေးဖို့ ပြင်ဆင်လိုက်ရအောင်များ

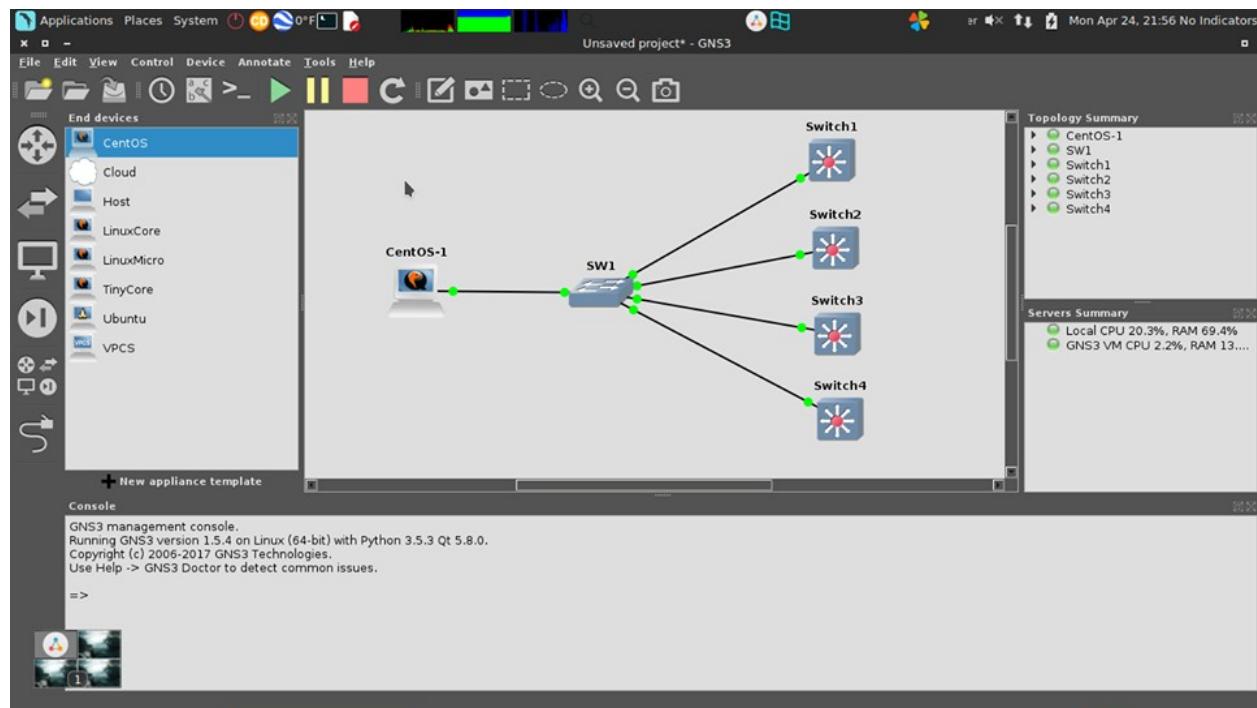
<http://crossnetmm.com/>

<https://sourceforge.net/proj.../gns-3/files/Qemu%20Appliances/>

အဲလင့်ကန္ဒြီးတွေ၊ lisa-2.0.1-centos-6.3.tar.bz2 ဆိတဲ့ centos အတွက် qemu appliance လေးကို ဒေါင်းလိုက်ဟာ ပြီးတွေ \$ tar xjf lisa-2.0.1-centos-6.3.tar.bz2 ဆိုပြီး extract လုပ်လိုက်ပါ lisa-2.0.1-centos-6.3.img ဆိုပြီး ရလာပါလိမ့်မယ် image file လေးကို QEMU မှာ ထည့်ပုံထည့်နည်း အရင်ကရေးပြားပါတယ် page မှာ ပြန်ရှာရှုပြုပါ .. ခုတွေ အလွယ်ပဲ ပြန်ပြောပြီတွေမယ်..

GNS3 ရဲ့ edit ကနေ preference ကို သွားပါ ပြီးတွေ Qemu VMs ကို နိုင်ပါ... New ကိုရွေးပါ run this Qemu VM on GNS3 VM ကို select လုပ်ပါ next နိုင်ပါ qemu binary ကို default အတိုင်းထားပါ next နိုင်ပါ new image ရွေးပေးပါ upload လုပ်မှာလားဆို yes ရွေးပေးပါ finished ကိုနိုင်ပါ... အဲတာဆုံး centos ကို qemu မှာ သွင်းထားပါးဖြစ်ပါတယ် တစ်ချို့ GNS3 appliance မရတဲ့ ဘူတွေ အတွက်ပါ။

Topology



ပြီးတွေ switch တွေကို control access ရနိုး local password ပေးပါ

`#enable password cisco`

ပြီးတွေ telnet login အတွက် username & password create လုပ်ပါ

`#username spacex password cisco`

`#line vty 0 4`

`#login local`

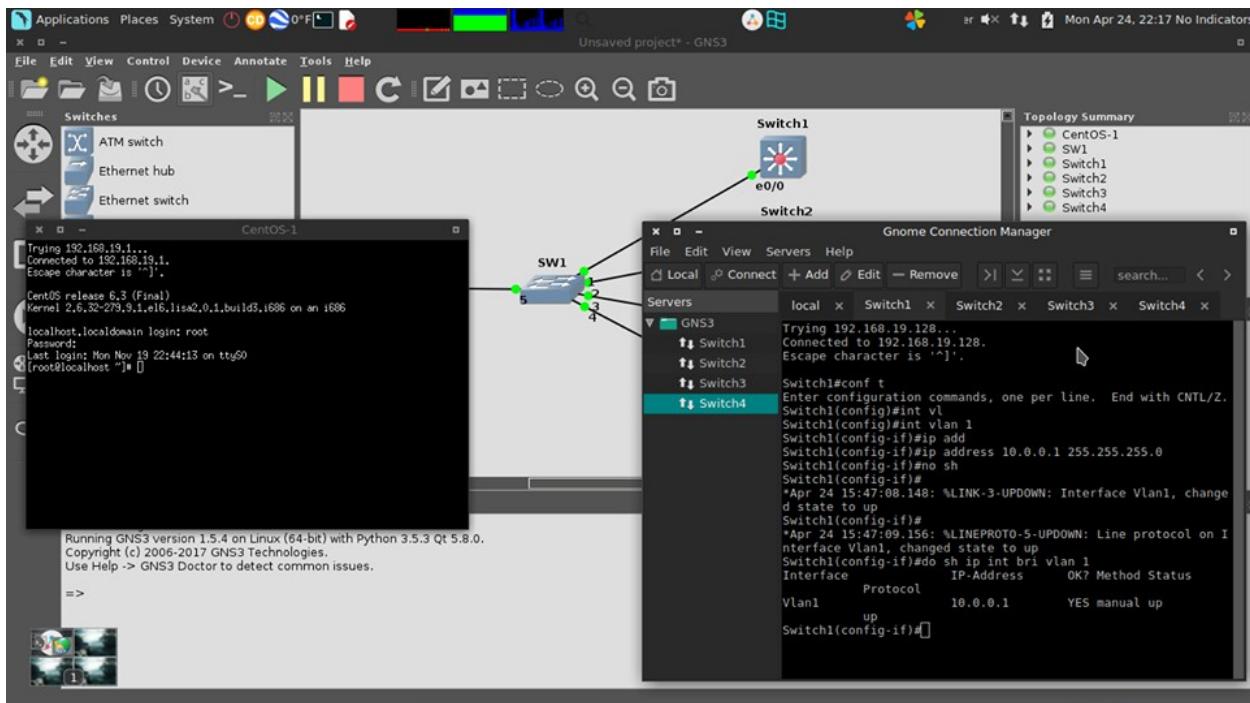
`#transport input all`

`#end`

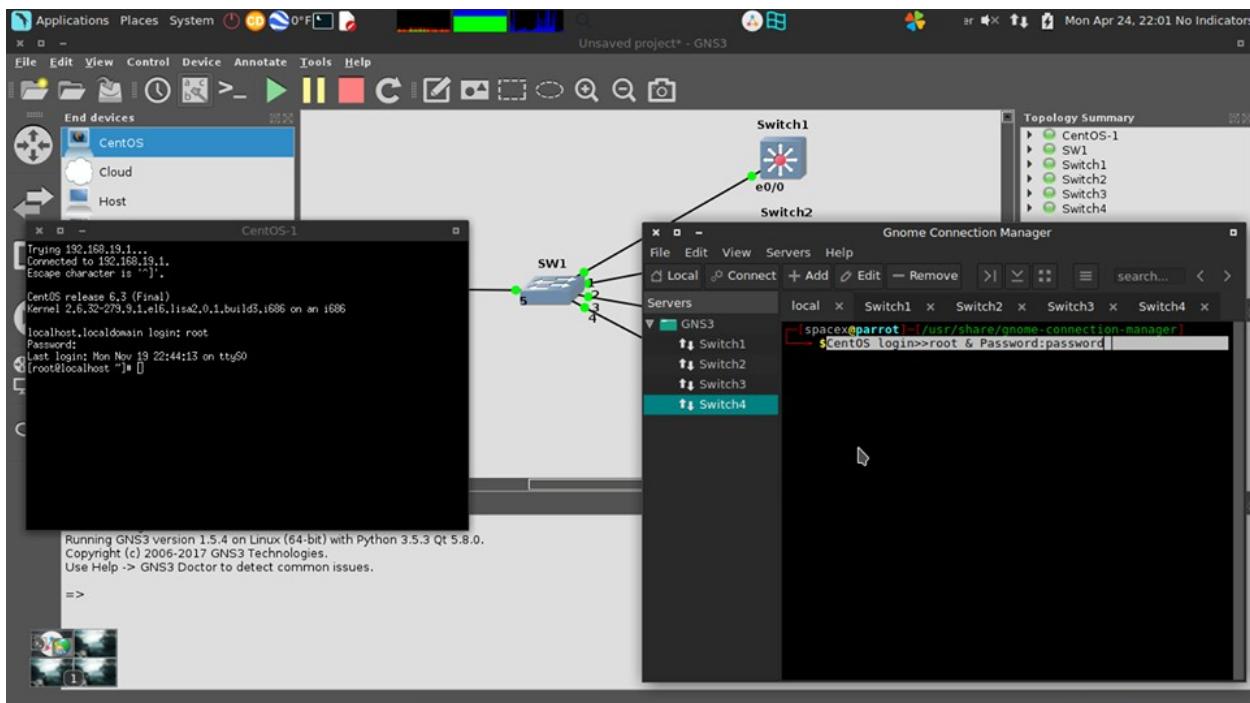
ip ပေးပါ

<http://crossnetmm.com/>

```
#int vl 1  
#ip addr 10.0.0.1 255.255.255.0  
#no sh
```

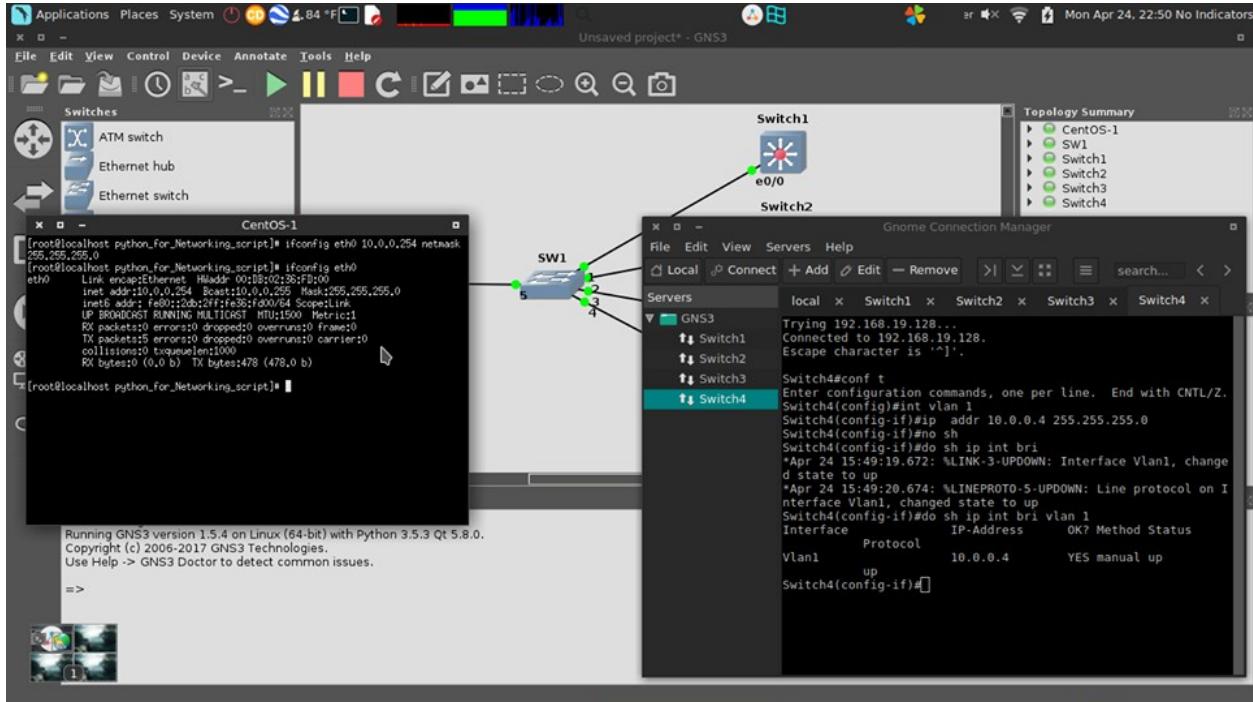


အဲတာဆို switch ဘက်ခြမ်းပြီးသွားပါပြီ နောက် linux ဘက်ခြမ်းမှာ ip configure လုပ်ရပါမယ် centos >> login:root & password:password

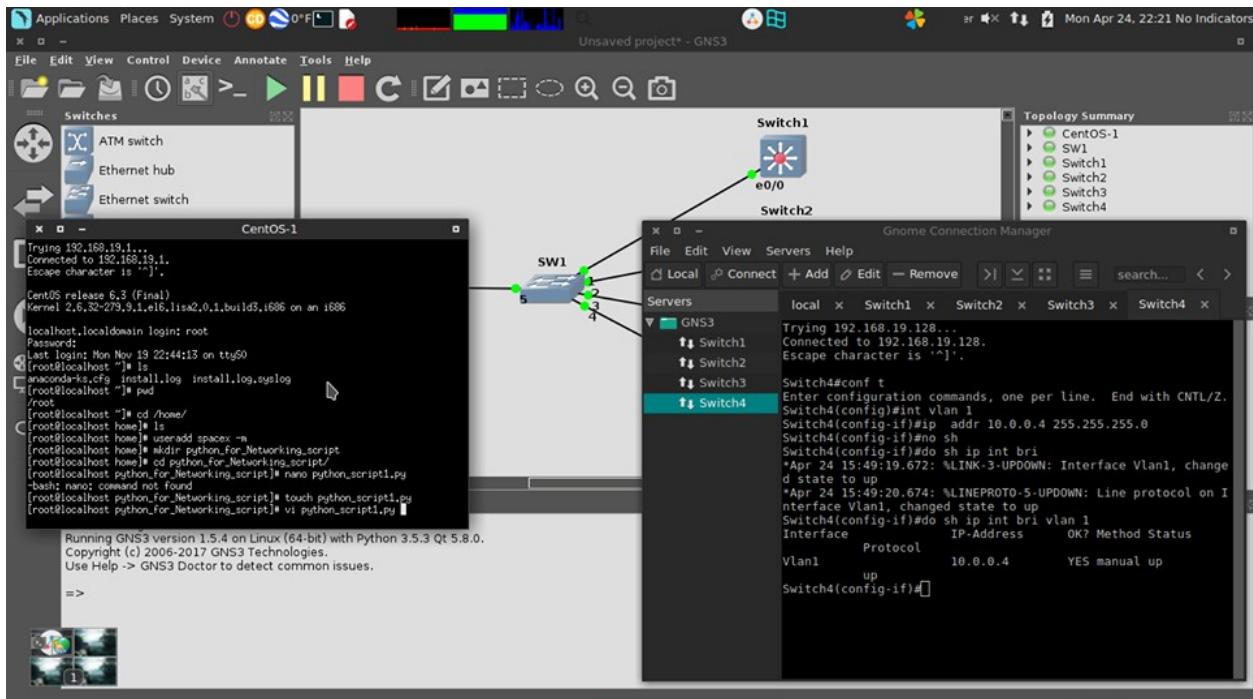




```
ifconfig eth0 10.0.0.254 netmask 255.255.255.0
```



ပြုခဲ့သူ python script စပါမယ်...



```
import getpass
```

```
import sys
```

```
import telnetlib //library တွေကို အခင် importလုပ်ပါ
```

<http://crossnetmm.com/>

```

HOST = raw_input("Enter your Device IP: ") /// user သိက input တောင်းပြီး HOST ဆိုတဲ့ variable
ထဲsave ပါတယ်

user = raw_input("Enter your telnet username: ") /// username ကို user ဆိုတဲ့ variable ထဲsave မှု
တယ်

password = getpass.getpass() /// "Password:" ဆိုတောင်းပြီး password ဆိုတဲ့ variable ထဲsave ပါတယ်
tIn = telnetlib.Telnet(HOST) /// telnet connection ဆောက်ပါတယ် HOST ထဲမှာသိမ်းထားတဲ့ IP နဲ့
tIn.read_until("Username: ") ///"Username: "ဆိုပြီးမေးမှာကို တောင့်ပါတယ် မေးတာကို မြင်တာနဲ့ အောက်
တစ်ခြားလုပ်မှု ဆင်းပါတယ်

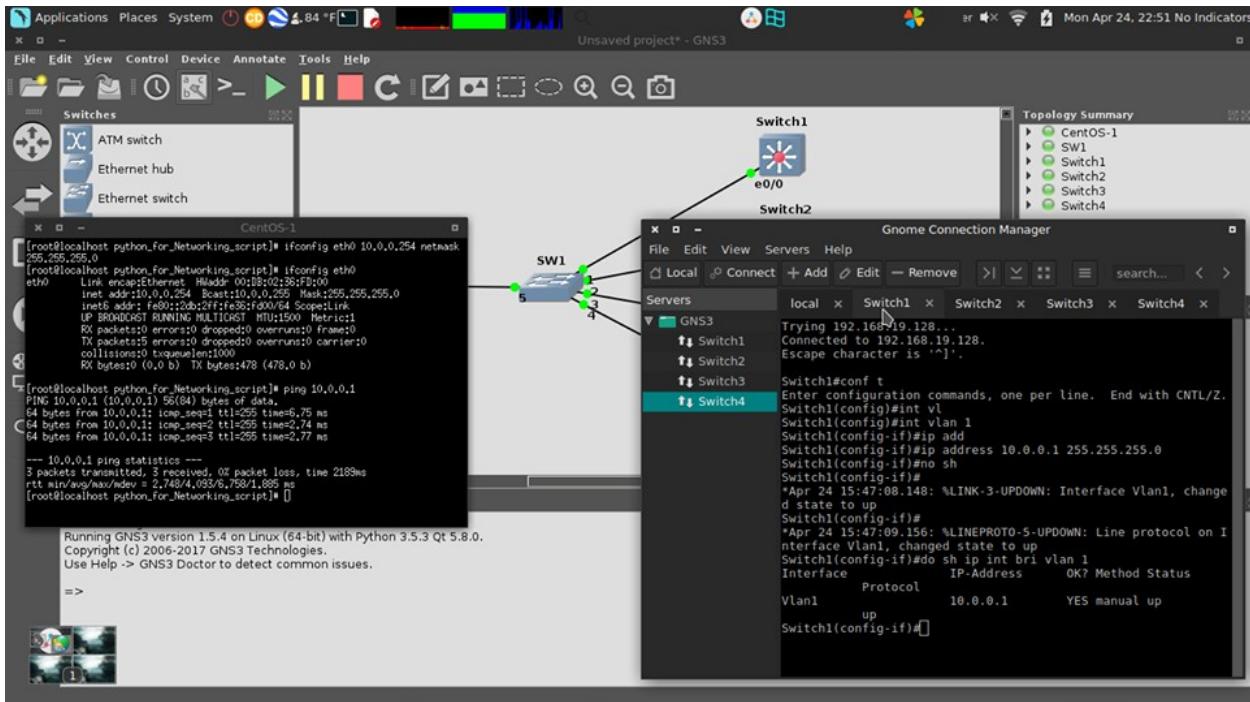
tIn.write(user + "\n") /// user ထဲမှာ သိမ်းထားတဲ့ username ရှိက်ပါတယ်
if password: /// password တောင်းတုန်းက ရှိက်ခဲ့ရင်
    tIn.read_until("Password: ") ///"Password: " ဆိုပြီးတောင်းတာနဲ့ နောက်တစ်ခြားလုပ်မှု ဆင်းပါတယ်
    tIn.write(password + "\n") /// password ထဲမှာ သိမ်းထားတဲ့ ရွှေ့နံပါတ်တို့ ရှိက်လိုက်တဲ့ password ကိုထွေးပါတယ်

/////////// UserName & Password မှန်သွားရင် switch ထဲကို ရောက်ပါပြီ loopback configure ချွဲ့ .. 
ရှိက်ရမယ့် command တွေကို တစ်ခုရှုပ်ငါးအတိုင်း အစဉ်လိုက်ရေးပါ////////

tIn.write("enable\n")
tIn.write("cisco\n")
tIn.write("conf t\n")
tIn.write("int lo 0\n")
tIn.write("ip addr 1.1.1.1 255.255.255.255\n")
tIn.write("exit\n")
tIn.write("end\n")
tIn.write("exit\n")

print tIn.read_all() /// ကိုယ် switch မှ ရှိက်ခဲ့တာတွေကို မြင်ရအောင် ပြန်ထုတ်တာပါ... 😊
script မှ run ခင် ping ဖြေားပါ

```

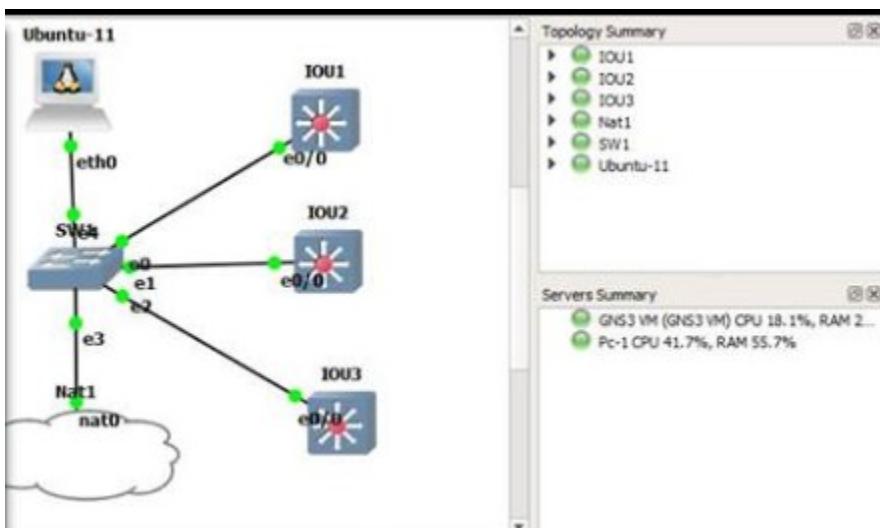


python script run යා

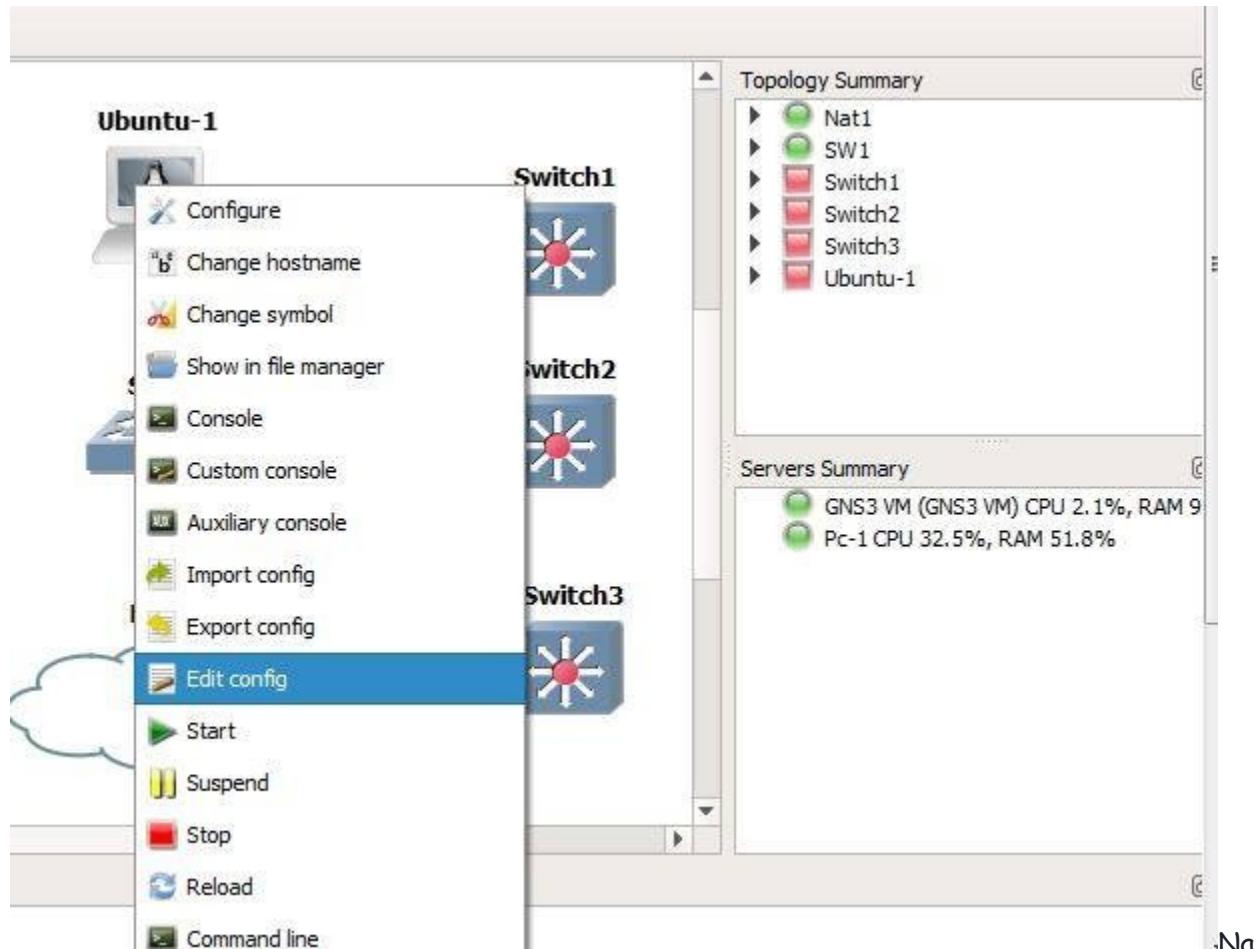
ඇම් script ගෙනරුත් telnetlib තිබූ යුතුයේ cisco device න් හෝ telnet login යැයි පිටතයි. පෙනෙනු ලද loop back interface create ලදී. IP පෙළිවාගාපි

14. Config vlan with for loop

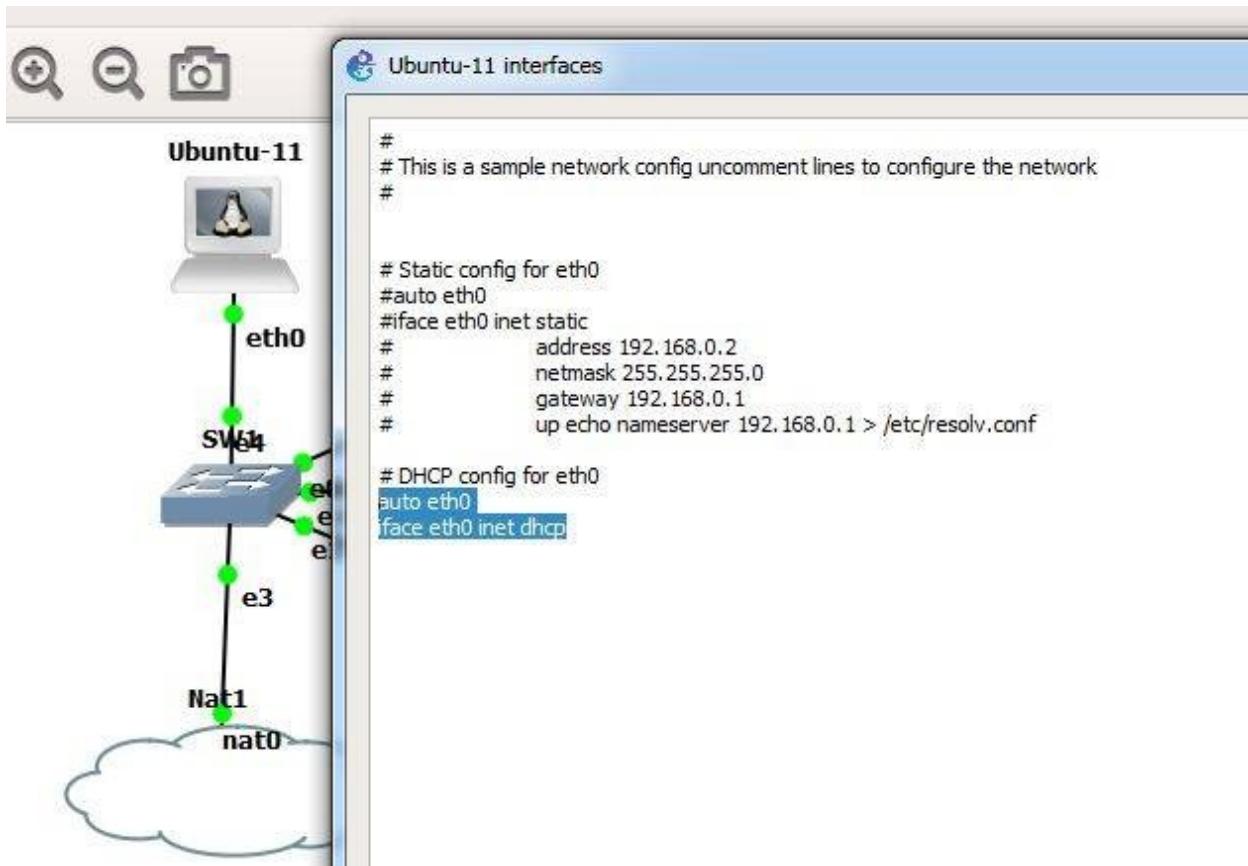
Topology යා



Ubuntu PC තිබූ right click ගෙනරුත්: Edit config නිව්‍යාපි



† Cloud သဲ DHCP support ပေးတဲ့ အတွက် Ubuntu ရဲ့ eth0 ကို dhcp on ပေးလိုက်ပါ



Nat Cloud ၂ GNS3 2.0 မှာ support ပေးပါတယ်...

GNS3 1.5 အတွက်ဆုံး <https://websistent.com/how-to-connect-gns3-to-the-internet/>

အဲလင့်မှာ ပိတ္ထအတိုင်း config ခဲ့ပါ

second 20 လောက် နေရင် ip ရာလာပါလိမ့်မယ် .. မရခဲ့ရင်

\$ifconfig eth0 down

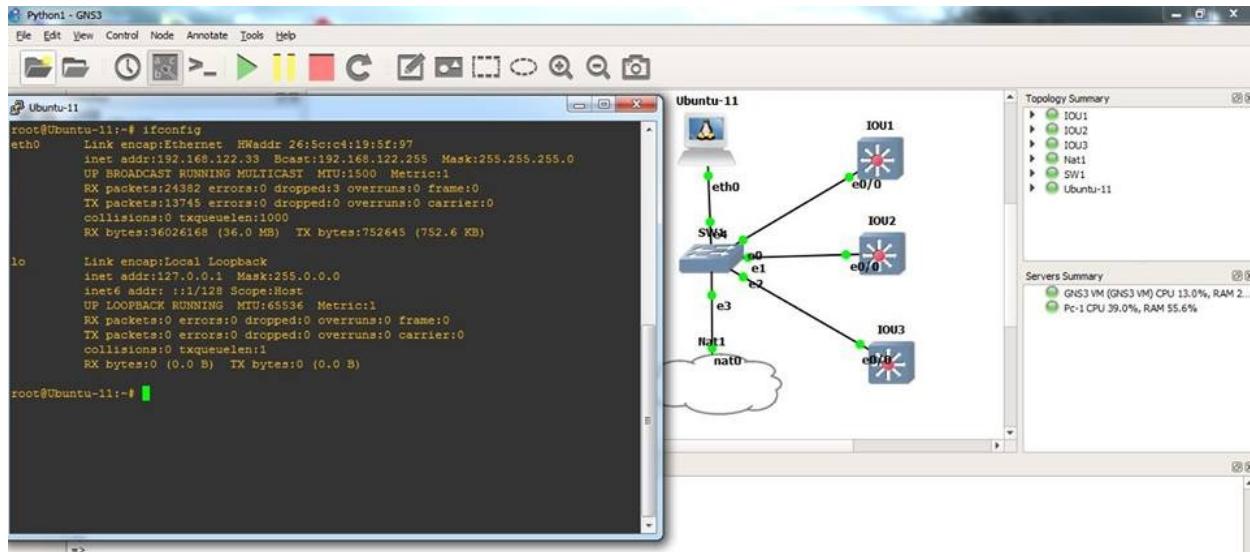
\$ifconfig eth0 up

eth0 interface ကို disable/enable လုပ်ပေးပါ

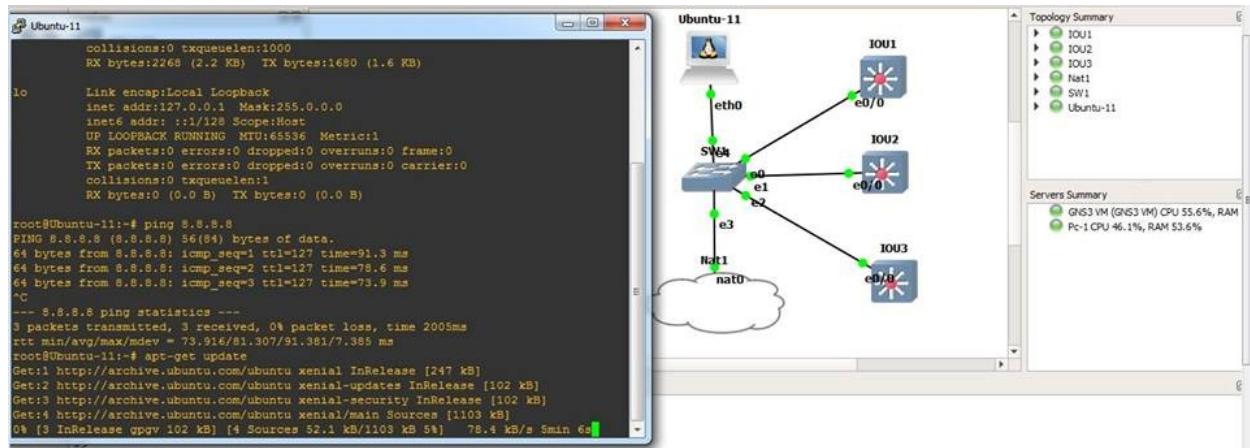
Now got IP



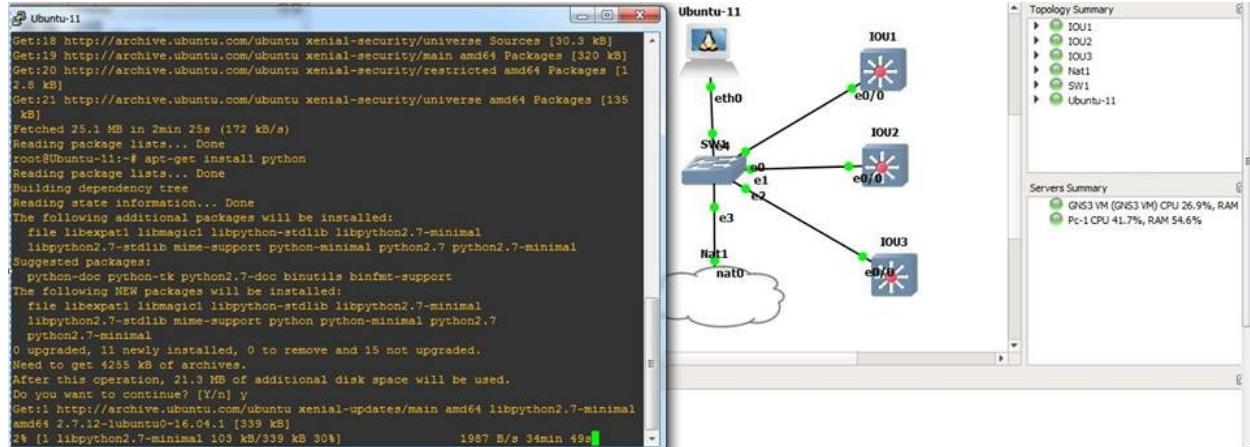
<http://crossnetmm.com/>



internet connection ရ မရ ping တို့ပါ... ြေးတွေ \$apt-get update လုပ်ပါ



လုပ်ပြီးရင် python install လုပ်ပါ



python အလုပ် လုပ် မလုပ် စမ်းတို့ပါ

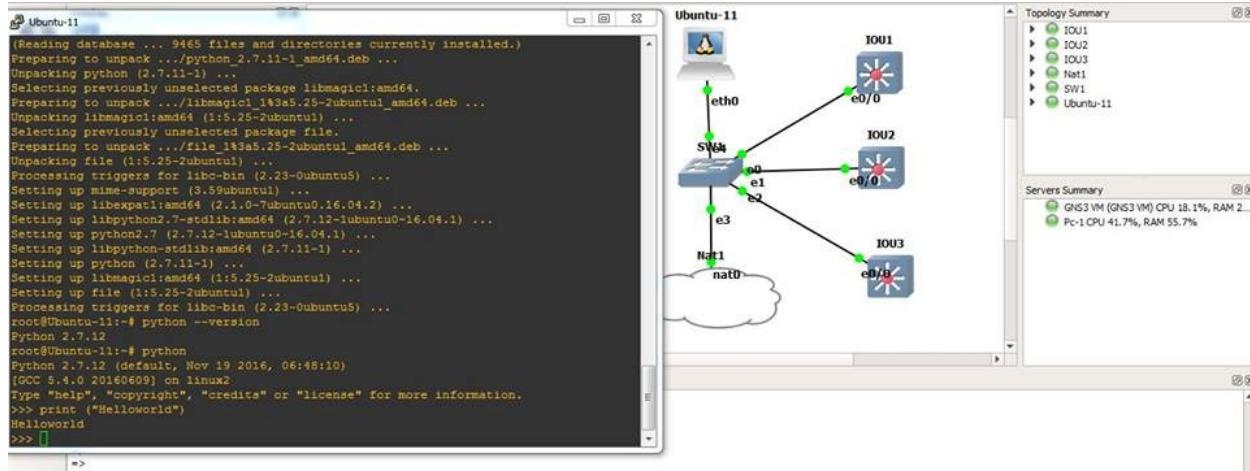
root@Ubuntu-11:~#python



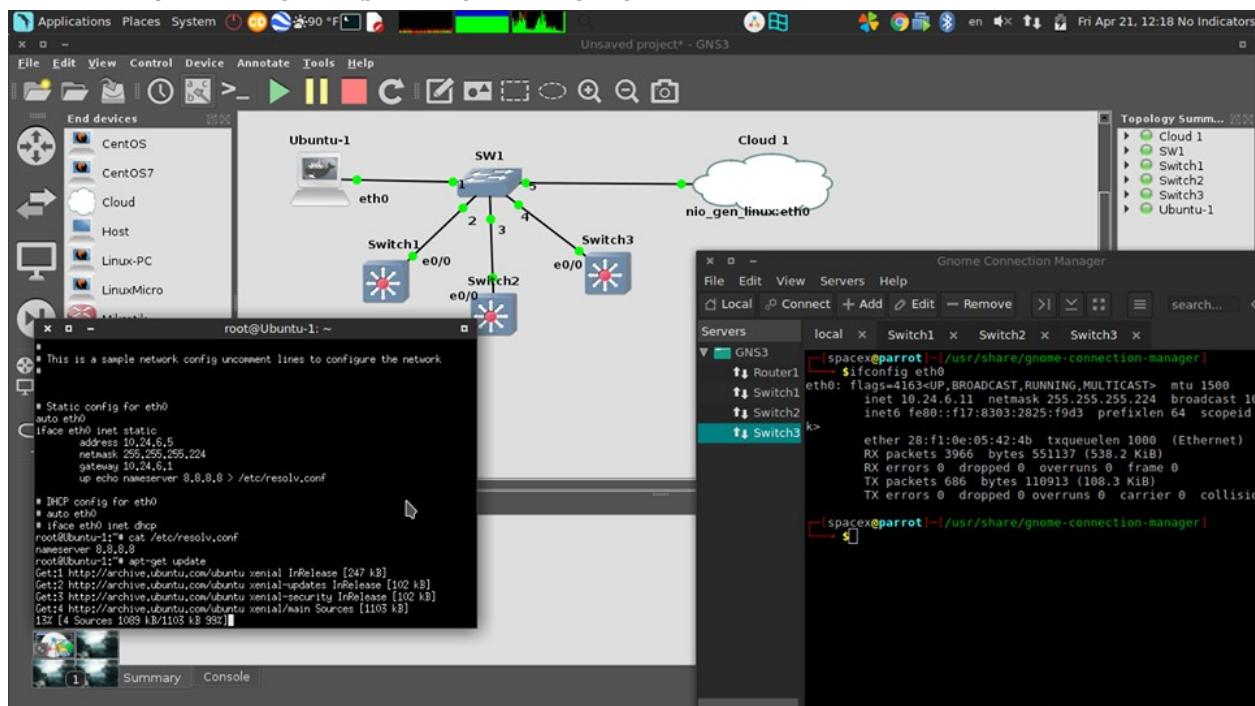
```
>>>print("Helloworld")
```

Helloworld

```
>>>
```



window အတွက်တင်မဟုတ်ပါဘူး linux မှာလ အလုပ်လုပ်ပါတယ်...



တို့တွေ Vlan configuration လုပ်မယ့် switch တွေကိုလဲ Ubuntu ရဲ့ IP နဲ့ Same Network ဖြစ်အောင် ပေးပါ
\$ifconfig eth0 ဆိုပြီး ကြည့်လိုက်ရင် ip:192.168.122.33 & Mask:255.255.255.0
အဲတွော switch1,switch2,switch3 ကို vlan 1 interface ကို 192.168.122.11, 192.168.122.12, 192.168.122.13 ဆိုပြီး ပေးလိုက်တယ်မှာ



The screenshot displays three terminal windows:

- Ubuntu-11:** Shows the output of the `ifconfig` command, listing interfaces eth0 and lo. It also shows the result of a `ping` command to 192.168.122.11.
- Ubuntu-11:** Shows the configuration of a Cisco switch interface. It includes commands like `enable password cisco`, `username spacex password cisco`, and `ip add 192.168.122.11 255.255.255.0`.
- Topology Summary:** A network topology diagram showing two nodes: IOU1 and IOU2.

<https://drive.google.com/file/d/0B6-W5T0kUugea2JSc0JhTG5UR1k/view>

အဲလင့်က script အတိုင်း ရှိ၍ပါ.. python မှာ space စွဲ tab စွဲ သတိထားပါ.. for loop ပါ၏တဲ့အခိုင်မှာ..





```
import getpass
import sys
import telnetlib

vlan_count=1

no_device=raw_input("How many device do you want to configure: ")
device=int(no_device)
no_vlan=raw_input("How manay Vlan do you want to config: ")
vlan=int(no_vlan)

for i in range(device):
    HOST=raw_input("Enter Your Devie IP: ")
    user=raw_input("Enter Your Telnet UserName: ")
    password=getpass.getpass()
    tln=telnetlib.Telnet(HOST)
    tln.read_until("Username: ")
    tln.write(user + "\n")
    if password:
        tln.read_until("Password: ")
        tln.write(password + "\n")
    tln.write("enable\n")
    tln.write("cisco\n")
    tln.write("conf t\n")
    for i in range(vlan):
        vlan_count=vlan_count+1
        tln.write("vlan " + str(vlan_count) + "\n")
        tln.write("name VLAN" + str(vlan_count) + "\n")
        tln.write("exit\n")
    tln.write("end\n")
    tln.write("exit\n")
    print tln.read_all()
```

```
#####
#####
```

Script

```
import getpass
import sys
import telnetlib
vlan_count=1

no_device=raw_input("How many device do you want to configure: ")
device=int(no_device) // ဝင်လာတောက str typeပါ အဲဒြောင့် int type ပြောင်းတာပါ..for loopမှာ
integer ထည့်ရမှာမို့
no_vlan=raw_input("How manay Vlan do you want to config: ")
vlan=int(no_vlan) // ဝင်လာတောက str typeပါ အဲဒြောင့် int type ပြောင်းတာပါ..for loopမှာ integer ထည့်ရမှာမို့
for i in range(device): // for loop ပါတယ်ပါတယ် i=0 ကနေစူး 1 တိုး တိုးသွားဖြီး <device ထိပါ
HOST=raw_input("Enter Your Devie IP: ")
```

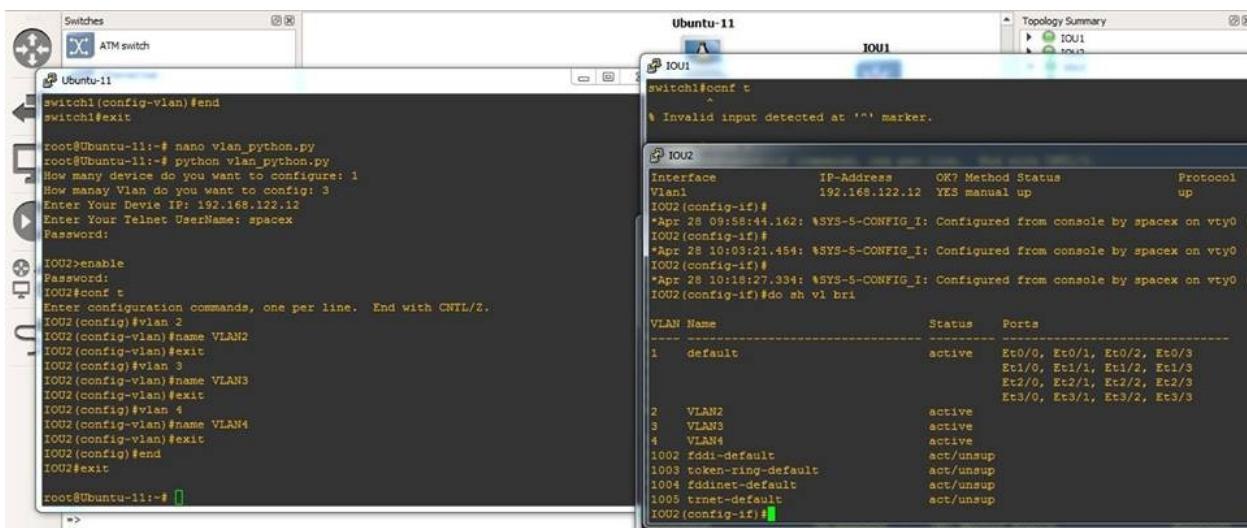


```

user=raw_input("Enter Your Telnet UserName: ")
password=getpass.getpass()
tln=telnetlib.Telnet(HOST)
tln.read_until("Username: ")
tln.write(user + "\n")
if password:
    tln.read_until("Password: ")
    tln.write(password + "\n")
tln.write("enable\n")
tln.write("cisco\n")
tln.write("conf t\n")
for i in range(vlan): /// for loop ဝါတ်ပါတယ် i=0 ကနေစပိုး 1 တိုး တိုးသွားပြုး <device ထိပါ
    vlan_count=vlan_count+1
    tln.write("vlan " + str(vlan_count) + "\n")
    tln.write("name VLAN" + str(vlan_count) + "\n")
    tln.write("exit\n")
    tln.write("end\n")
    tln.write("exit\n")
print tln.read_all()

```

sublime text နဲ့ရေးပြောင်းပါ ကိုင့် loop ဘယ်နားထိရောက်လဲ မြင်သာပါတယ်.... 😊
 for loop ဘယ်လို အလုပ်လုပ်လဲ ဆို တာကို နောက်ဆုံးပုံမှာ ပြောင်းပါ
 vlan creation finished with python script



သုတေသနတွင် အပြီးရေးထားတာဆိုတော့.. Script နှစ်ခုပဲ ရှိပါသေးတယ်.. ပြီးတော့အရင်က ရေးဖူးတာကို ပြန်ထုတ်ထားတို့တော့ ရှင်းဖို့ လိုအပ်တာလေးတွေလဲ တွေပါတယ်။ အဲတာကို version 2 ရှာရင် ပြန်ပြီးဖြည့်ရေးပေးထားမယ် မှာ ။။။ ပြီးတော့ SSHv2 အတွက် script တွေ backup script တွေ ထပ်ပြီး အားဖြည့်ပေးမယ် .. see ya !!

With Best Regards,

Khant Phyo

Core Team Leader

khant.phyo@ericsson.com

+09442329009