



# Computer Fundamentals “Software”



# Agenda



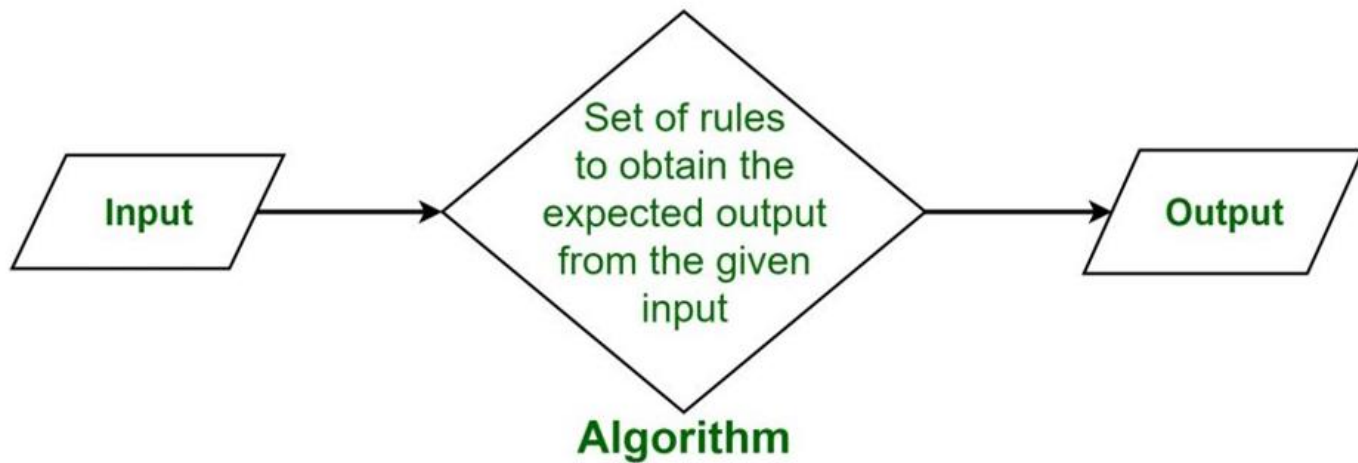
- ▶ What is Algorithm
- ▶ Software
- ▶ Machine Language
- ▶ Assembly Language
- ▶ High Level Languages
- ▶ Libraries/Packages/Frameworks
- ▶ Backend/Frontend



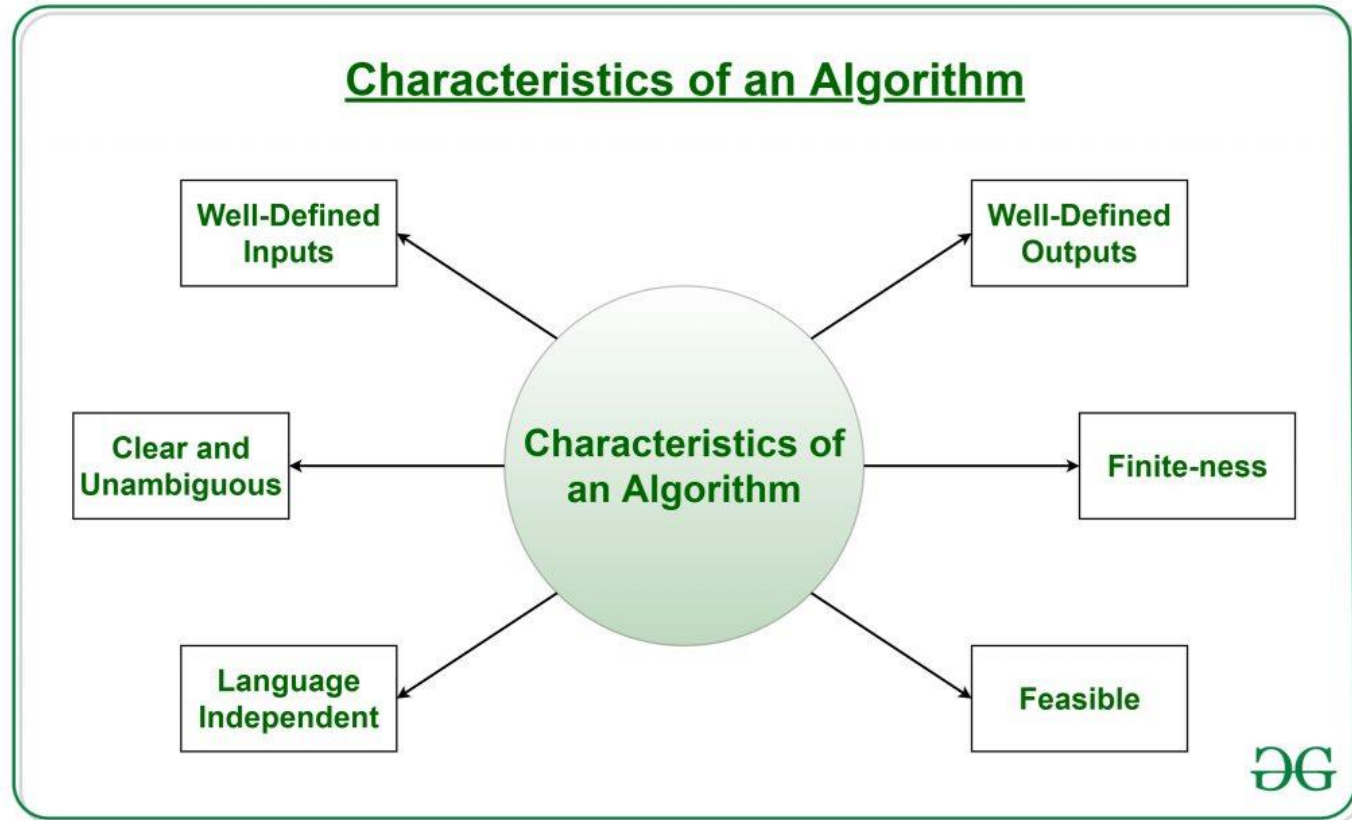
# Kahoot!



# What is Algorithm



# What is Algorithm



# What is Algorithm



## **Tea Brewing Algorithm:**

Put the teabag in a cup.

Fill the kettle with water.

Boil the water in the kettle.

Pour some of the boiled water into the cup.

Add milk to the cup.

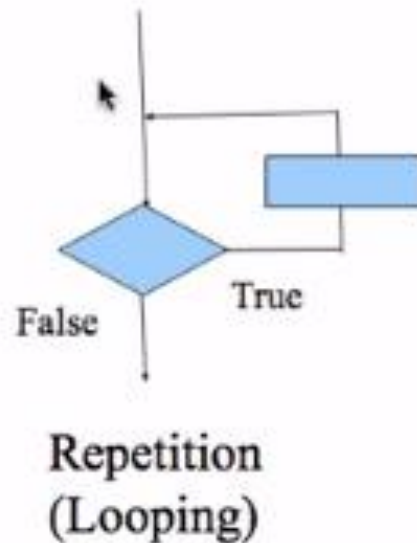
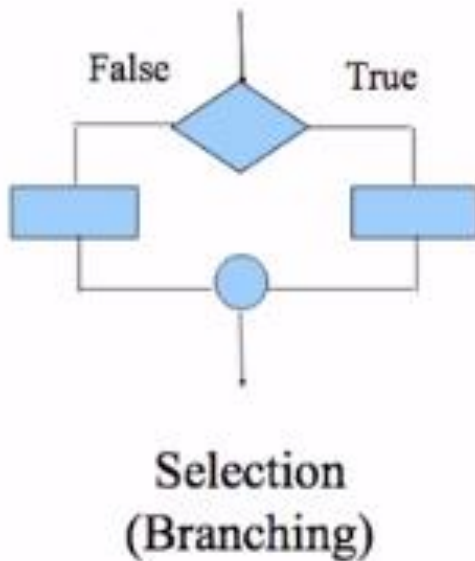
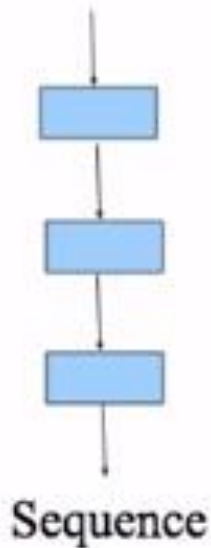
Add sugar to the cup.

Stir the tea.

Drink the tea.



# What is Algorithm

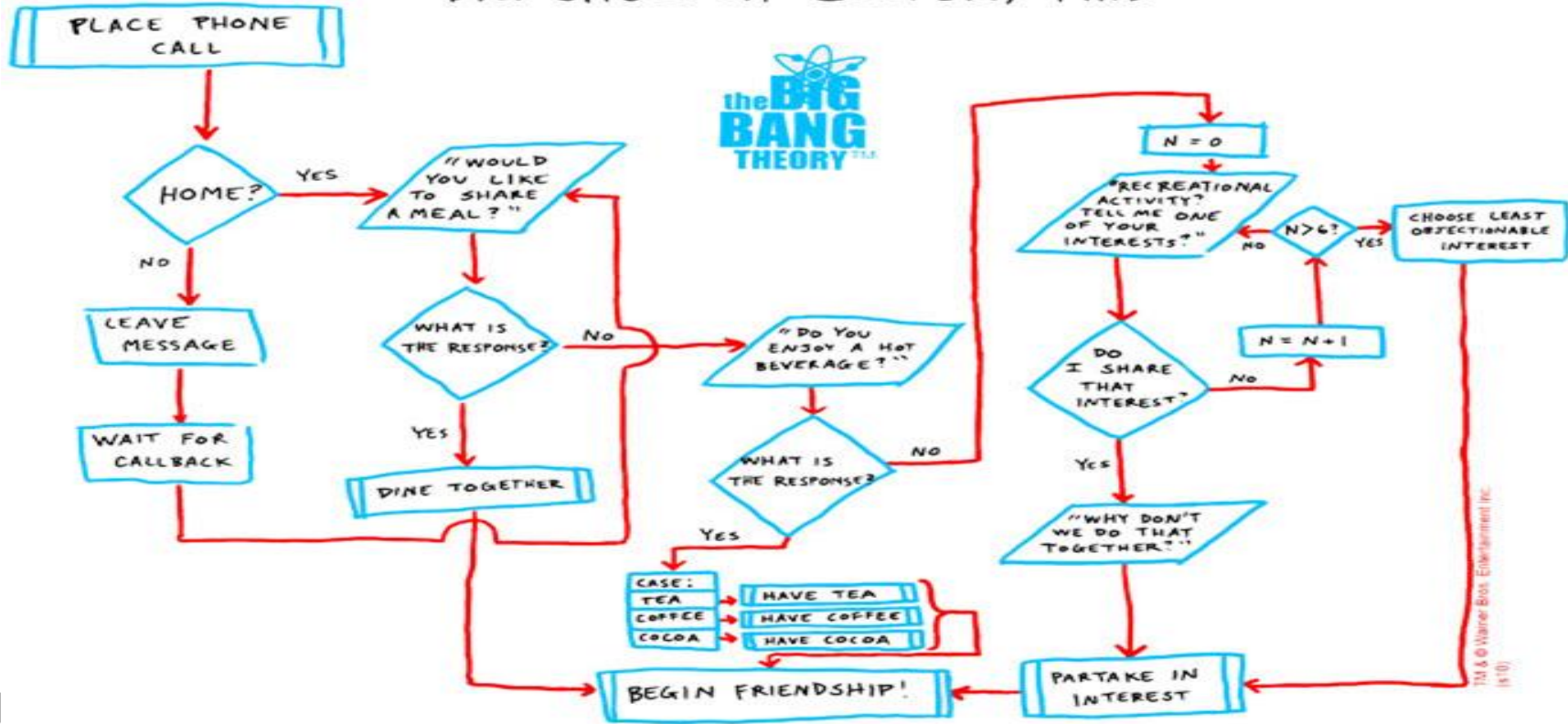


# What is Algorithm



## THE FRIENDSHIP ALGORITHM

DR. SHELDON COOPER, Ph.D





# What is Algorithm





# What is Algorithm

## PSEUDOCODE

---

set total to zero

get list of numbers

loop through each number in the list  
    add each number to total  
end loop

if number more than zero  
    print "it's positive" message  
else  
    print "it's zero or less" message  
end if

# What is algorithm? What is pseudocode?



Students, write your response!

Pear Deck Interactive Slide  
Do not remove this bar



# What is Algorithm

## ALGORITHM VERSUS PSEUDOCODE

### ALGORITHM

An unambiguous  
specification of how to  
solve a problem

Helps to simplify and  
understand the problem

### PSEUDOCODE

An informal high-level  
description of the operating  
principle of a computer  
program or other algorithm

A method of developing an  
algorithm



## What is software?

- unlike hardware it can't be physically touched
- it's the missing link between the computer hardware and the data which it is processing
- has to be “loaded” into the computer's RAM before it can be “run”
- a set of pre-written instructions which the computer executes in order to perform a particular task
- typically written using programming languages such as C, C++, BASIC, Java etc.

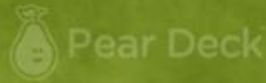
# Software



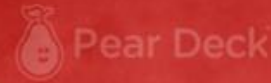


Have you heard of any of Programming Languages?

Yes



No

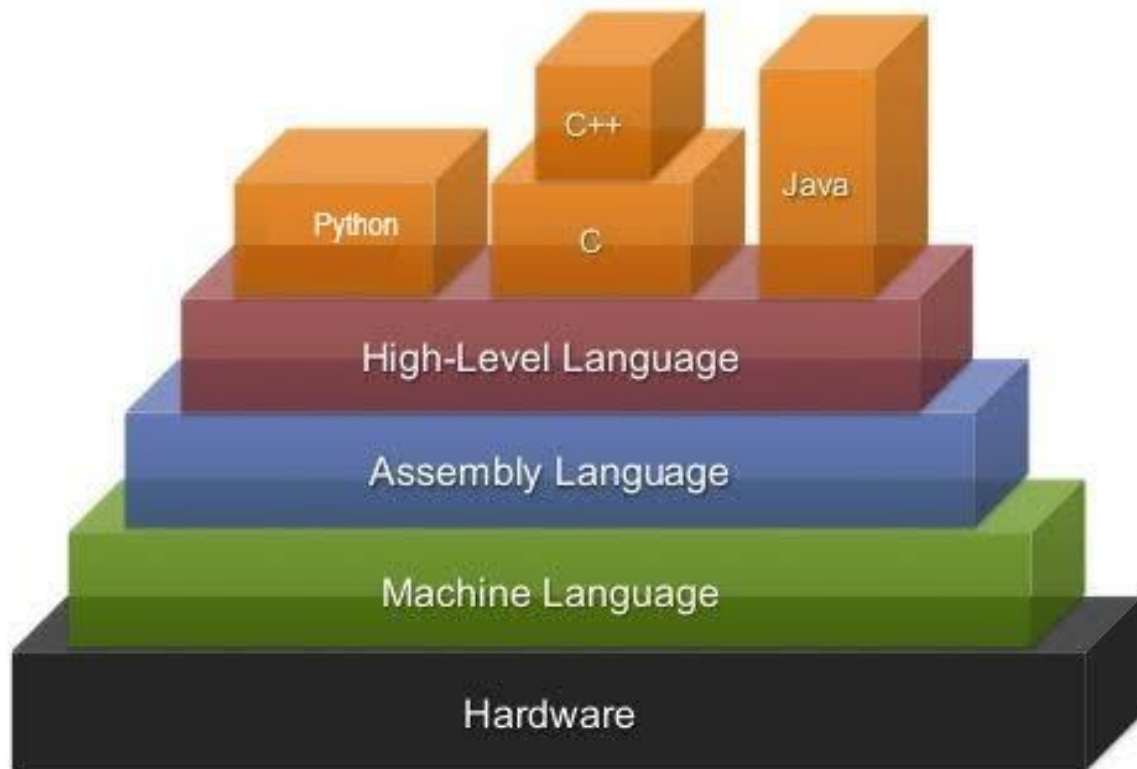


Students choose an option

Pear Deck Interactive Slide  
Do not remove this bar



# Software







# Software



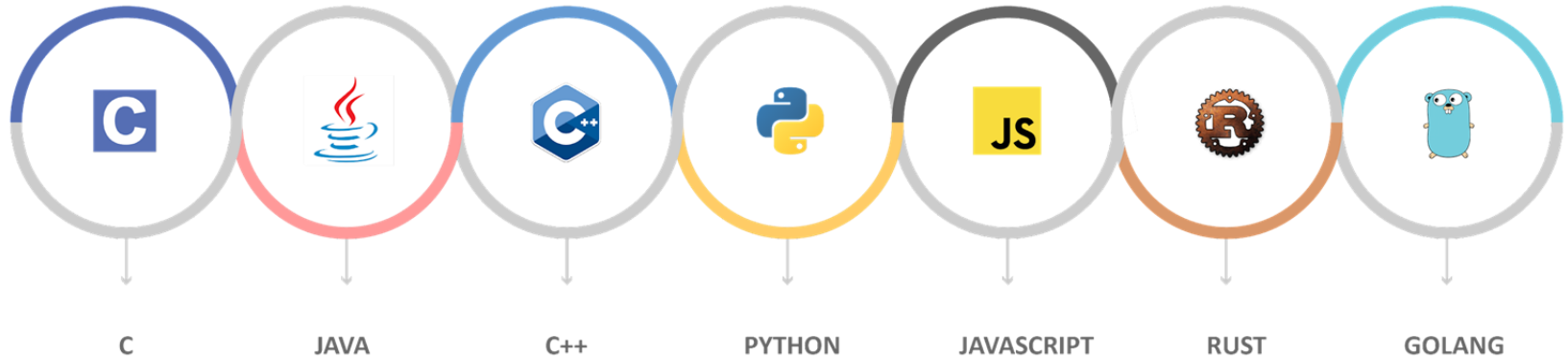
## Low Level Languages:

- Assembly Language
- Machine Language

# Software



## High Level Languages:





# Software



High Level Language	Low Level Language
These are Interpreted	Direct memory management
They have open classes and message-style methods which are known as Dynamic constructs	Hardware has extremely little abstraction which is actually close to having none.
Poor performance	Much fast than high level
Codes are Concise	Statements correspond directly to clock cycles
Flexible syntax and easy to read	Superb performance but hard to write
Is object oriented and functional	Few support and hard to learn
Large community	

# What does highlevel/ lowlevel mean?



Students, write your response!

Pear Deck Interactive Slide  
Do not remove this bar



# Machine Language



## Example of machine-language

Here's what a program-fragment looks like:

```
10100001 10111100 10010011 00000100
00001000 00000011 00000101 11000000
10010011 00000100 00001000 10100011
11000000 10010100 00000100 00001000
```

It means:         $z = x + y;$



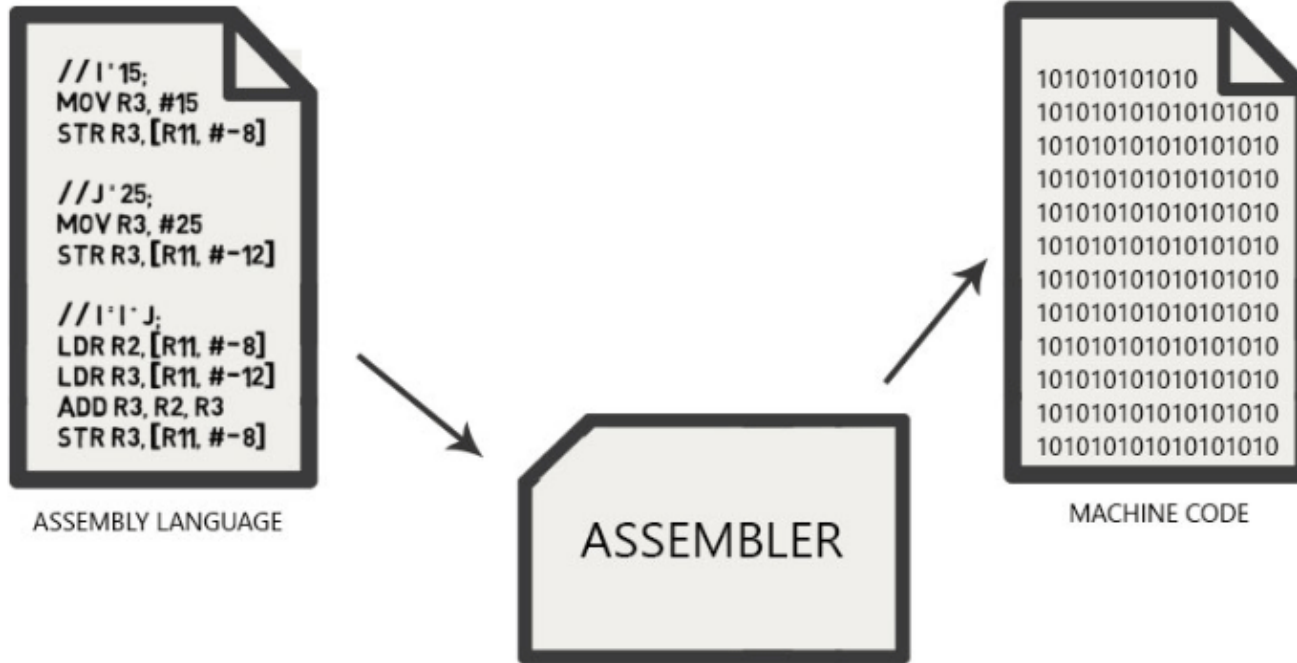
# Assembly Language

- Assembly is converted to machine code

```
mov    %esp, %ebp
sub    $0x28, %esp
mov    0x804d300, %eax
add    $0x1, %eax
mov    %eax, 0x804d300
mov    0x804d300, %eax
cmp    0x8(%ebp), %eax
```

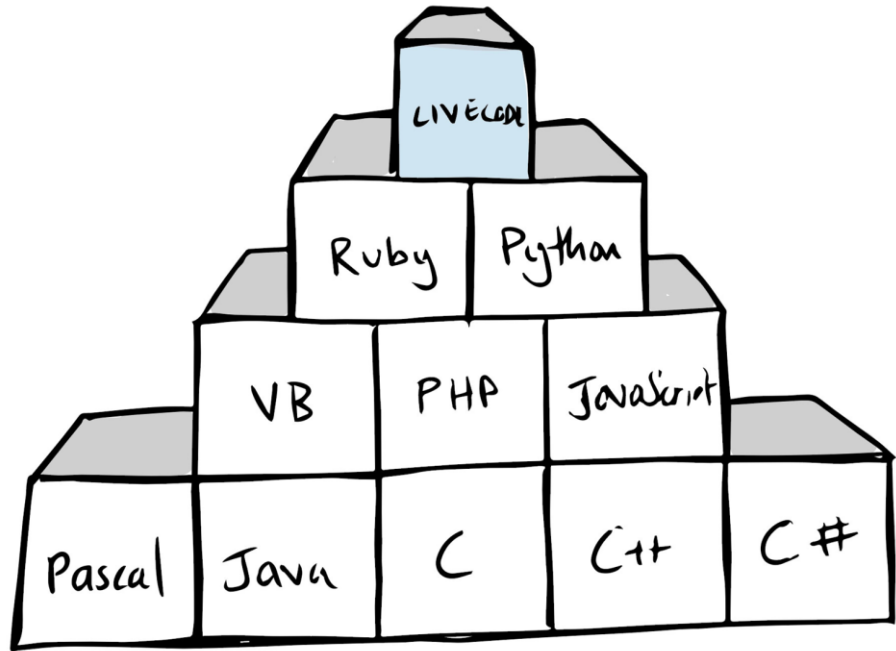


# Assembly Language





# High Level Languages







# High Level Languages



## Source Code:

Source code is a human-readable text written in a specific programming language.

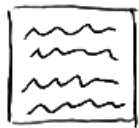
```
40
41 $(function){cards();});
42 $(window).on('resize', function(){cards();});
43 function cards(){
44   var width = $(window).width();
45   if(width < 750){
46     cardssmallscreen();
47   }else{
48     cardsbigscreen();
49   }
50 }
51 function cardssmallscreen(){
  var cards = $(''.card').length;
  height = 0;
  for (var i=0; i<=cards; i++){
    height += cards[i].height;
  }
}
```



# High Level Languages

Source code:

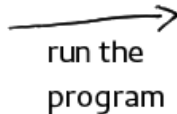
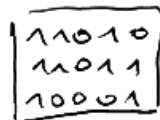
hello.c



COMPILER



Machine code:



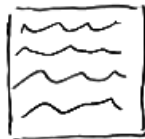
result



Program (also  
called binary,  
executable ...)

Source code:

hello.py



INTERPRETER



result



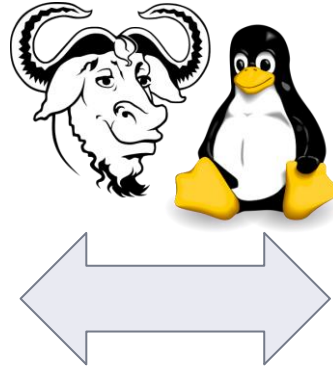
# High Level Languages



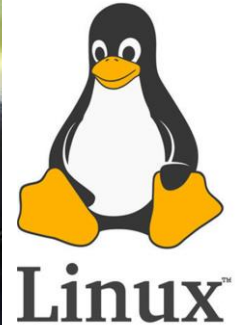
## Free Software vs. Open Source Software



**Richard Stallman**



**Linus Torvalds**





# High Level Languages

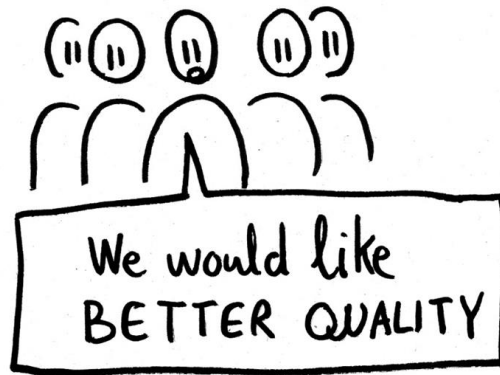


Free software  
activists

Open source  
boosters



**ethical  
approach**



**technical  
approach**

# High Level Languages

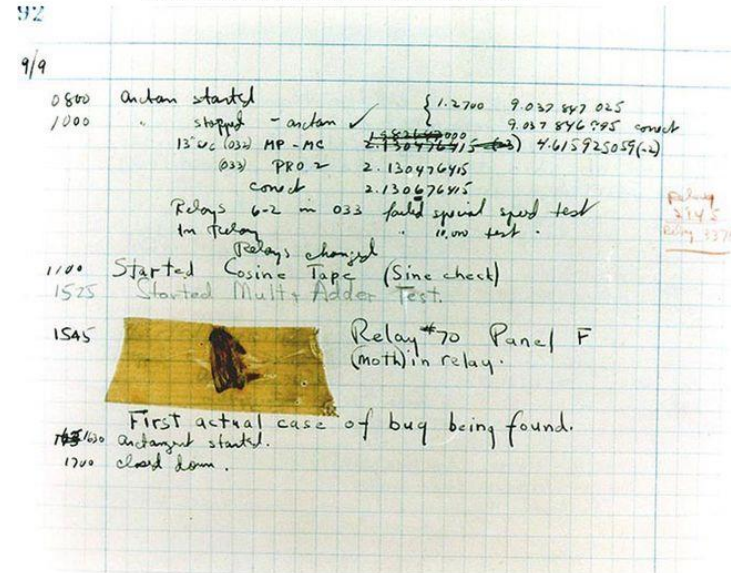


## Bug:

- Story: named after a moth
- Two types: syntax and logic errors
  - `prnt("I could forgotten something.")`



Photo # NH 96566-KN (Color) First Computer "Bug", 1947



# ► High Level Languages



```
print("Clarusway Rocks")
```

# Find bugs and write the correct ones to right hand side:

## Instructions

```
print(5, 7;  
prnt(1, 2);  
print("hello world);
```



Students, draw anywhere on this slide!



# Libraries/Packages/Frameworks



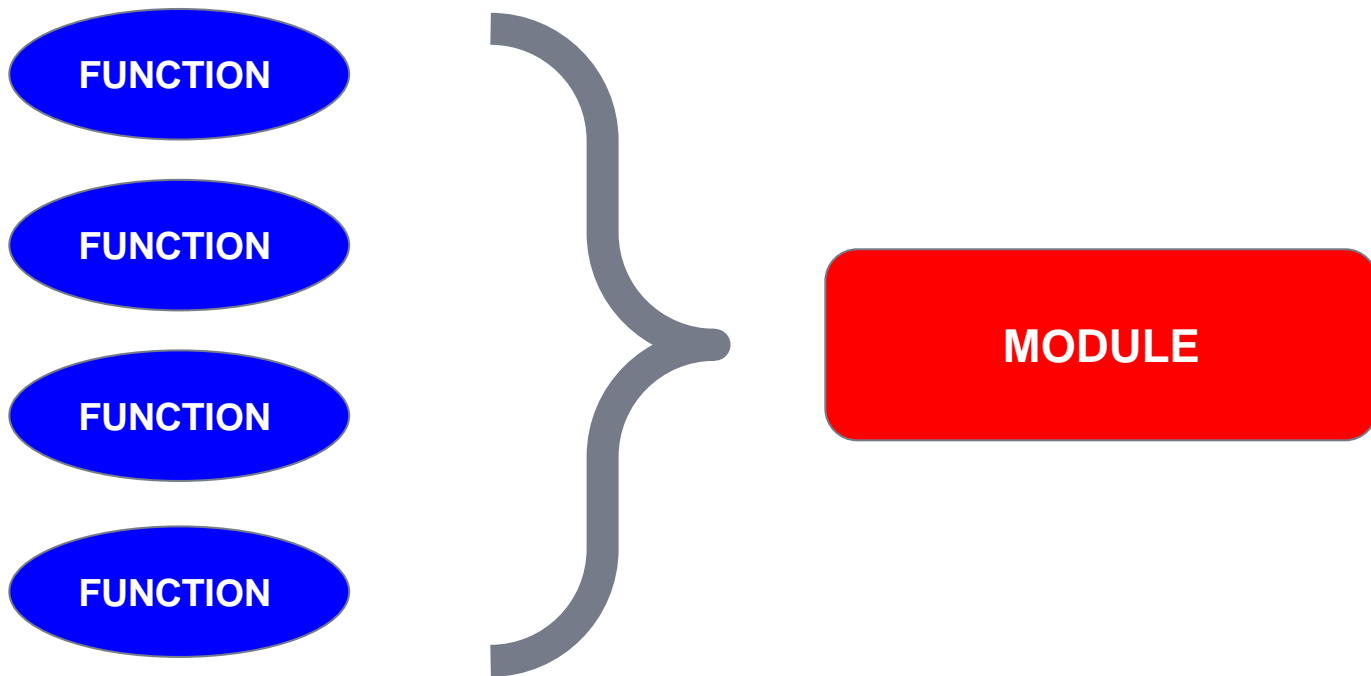
## **Library:**

A software library generally consists of pre-written code, classes, procedures, scripts, configuration data and more. Typically, a developer might manually add a software library to a program to achieve more functionality or to automate a process without writing code for it.





# Libraries/Packages/Frameworks





# Libraries/Packages/Frameworks



```
1 def square(x):  
2     return x*x  
3  
4 print(square(4))  
5  
6  
7  
8  
9  
10
```

Run Python10.2

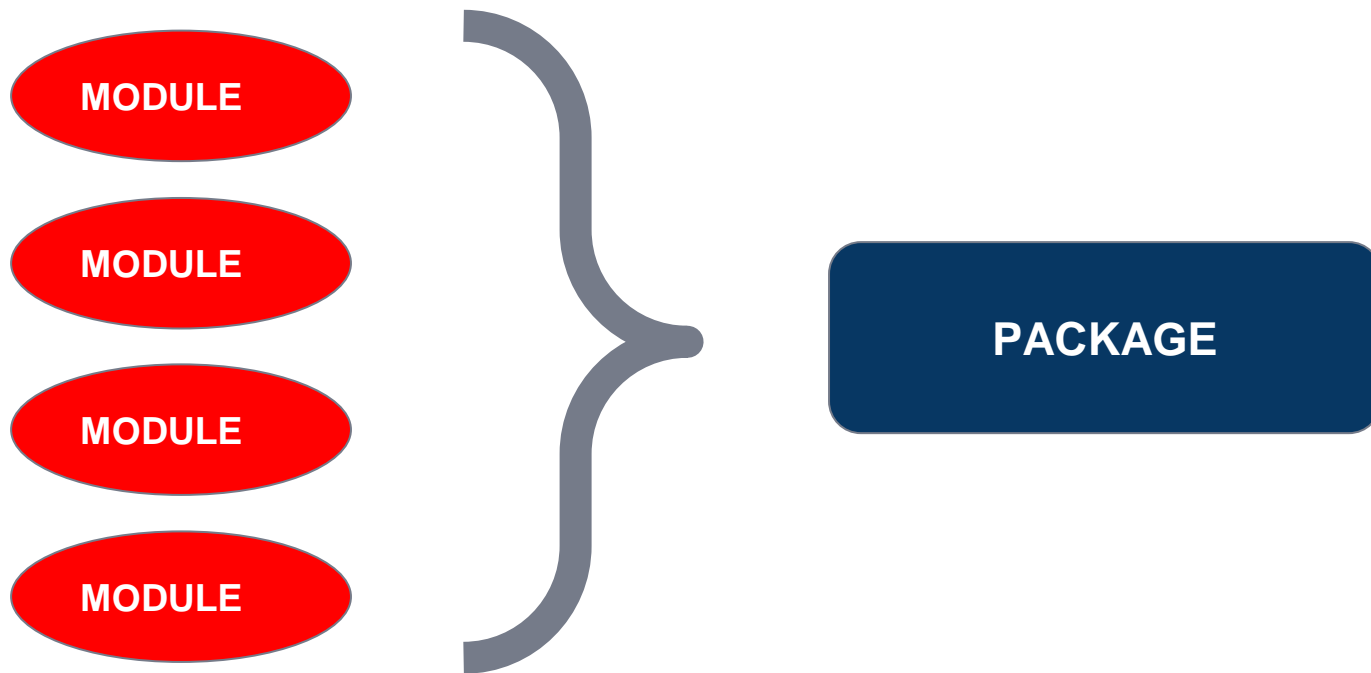
"C:\Users\DK\Desktop\Pyth..."

16

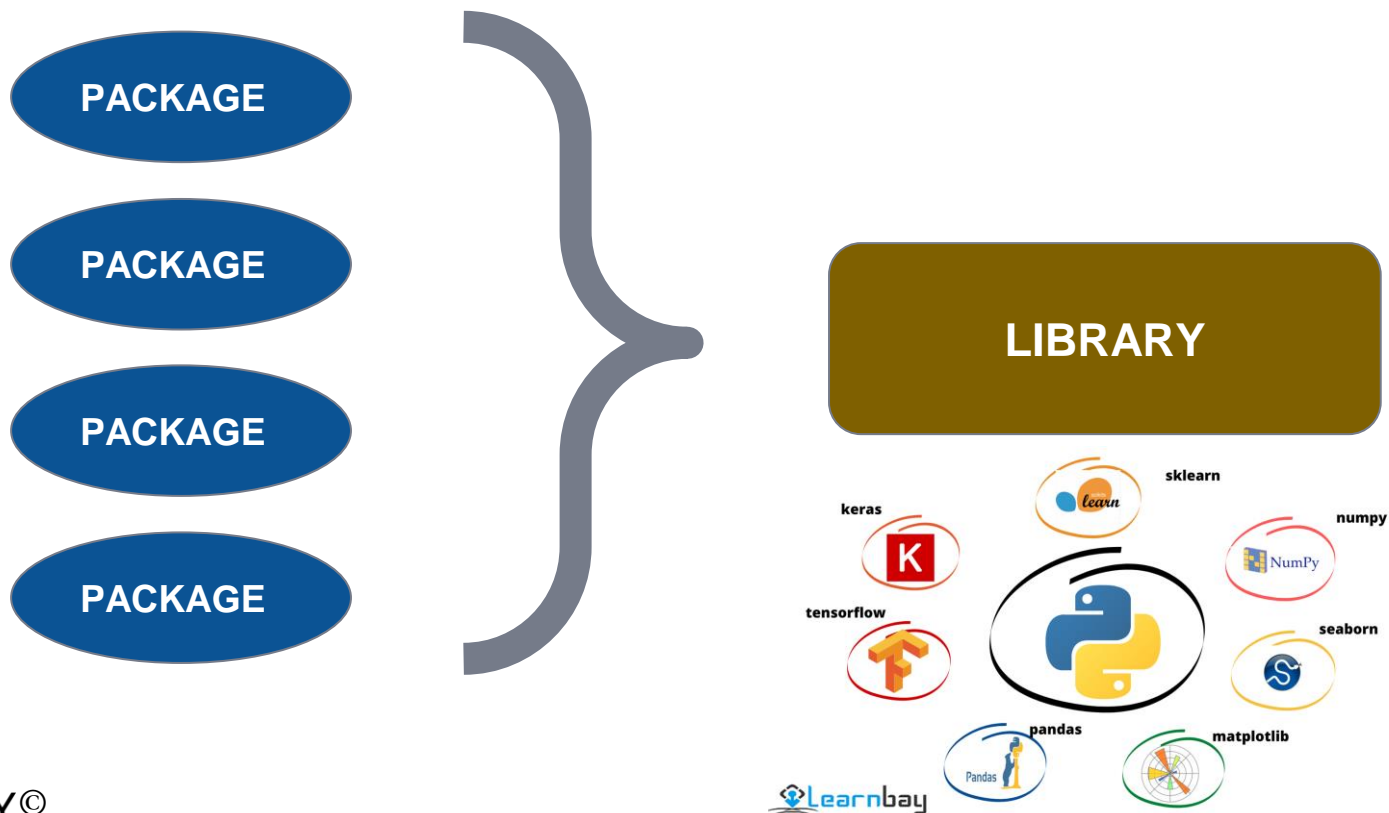
Here we have used "return command" to return the value of function, which is square of (4) i.e 16



# Libraries/Packages/Frameworks



# Libraries/Packages/Frameworks



# Libraries/Packages/Frameworks



## Framework:

Frameworks are software that is developed and used by developers to build applications.

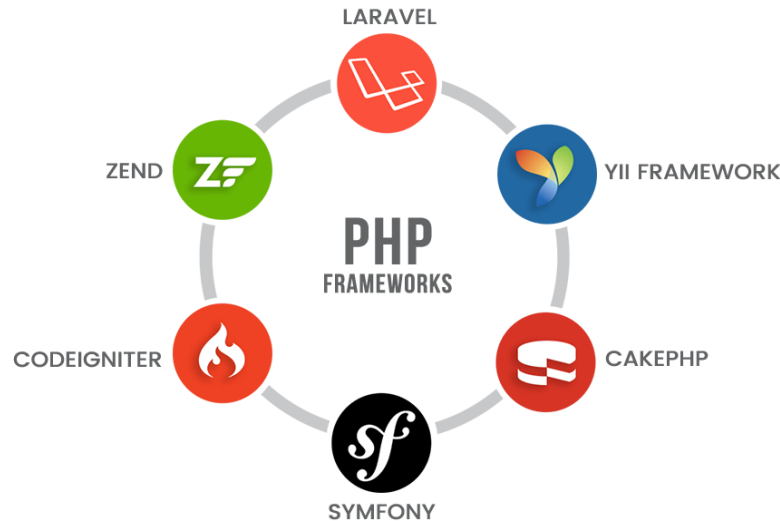


# Libraries/Packages/Frameworks



## Framework:

- Web Application Framework
- Mobile Development Frameworks
- DataScience Frameworks





# Libraries/Packages/Frameworks



Web Framework

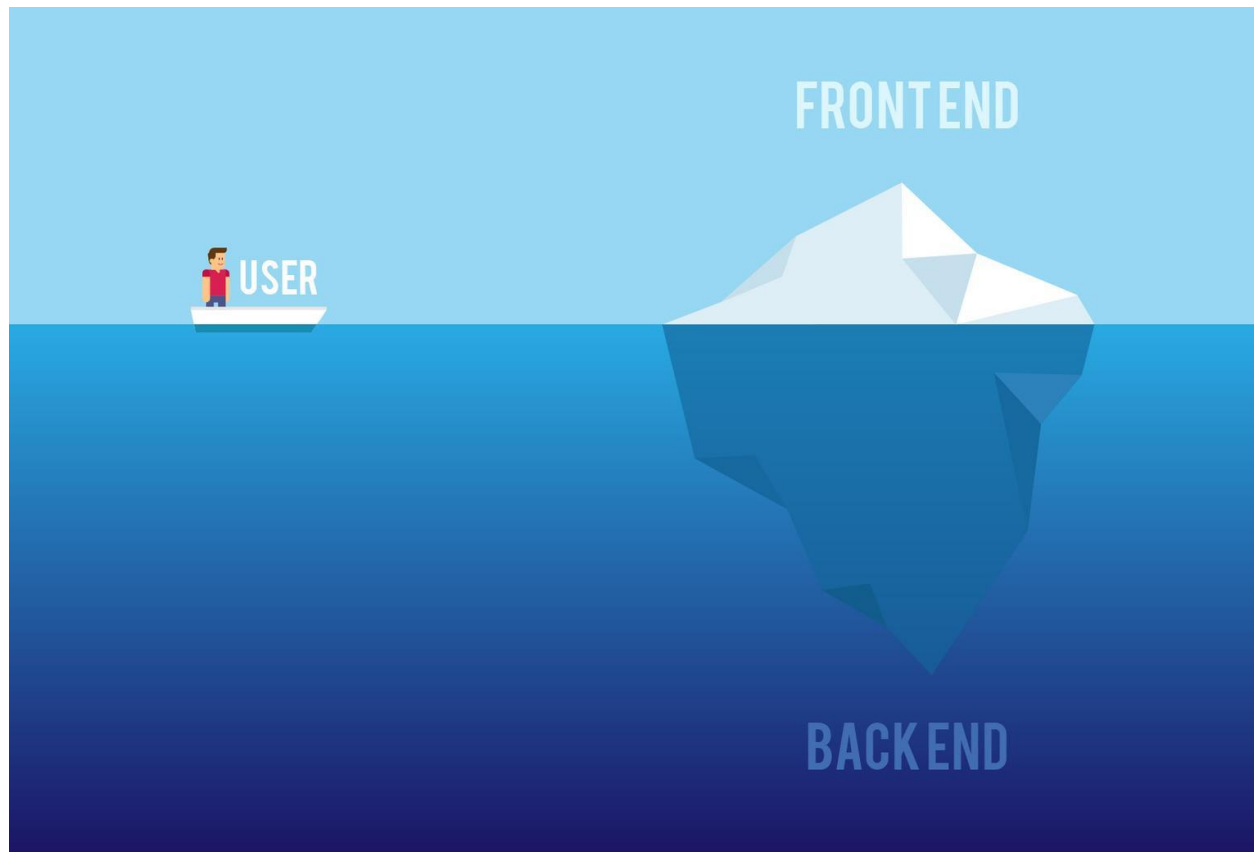
# Backend/Frontend







# Backend/Frontend



# Backend/Frontend



## FRONTEND VERSUS BACKEND

Frontend	Backend
Frontend refers to the client-side of the application.	Backend refers to the server-side of the application.
It is the part of the website users can see and interact with.	It constitutes everything that happens behind the scenes.
It typically includes everything that attributes to the visual aspects of websites.	It generally includes a web server that communicates with a database to serve requests that the frontend presents.
It forms the basis of what users can touch and experience on their web browsers.	It is the brain of the website that is never visible to the end users.
The essentials of frontend web development include HTML, CSS, and JavaScript.	The essentials of backend development include Ruby, Python, Java, .Net, etc.

In one minute,  
write the most  
important thing  
from Software's  
topic.



Students, write your response!

Pear Deck Interactive Slide  
Do not remove this bar



# Computer Fundamentals “Internet”



# Agenda



- ▶ Internet
- ▶ TCP/IP Protocol
- ▶ DNS
- ▶ LAN/WAN



# Kahoot!



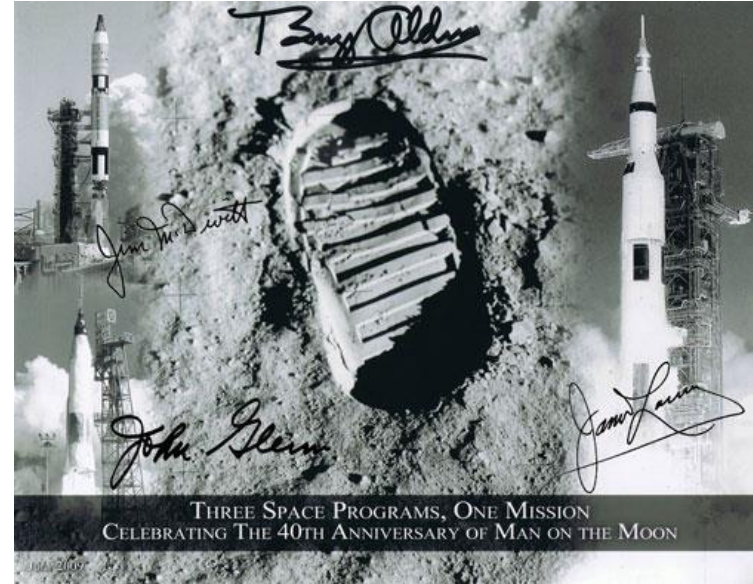
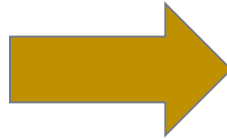
# Internet



- **Network of networks!**



# Internet

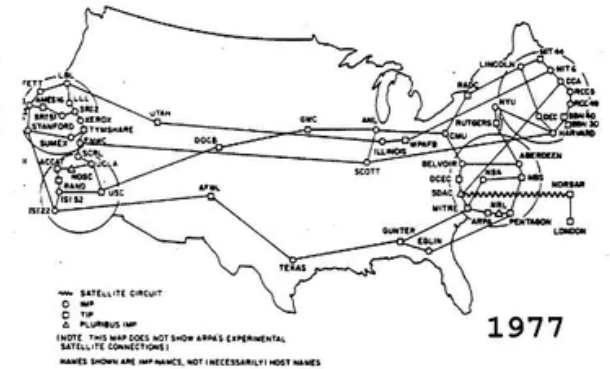
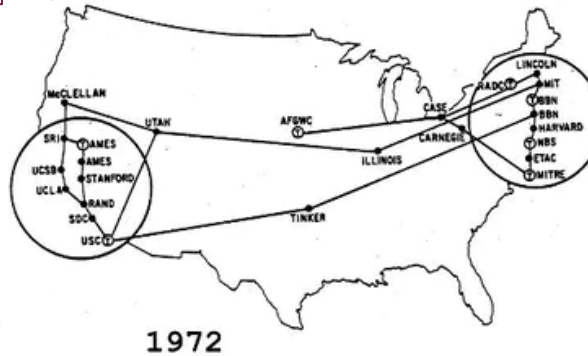
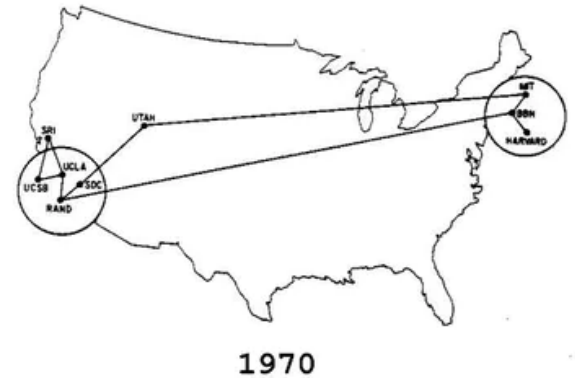
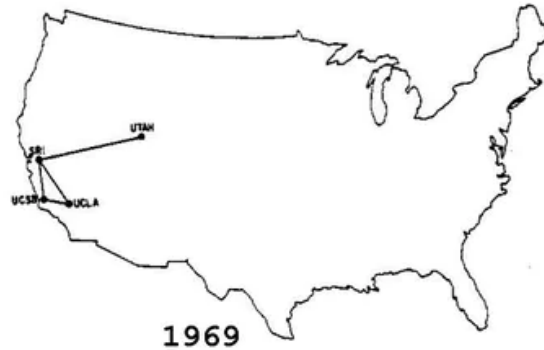


Common Sense

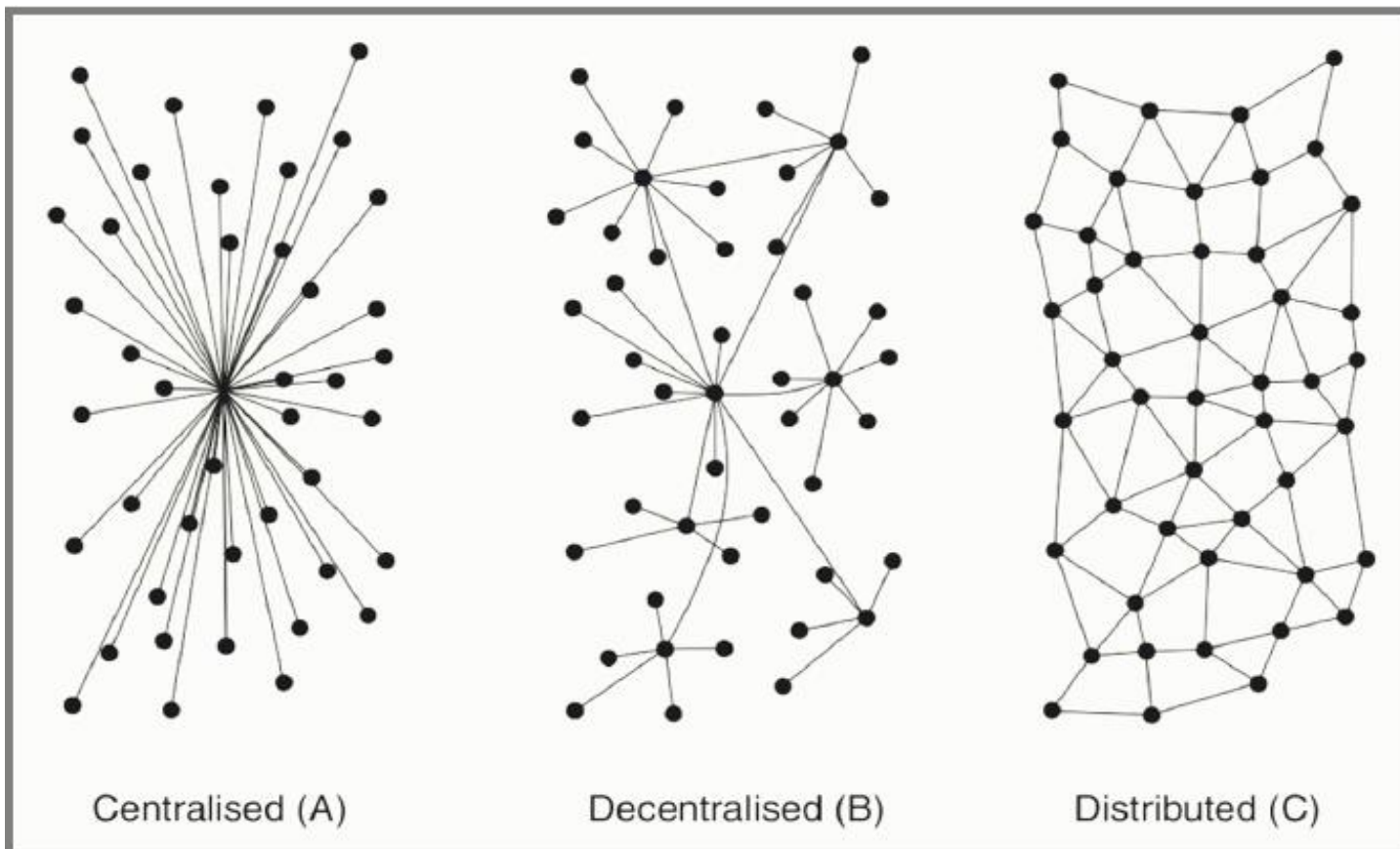


# Internet

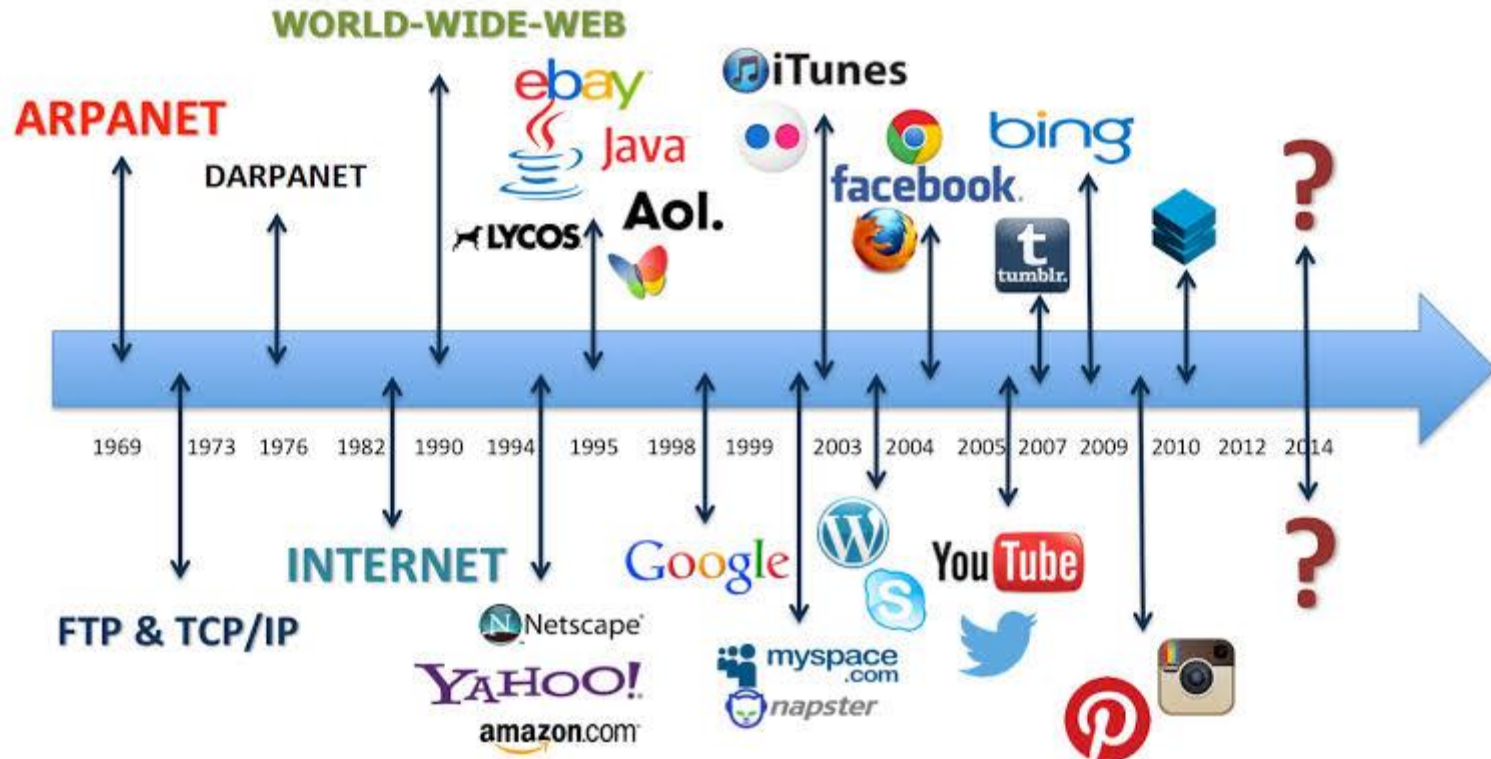
- **ARPANET**  
(The Advanced Research Projects Agency Network)



# Internet



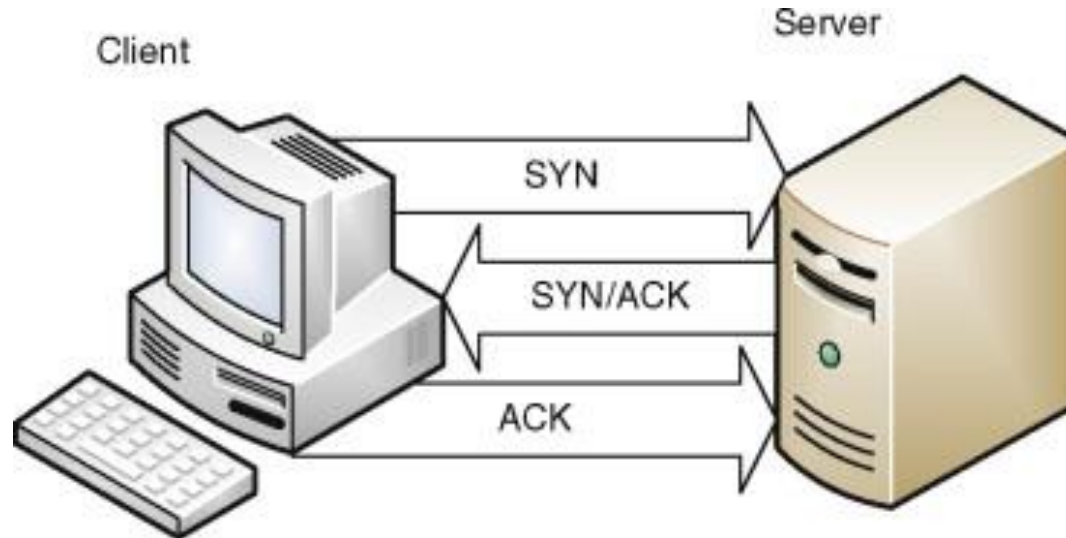
# Internet



# TCP/IP Protocol



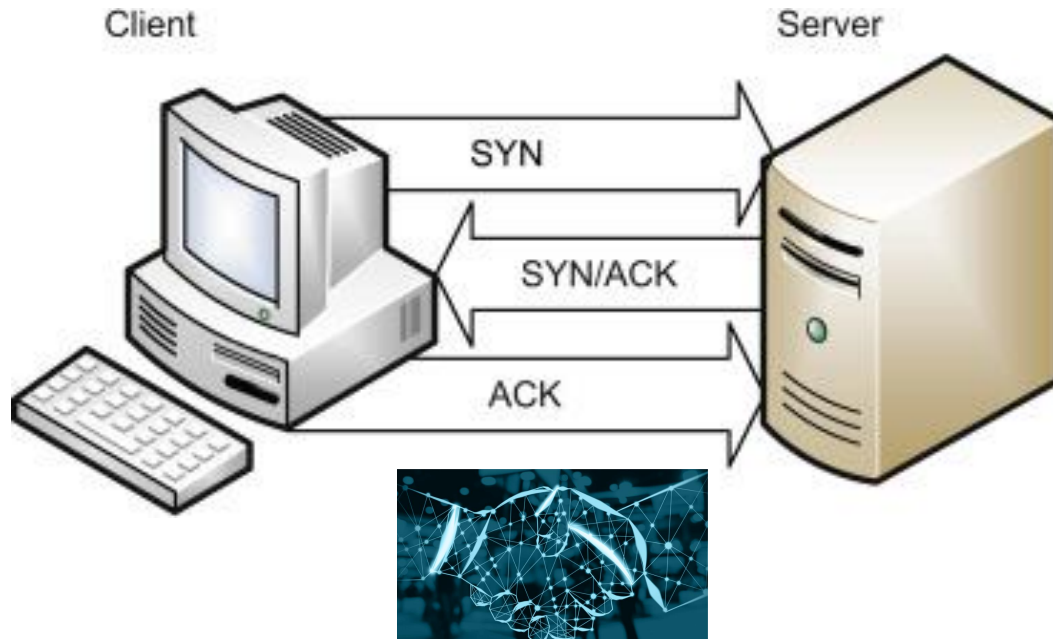
- IP: Internet Protocol
- TCP: Transmission Control Protocol
- Rules for sending information between computers
- Followed by both clients and servers



# TCP/IP Protocol



- Before connection, server and client should shake their hands three times!
  - Three hand shake:





# TCP/IP Protocol

- IP Number:

IPv4 address in dotted-decimal notation

**172 . 16 . 254 . 1**



10101100 , 00010000 , 11111110 , 00000001



8 bits

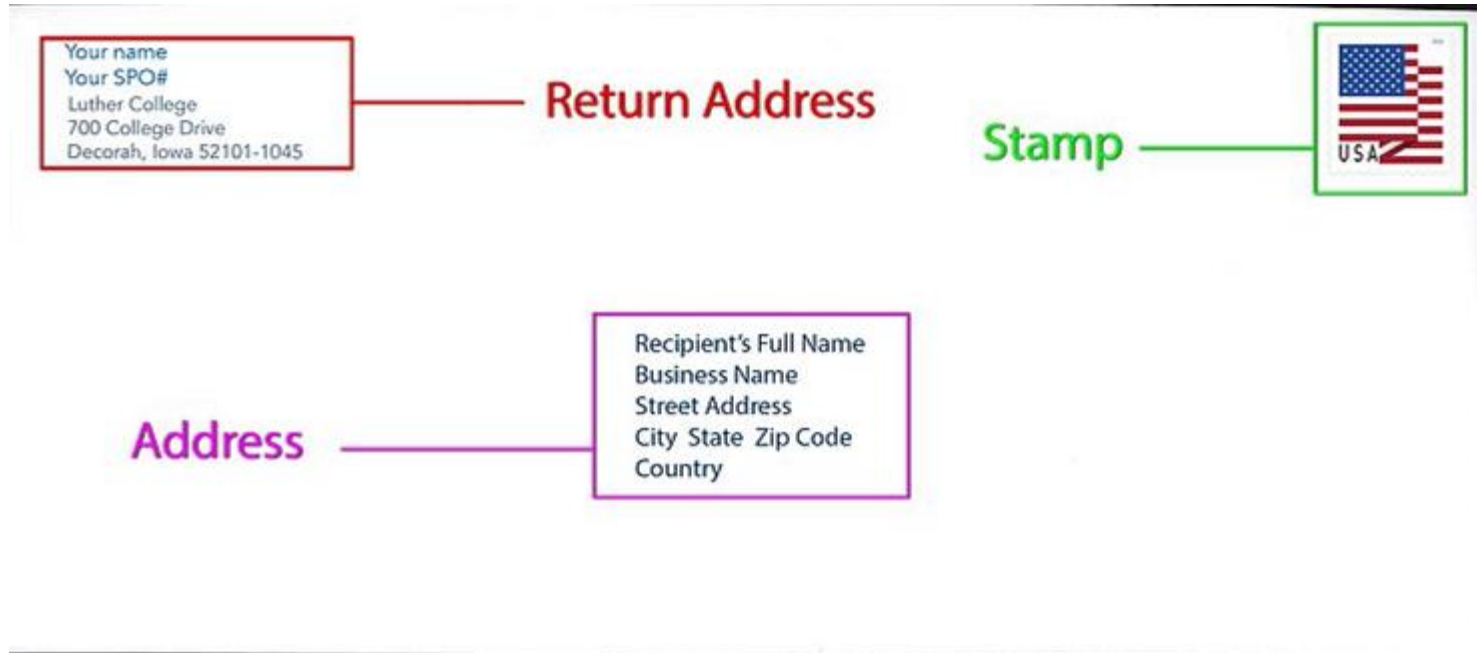


32 bits (4 bytes)

4,294,967,296

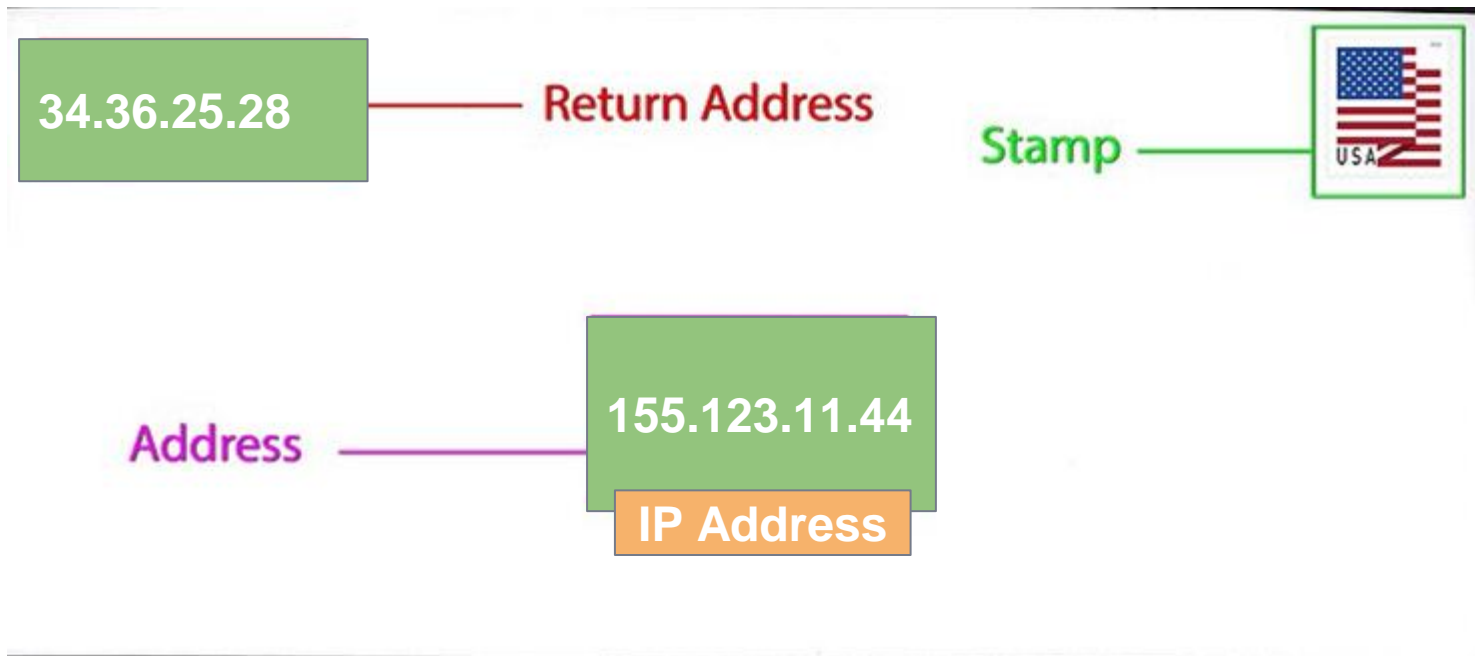


# TCP/IP Protocol





# TCP/IP Protocol



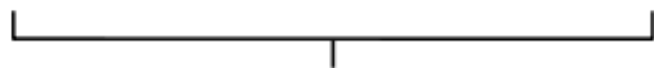




# TCP/IP Protocol

An IPv6 address 128 bit (in hexadecimal)

**2001:0DB8:AC10:FE01:0000:0000:0000:0000**



**2001:0DB8:AC10:FE01::**

Zeroes can be omitted

0010000000000001:0000110110111000:1010110000010000:1111111000000001:

0000000000000000:0000000000000000:0000000000000000:0000000000000000



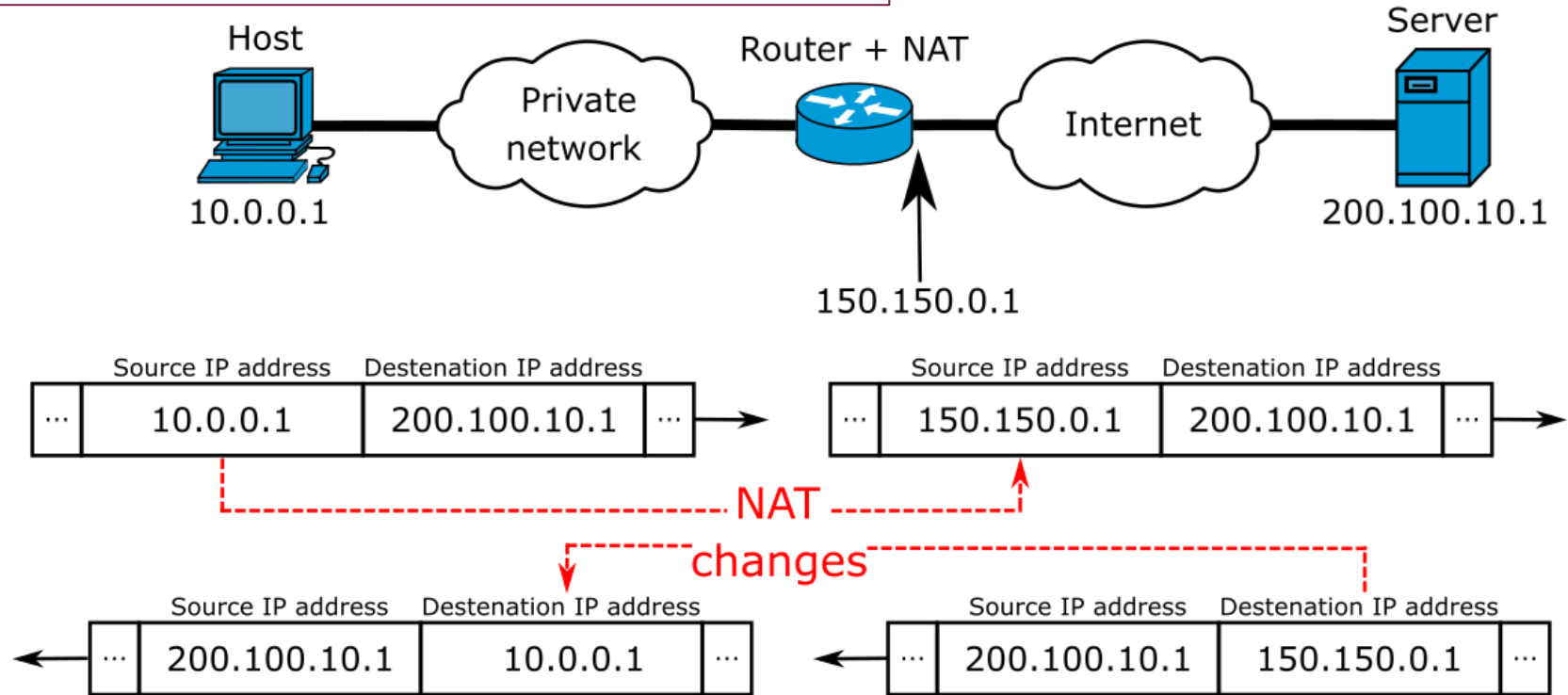
# TCP/IP Protocol



# TCP/IP Protocol

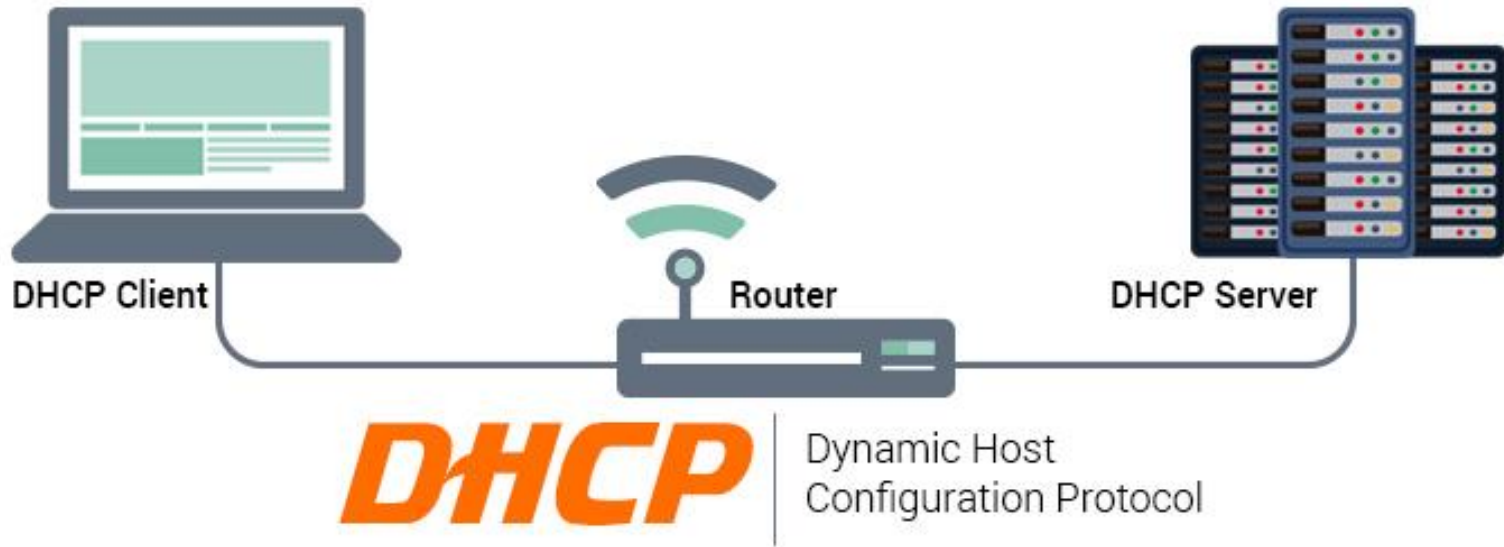


- NAT : Network Address Translation**





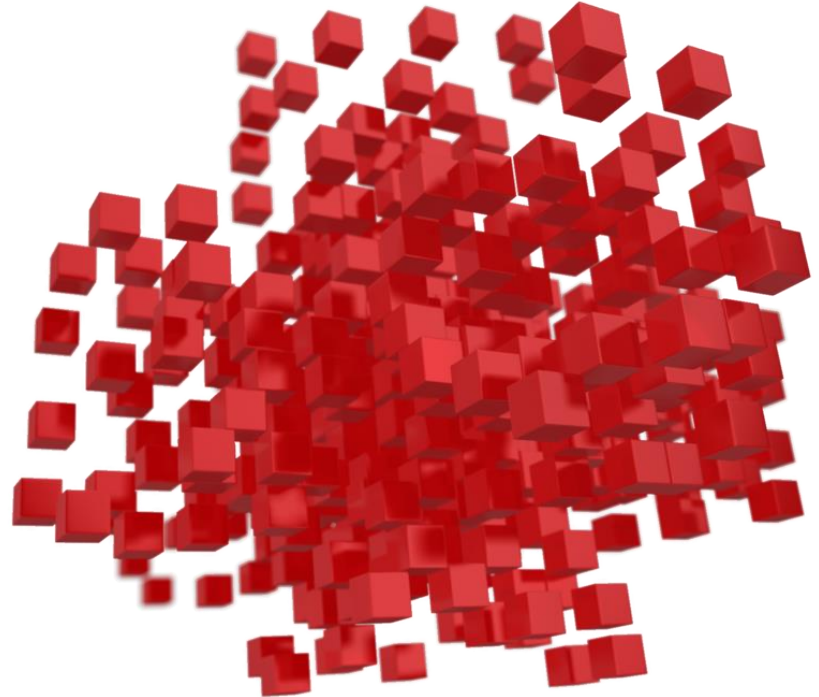
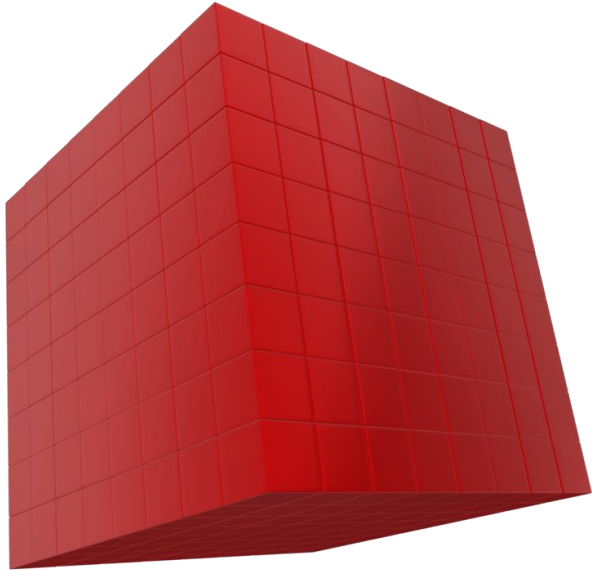
# TCP/IP Protocol



# TCP/IP Protocol

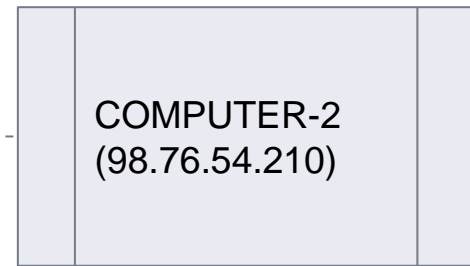
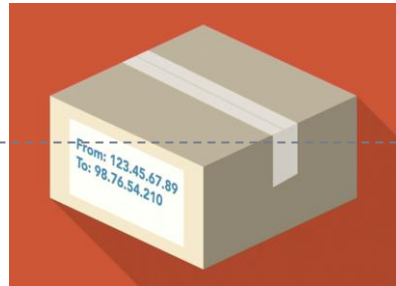
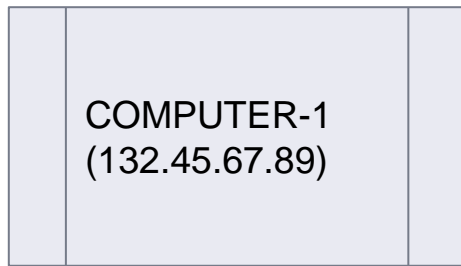


- Packages





# TCP/IP Protocol



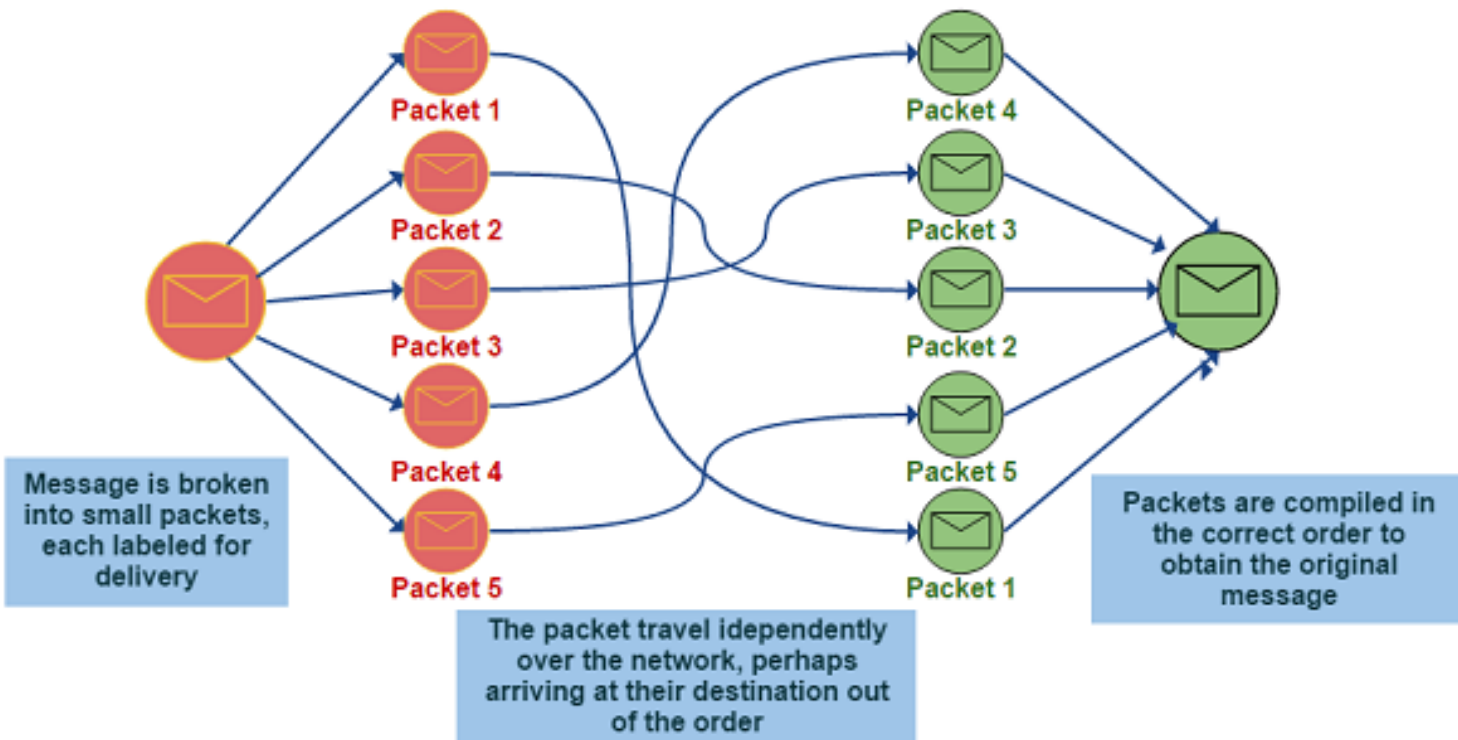
FROM 132.45.67.89 TO 98.76.54.210 2KB PACKET#1

FROM 132.45.67.89 TO 98.76.54.210 2KB PACKET#2

FROM 132.45.67.89 TO 98.76.54.210 2KB PACKET#3

FROM 132.45.67.89 TO 98.76.54.210 2KB PACKET#4

# TCP/IP Protocol

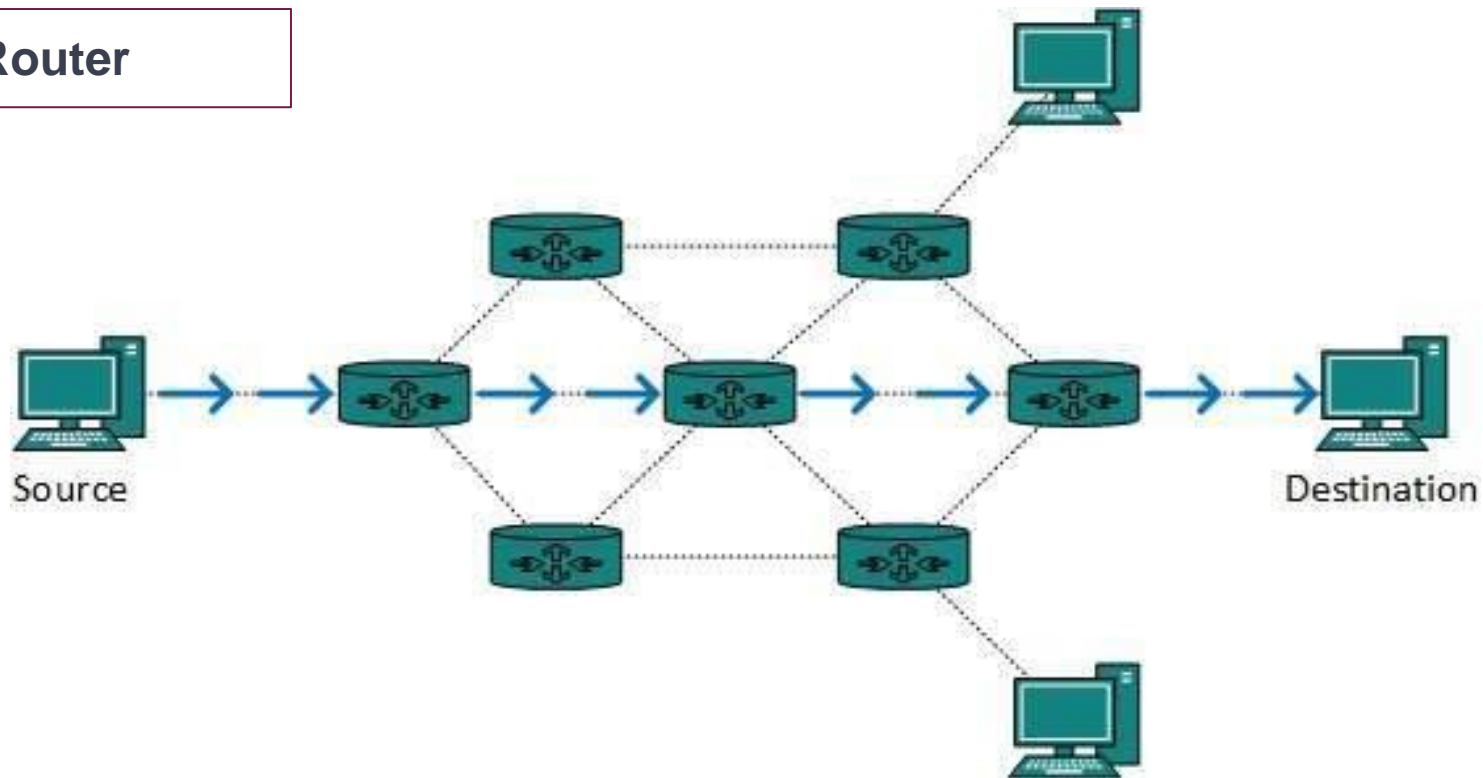


## Internet Packets Transmission

# TCP/IP Protocol



- Router

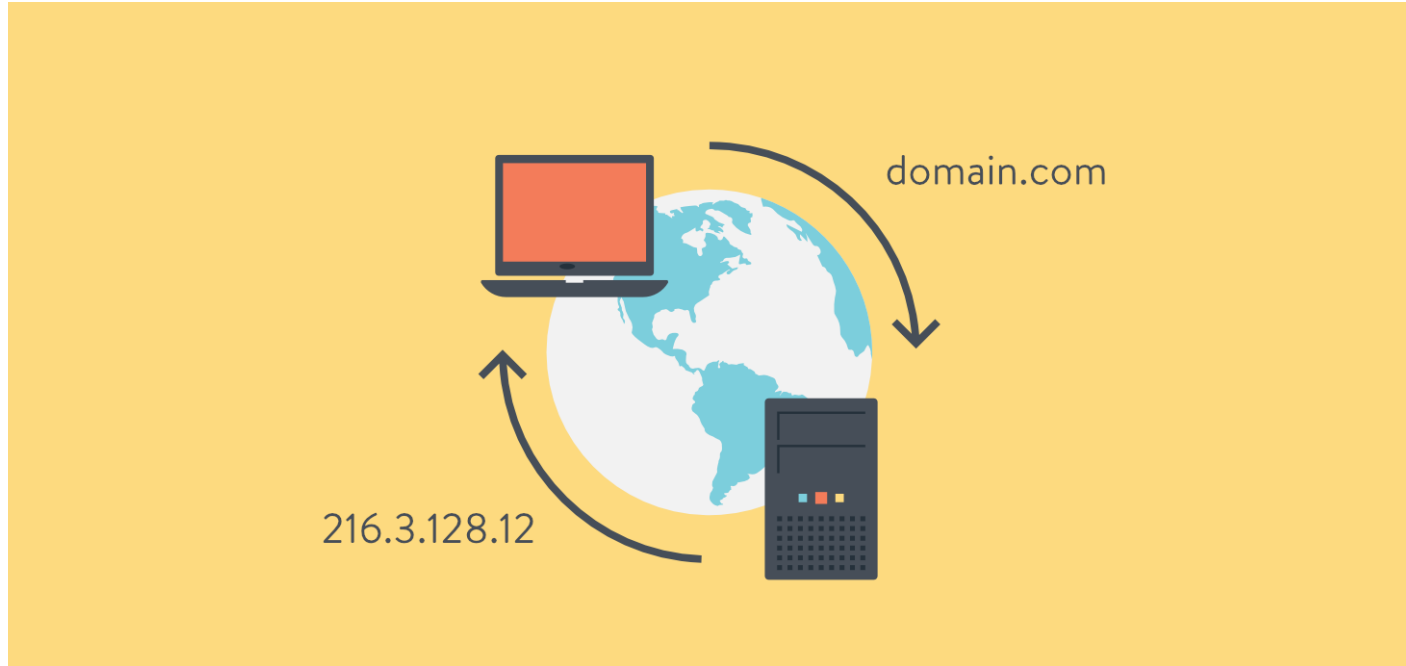




# DNS



- **Domain Name Server**



# DNS



- URL: Uniform Resource Locator

**https://www.example.com**

**Protocol  
(scheme)**

**Sub-  
domain**

**Domain  
name**

**Top level  
domain (TLD)**

# DNS

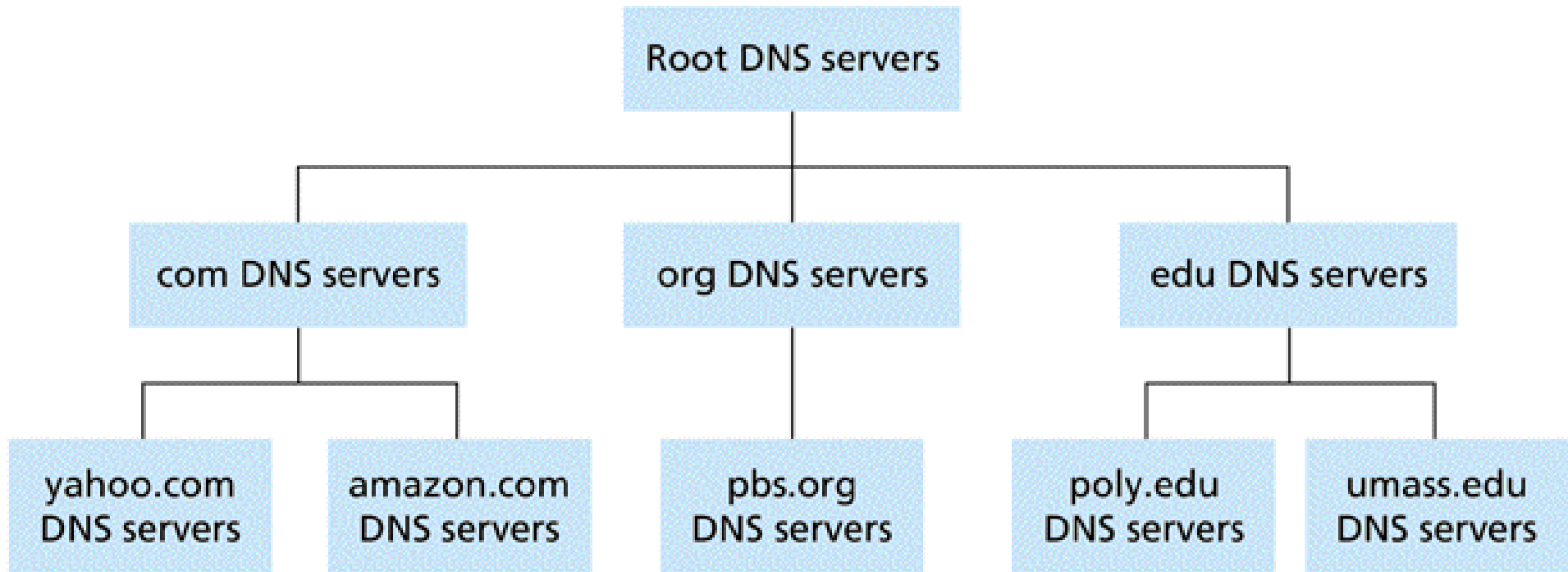


- Root DNS



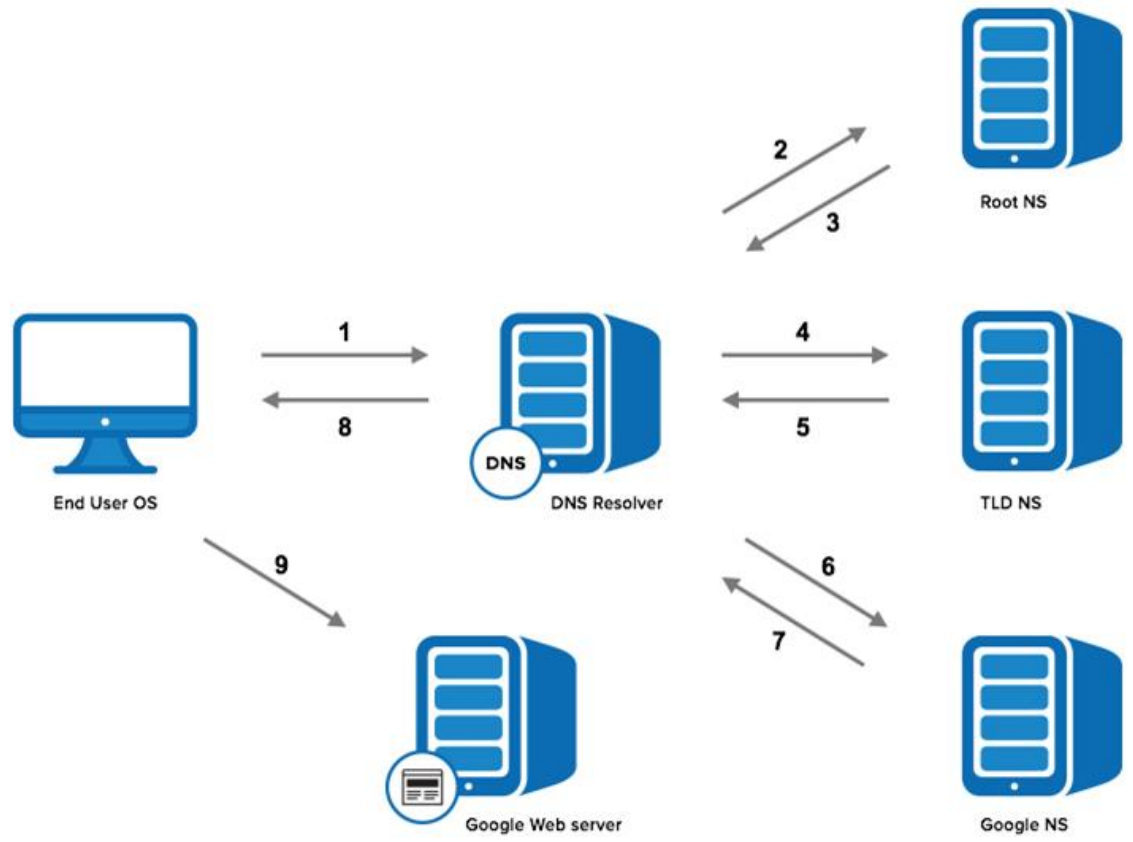


# DNS





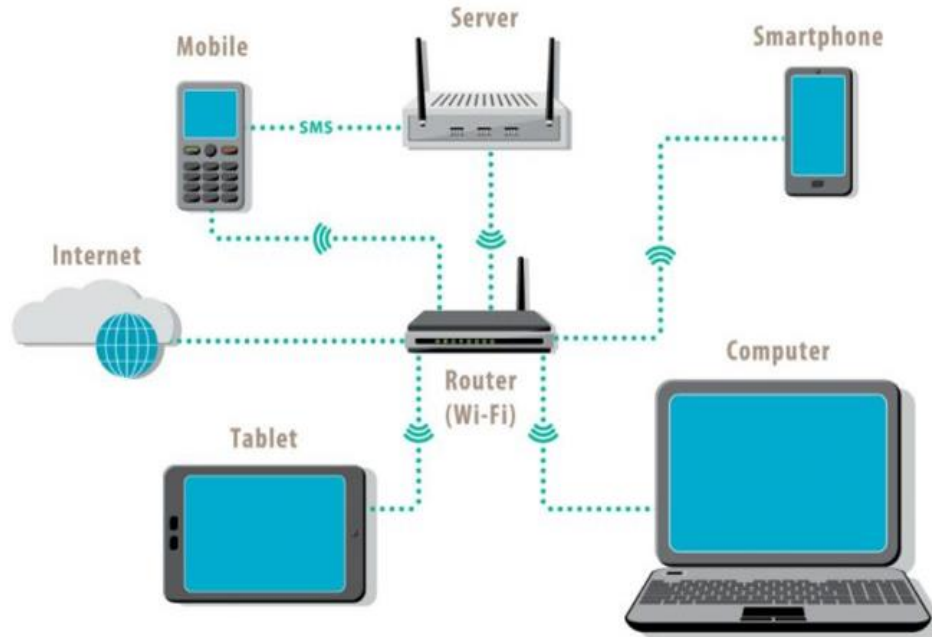
# DNS



# LAN/WAN



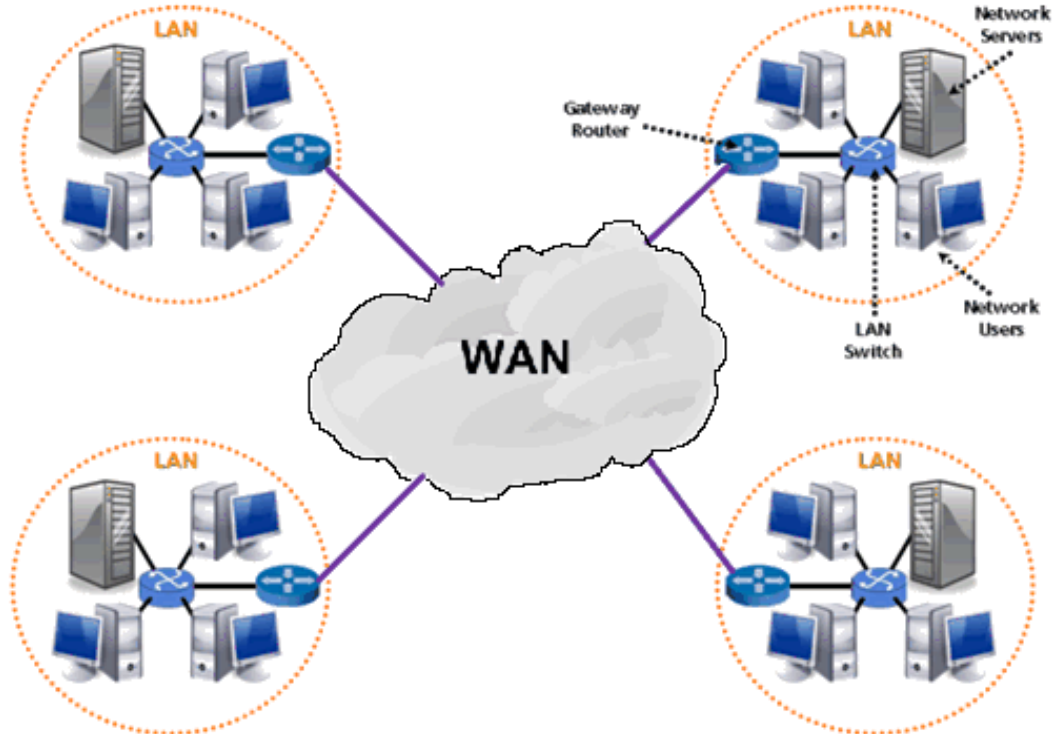
- **LAN: Local Area Network**



# LAN/WAN



- **WAN: Wide Area Network**

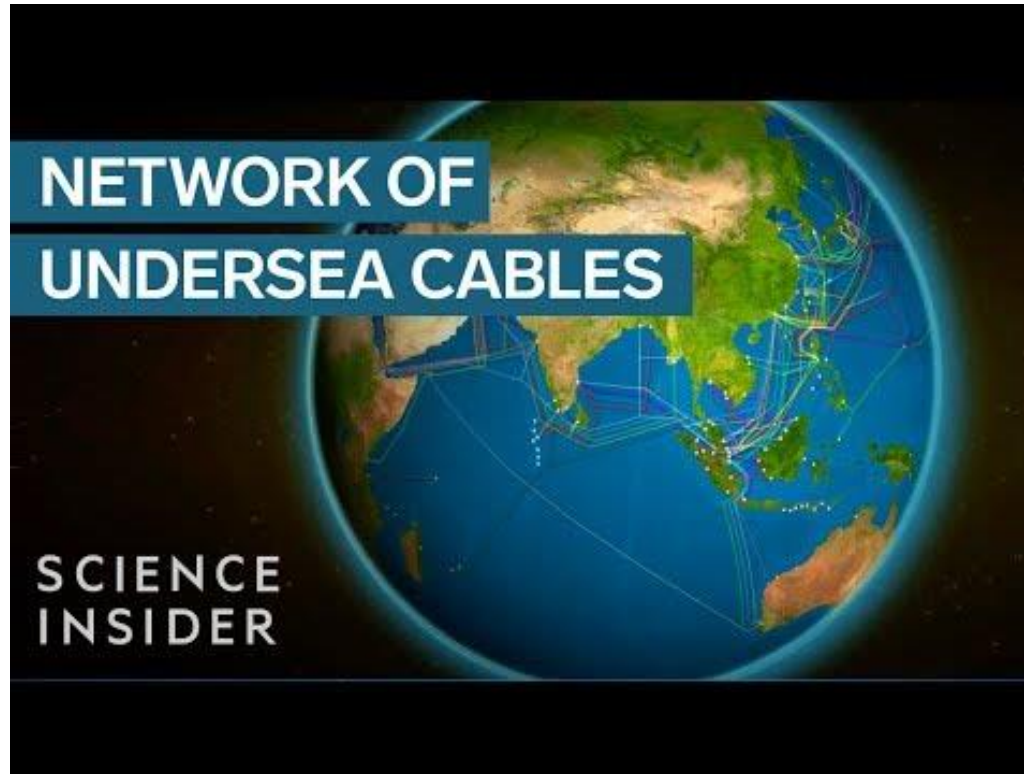


# Let's Practice



```
clarus-linux@professor: ~  
clarus-linux@professor:~$ traceroute clarusway.com  
traceroute to clarusway.com (54.164.151.235), 64 hops max  
 1  192.168.1.1  2,418ms  110,443ms  1,356ms  
 2  212.57.0.115  7,721ms  9,578ms  10,913ms  
 3  10.36.253.221  9,185ms  *  *  
 4  10.58.19.21  12,654ms  *  *  
 5  10.58.19.30  12,150ms  *  *  
 6  10.40.141.12  8,843ms  *  *  
 7  10.36.6.2  11,598ms  *  *  
 8  195.22.206.0  140,022ms  *  *  
 9  195.22.206.63  132,805ms  *  *  
10  54.239.111.232  144,067ms  *  *  
11  52.93.114.45  136,734ms  *  *  
12  52.93.28.110  143,479ms  *  *  
13  *  *  *  
14  *  ^C  
clarus-linux@professor:~$
```





# Please write what is

- Internet
- IP address
- TCP/IP
- DNS



Pear Deck



Students, write your response!

Pear Deck Interactive Slide  
Do not remove this bar

Circle how you are feeling:



 Pear Deck



Students, draw anywhere on this slide!

Pear Deck Interactive Slide  
Do not remove this bar



# THANKS!

## Any questions?

You can find me at:

- ▶ @Jamil
- ▶ jamil@clarusway.com
- ▶ @Tomy
- ▶ tomy@clarusway.com

