
Application Layer and
Presentation Layer
Module 5

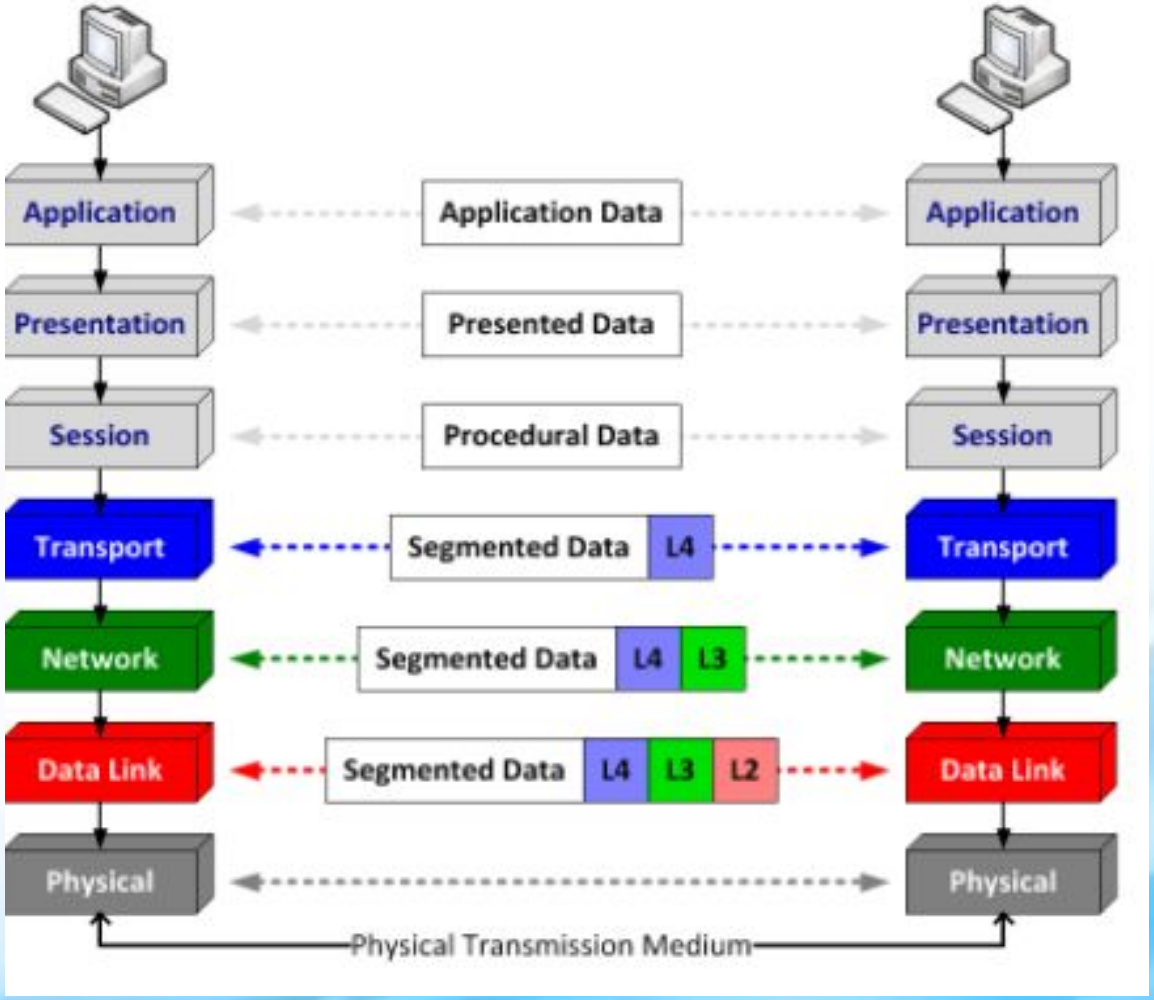
Application Layer

Application Layer

- The Application layer is the seven level of the seven layer OSI model.
- Application layer provides services to the user
- Communication at this layer is logical not physical

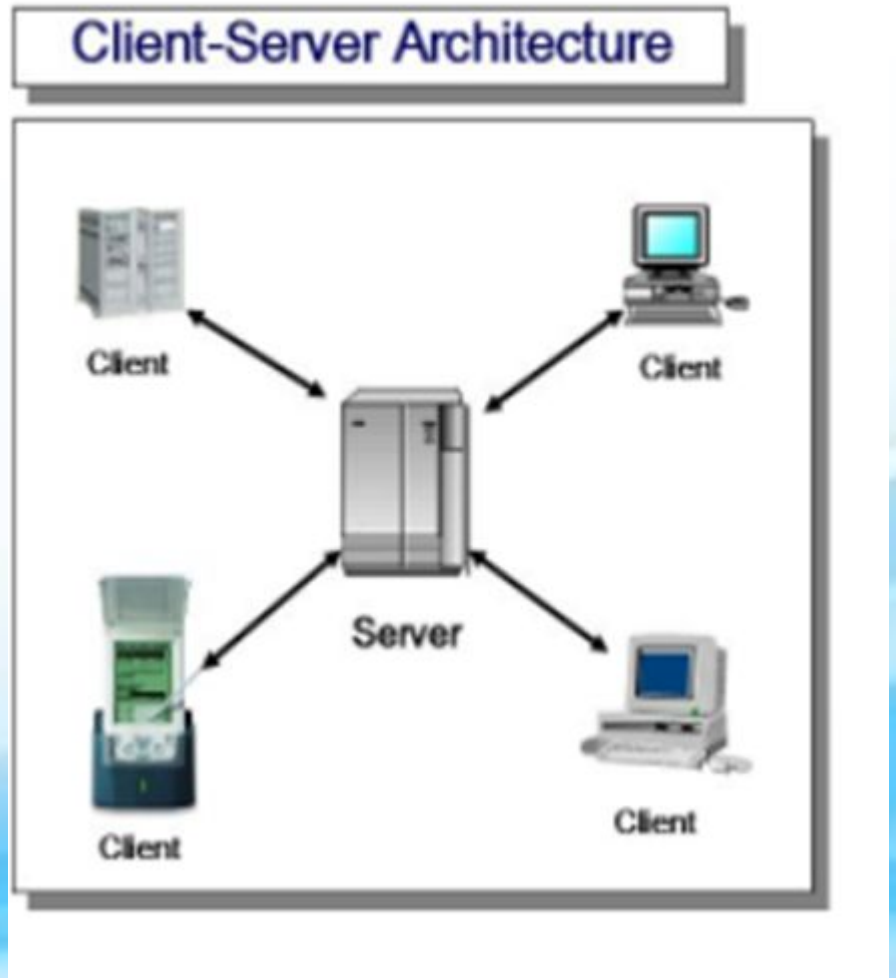
Providing Services

- Flexibility of the application layer allows the application protocols to be easily added

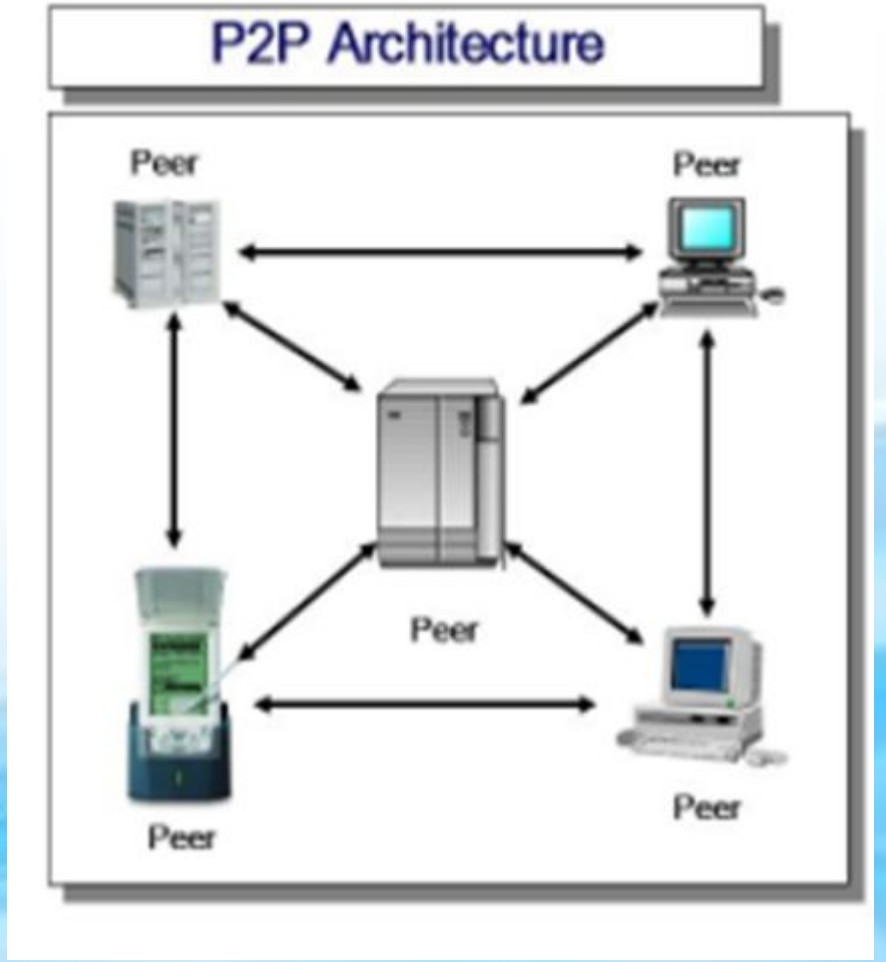


Application layer Paradigm

- Tradition Paradigm : Client Server Model
  - Server is the service provider
  - Client asks for services
  - Server has to be running all the time
  - Role of both are entirely different
  - WWW, HTTP, FTP, SSH, email
  - Communication at the application layer is between the 2 application processes: client and server
  - Client initialises the communication by sending request
  - Server waits for he request from client
  - Server handles the request received from a client, prepares result and sends the result back to the client



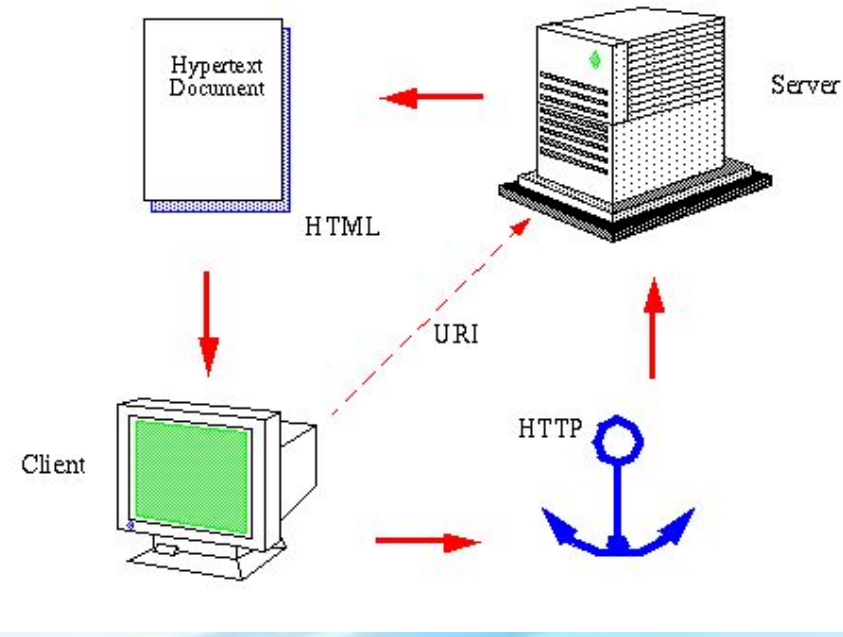
- New Paradigm: Peer to Peer
  - Responsibility is shared b/w peers
  - Easily scalable and cost effective
  - Problem is acceptability and adaptability
  - BitTorrent, Skype, Internet telephony, Cryptocurrency



Using services of Transport Layer

- Transport layer protocol : **UDP** (User Datagram Protocol), **TCP** (Transmission Control Protocol) and **SCTP(**Stream Control Transmission Protocol)
- The choice of the transport layer protocol effects the capability of the application
- **UDP** – connectionless, unrealiable, datagram service, message – oriented
- **TCP Protocol-** connection-oriented, reliable, byte-stream service/ stream- oriented.
- **SCTP** - **Protocol-** connection-oriented, reliable, message oriented

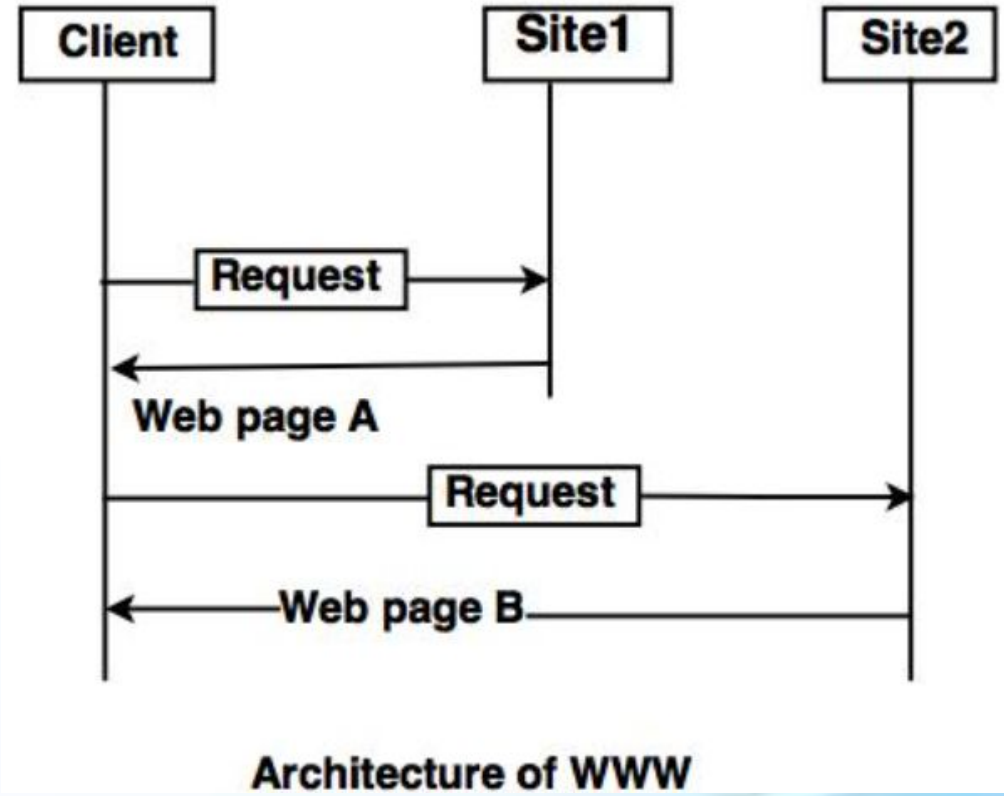
Standard Client Server Applications



A)World Wide Web and HTTP

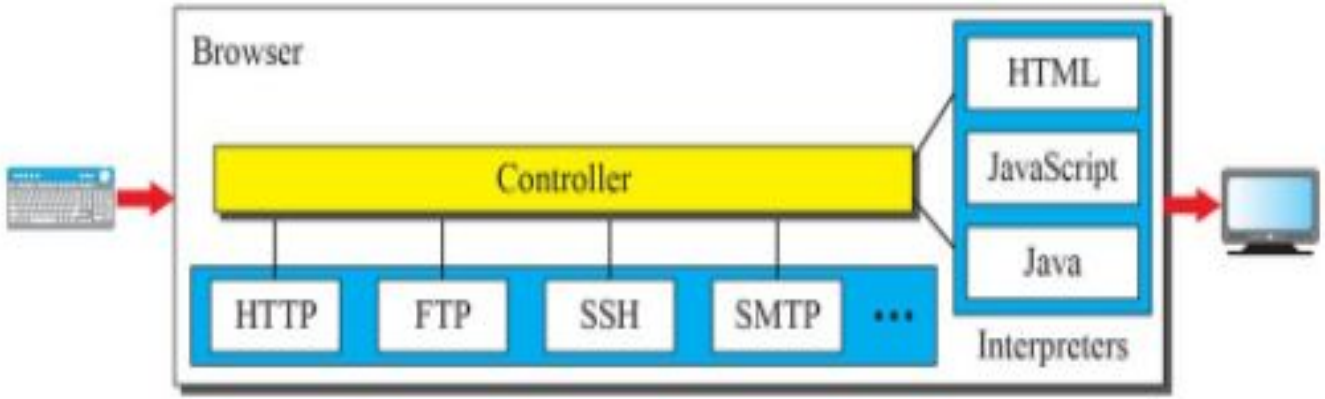
- The World Wide Web (WWW) is a collection of documents and other web resources which are identified by URLs, interlinked by hypertext links, accessed and searched by browsers via the Internet.
- World Wide Web is also called the Web and it was invented by Tim Berners-Lee in 1989.
- Website is a **collection of web pages** belonging to a particular organization.
- It is basically a way of exchanging information between computers on the Internet.

- Information is stored in these Web pages
- Distributed
- Linked



Architecture of WWW

- **Architecture**
  - Distributed client server – distributed over many locations called **sites**
  - Client uses browser to access the service by the server
  - Webpage – simple or composite
    - Each page is a file with name and address



- **Web Client (browser)**
  - Browser interpret and display a web page
  - 3 parts : controller, client protocol and interpreter

- **Web Server**
  - Web page is stored at the server
  - When request arrives, document is sent to the client
  - Stores pages in cache memory
  - Multithreading, multiprocessing -> serve more than one request at a time
  - Apache, Ms IIS

- **Uniform Resource Locator (URL)**

A URL for HTTP (or HTTPS) is normally made up of three or four components:

1. **A scheme.** The scheme identifies the protocol to be used to access the resource on the Internet. It can be HTTP (without SSL) or HTTPS (with SSL).
2. **A host.** The host name identifies the host that holds the resource. For example, www.example.com. A server provides services in the name of the host, but hosts and servers do not have a one-to-one mapping. Refer to Host names.

Host names can also be followed by a **port number**. Refer to Port numbers. Well-known port numbers for a service are normally omitted from the URL. Most servers use the well-known port numbers for HTTP and HTTPS, so most HTTP URLs omit the port number.

3. **A path.** The path identifies the specific resource in the host that the web client wants to access. For example, /software/htp/cics/index.html.
4. **A query string.** If a query string is used, it follows the path component, and provides a string of information that the resource can use for some purpose (for example, as parameters for a search or as data to be processed). The query string is usually a string of name and value pairs; for example, term=bluebird. Name and value pairs are separated from each other by an ampersand (&); for example, term=bluebird&source=browser-search.

Web Documents

- Static Documents
  - Fixed content
  - Server side changes to content is possible, but not on the client side
  - HTML, XML, XSL, XHML
- Dynamic Documents
  - Runs an application/ script that creats dynamic document
  - Server returns the result of the program/script as a response to the browser that requested it
  - JSP, AJX

- Active Documents
  - Program/ script to be run at the client
  - Animated graphics
  - Java applets, Java scripts

Features of WWW

- Provides a system for Hypertext information
- Open standards and Open source
- Distributed.
- Mainly makes the use of Web Browser in order to provide a single interface for many services.
- Dynamic
- Interactive
- Cross-Platform

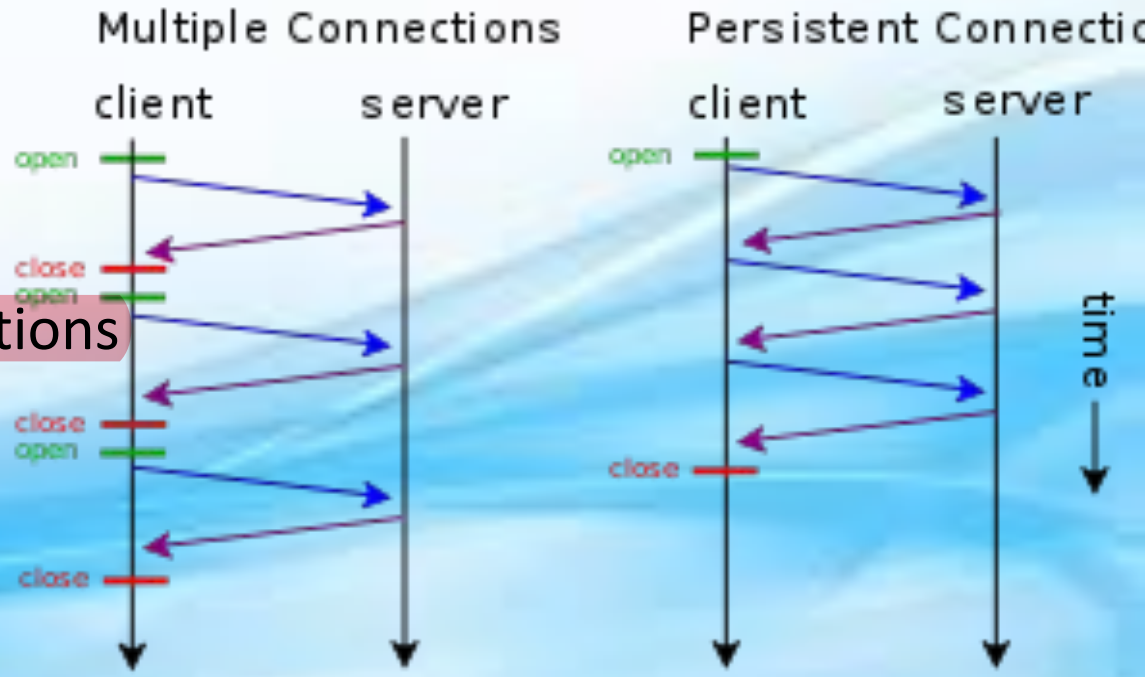
Advantages of WWW

- Given below are the benefits offered by WWW:
- It mainly provides all the information for Free.
- Provides rapid Interactive way of Communication.
- It is accessible from anywhere.
- It has become the Global source of media.
- It mainly facilitates the exchange of a huge volume of data.

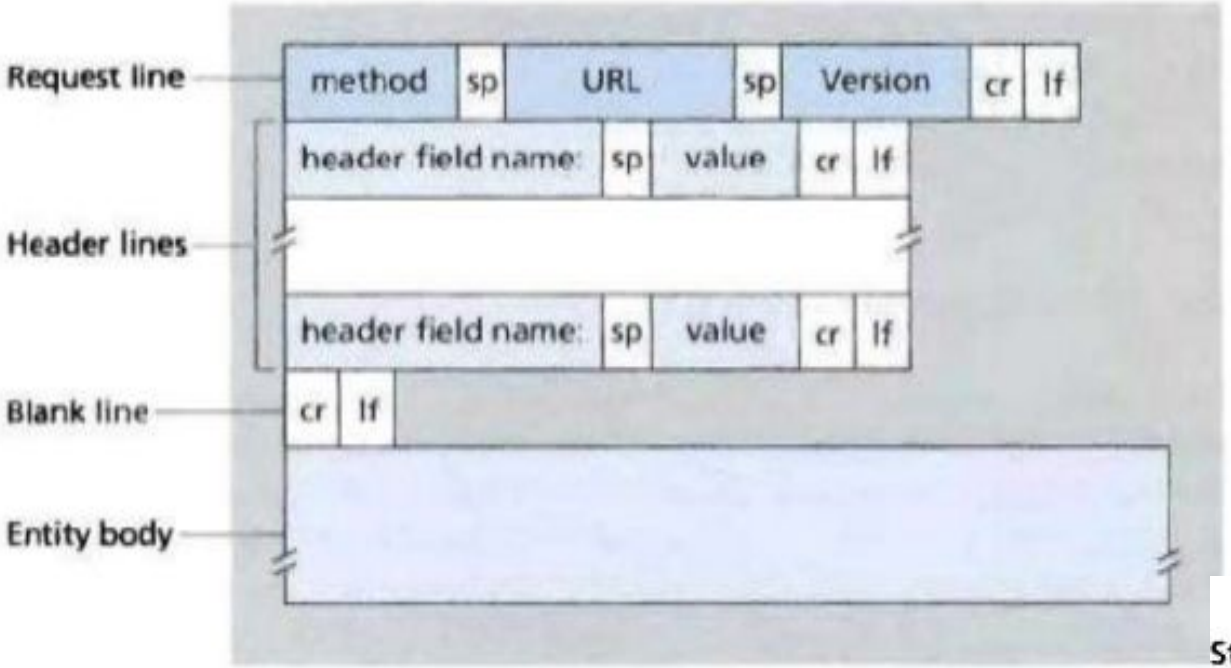
HyperText Transfer Protocol (HTTP)

- It is a protocol used to access the data on the World Wide Web (www).
- Client- server program
- Port 80
- Uses TCP services
- HTTP connections :
  - Multiple Connections/Persistent Connections

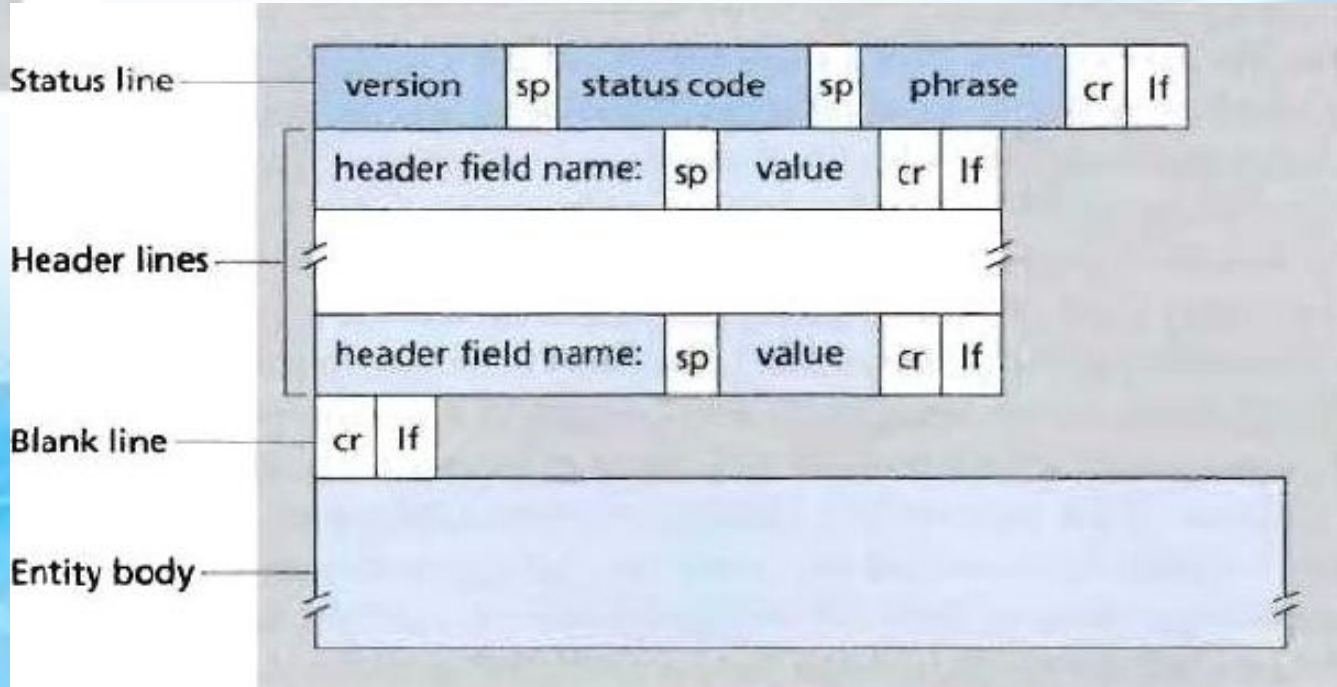
    Nonpersistent connections are the default mode for HTTP/1.0 and persistent connections are the default mode for HTTP/1.1.



HTTP Request Message



HTTP Response Message



Message Format

HTTP *Request* Message

- Request methods

Method	Description
GET	Request to read a Web page
HEAD	Request to read a Web page's header
PUT	Request to store a Web page
POST	Append to a named resource (e.g., a Web page)
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Reserved for future use
OPTIONS	Query certain options

- Request Header Names

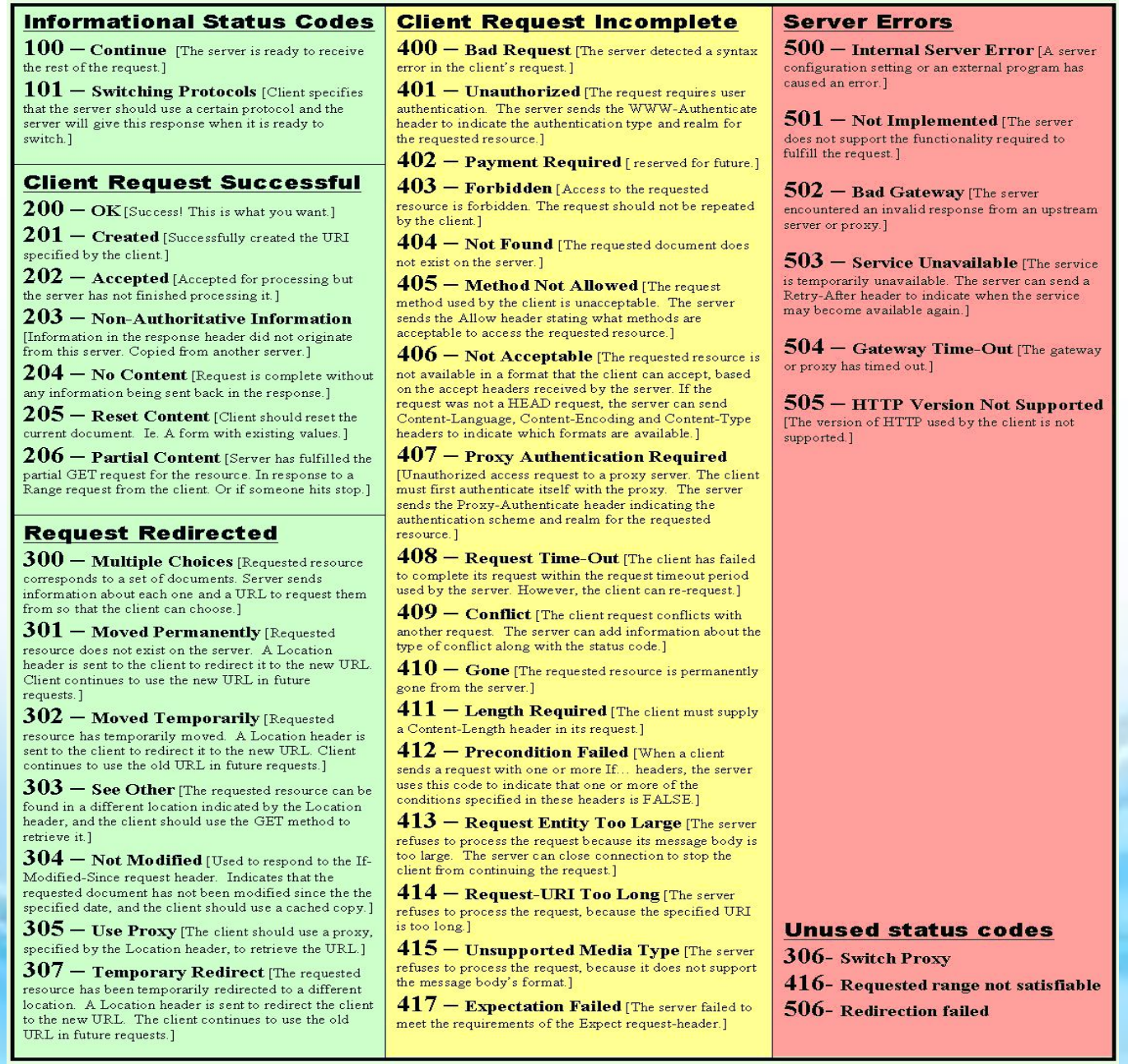
Header	*Description*
User-agent	Identifies the client program
Accept	Shows the media format the client can accept
Accept-charset	Shows the character set the client can handle
Accept-encoding	Shows the encoding scheme the client can handle
Accept-language	Shows the language the client can accept
Authorization	Shows what permissions the client has
Host	Shows the host and port number of the client
Date	Shows the current date
Upgrade	Specifies the preferred communication protocol
Cookie	Returns the cookie to the server (explained later)
If-Modified-Since	If the file is modified since a specific date

HTTP *Response* Message

- Response Header Names

Header	*Description*
Date	Shows the current date
Upgrade	Specifies the preferred communication protocol
Server	Gives information about the server
Set-Cookie	The server asks the client to save a cookie
Content-Encoding	Specifies the encoding scheme
Content-Language	Specifies the language
Content-Length	Shows the length of the document
Content-Type	Specifies the media type
Location	To ask the client to send the request to another site
Accept-Ranges	The server will accept the requested byte-ranges
Last-modified	Gives the date and time of the last change

status code



Informational Status Codes

100 — Continue [The server is ready to receive the rest of the request.]

101 — Switching Protocols [Client specifies that the server should use a certain protocol and the server will give this response when it is ready to switch.]

Client Request Successful

200 — OK [Success! This is what you want.]

201 — Created [Successfully created the URI specified by the client.]

202 — Accepted [Accepted for processing but the server has not finished processing it.]

203 — Non-Authoritative Information [Information in the response header did not originate from this server. Copied from another server.]

204 — No Content [Request is complete without any information being sent back in the response.]

205 — Reset Content [Client should reset the current document. Ie. A form with existing values.]

206 — Partial Content [Server has fulfilled the partial GET request for the resource. In response to a Range request from the client. Or if someone hits stop.]

Request Redirected

300 — Multiple Choices [Requested resource corresponds to a set of documents. Server sends information about each one and a URL to request them from so that the client can choose.]

301 — Moved Permanently [Requested resource does not exist on the server. A Location header is sent to the client to redirect it to the new URL. Client continues to use the new URL in future requests.]

302 — Moved Temporarily [Requested resource has temporarily moved. A Location header is sent to the client to redirect it to the new URL. Client continues to use the old URL in future requests.]

303 — See Other [The requested resource can be found in a different location indicated by the Location header, and the client should use the GET method to retrieve it.]

304 — Not Modified [Used to respond to the If-Modified-Since request header. Indicates that the requested document has not been modified since the the specified date, and the client should use a cached copy.]

305 — Use Proxy [The client should use a proxy, specified by the Location header, to retrieve the URL.]

307 — Temporary Redirect [The requested resource has been temporarily redirected to a different location. A Location header is sent to redirect the client to the new URL. The client continues to use the old URL in future requests.]

Client Request Incomplete

400 — Bad Request [The server detected a syntax error in the client's request.]

401 — Unauthorized [The request requires user authentication. The server sends the WWW-Authenticate header to indicate the authentication type and realm for the requested resource.]

402 — Payment Required [reserved for future.]

403 — Forbidden [Access to the requested resource is forbidden. The request should not be repeated by the client.]

404 — Not Found [The requested document does not exist on the server.]

405 — Method Not Allowed [The request method used by the client is unacceptable. The server sends the Allow header stating what methods are acceptable to access the requested resource.]

406 — Not Acceptable [The requested resource is not available in a format that the client can accept, based on the accept headers received by the server. If the request was not a HEAD request, the server can send Content-Language, Content-Encoding and Content-Type headers to indicate which formats are available.]

407 — Proxy Authentication Required [Unauthorized access request to a proxy server. The client must first authenticate itself with the proxy. The server sends the Proxy-Authenticate header indicating the authentication scheme and realm for the requested resource.]

408 — Request Time-Out [The client has failed to complete its request within the request timeout period used by the server. However, the client can re-request.]

409 — Conflict [The client request conflicts with another request. The server can add information about the type of conflict along with the status code.]

410 — Gone [The requested resource is permanently gone from the server.]

411 — Length Required [The client must supply a Content-Length header in its request.]

412 — Precondition Failed [When a client sends a request with one or more If... headers, the server uses this code to indicate that one or more of the conditions specified in these headers is FALSE.]

413 — Request Entity Too Large [The server refuses to process the request because its message body is too large. The server can close connection to stop the client from continuing the request.]

414 — Request-URI Too Long [The server refuses to process the request, because the specified URI is too long.]

415 — Unsupported Media Type [The server refuses to process the request, because it does not support the message body's format.]

417 — Expectation Failed [The server failed to meet the requirements of the Expect request-header.]

Server Errors

500 — Internal Server Error [A server configuration setting or an external program has caused an error.]

501 — Not Implemented [The server does not support the functionality required to fulfill the request.]

502 — Bad Gateway [The server encountered an invalid response from an upstream server or proxy.]

503 — Service Unavailable [The service is temporarily unavailable. The server can send a Retry-After header to indicate when the service may become available again.]

504 — Gateway Time-Out [The gateway or proxy has timed out.]

505 — HTTP Version Not Supported [The version of HTTP used by the client is not supported.]

Unused status codes

306- Switch Proxy

416- Requested range not satisfiable

506- Redirection failed

HTTP protocol version 1.1 Server Response Codes

http://www.w3.org/Protocols/rfc2616/rfc2616.html

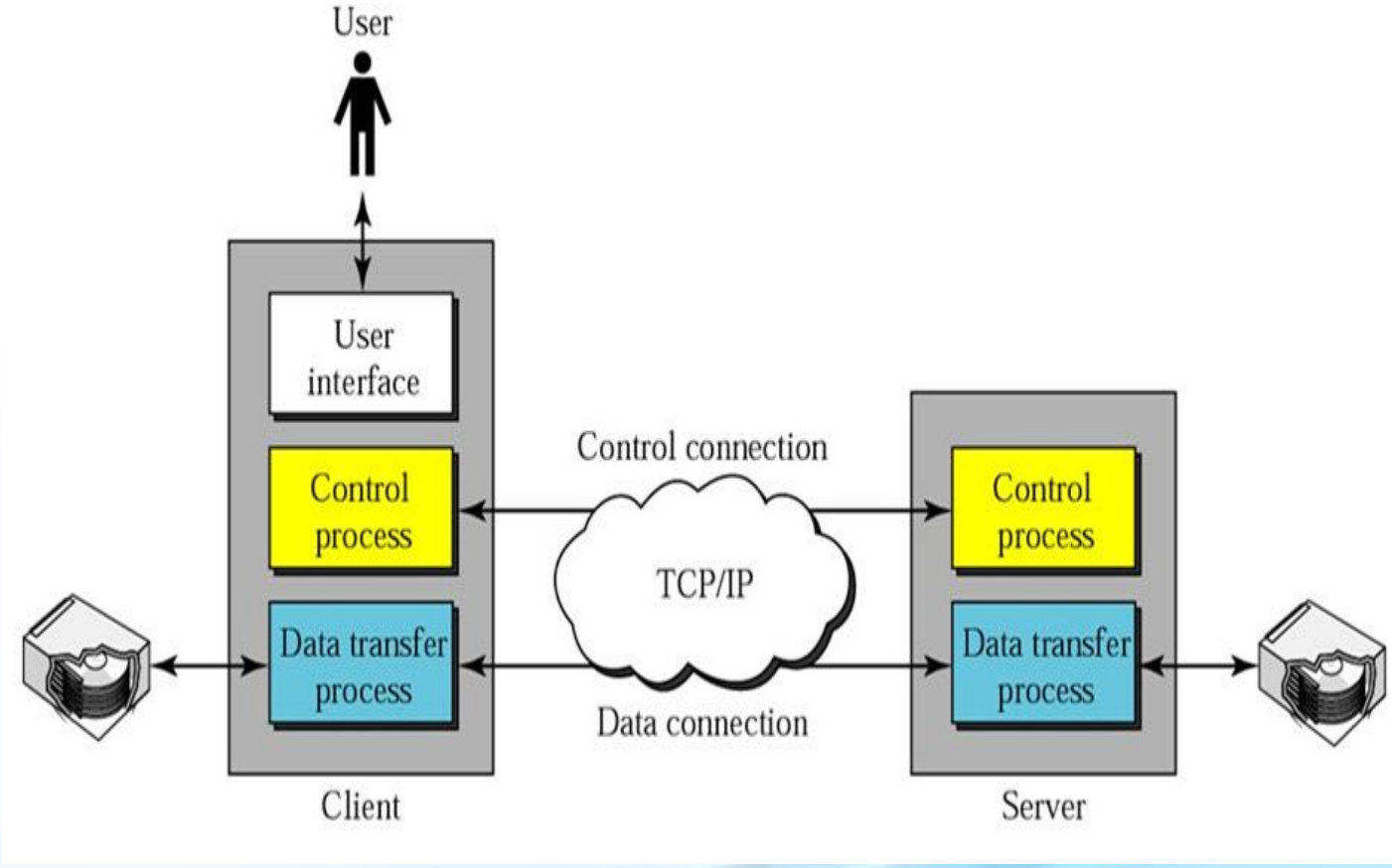
Chart created September 5, 2000 by Suso Banderas(suso@suso.org). Most of the summary information was gathered from Appendix A of "Apache Server Administrator's Handbook" by Mohammed J. Kabir.

HTTP Cookies

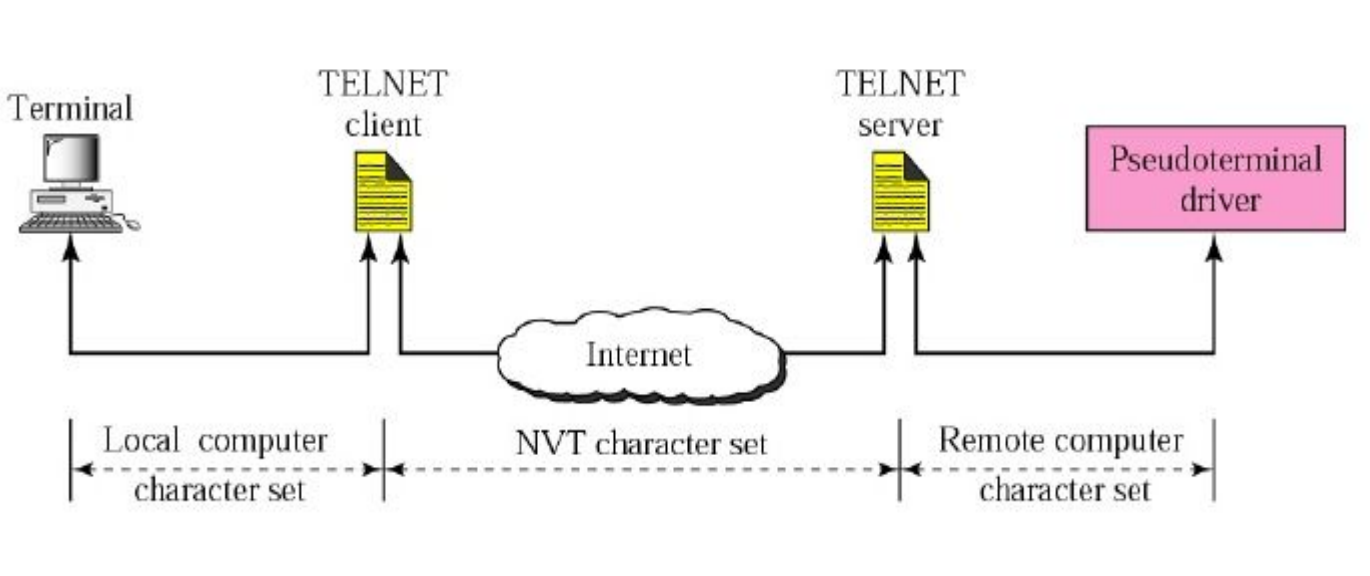
- Cookies are small files which are stored on a user's computer.
- They are designed to hold a modest amount of data specific to a particular client and website
- Cookies are a convenient way to **carry information from one session on a website to another**, or between sessions on related websites, without having to burden a server machine with massive amounts of data storage.
- Cookie can be **set for a period** - The time of expiry of a cookie can be set when the cookie is created. By default the cookie is destroyed when the current browser window is closed.
- When a cookie is created it is possible to control its **visibility** by setting its 'root domain'. It will then be accessible to any URL belonging to that root.

B) File Transfer Protocol (FTP)

- Better option when it comes to transferring large file
- TCP/IP
- Separation of Command and data transfer
- 2 port -> 21 : Control connection
  ->20 : Data Connection
- Uses NVT



- Network Virtual Terminal (NVT)
  - ✔ The Network Virtual Terminal (NVT) is a representation of a basic terminal and provides a standard that the computers on either end of a FTP/Telnet connection are assumed to follow.



  - ✔ allows interoperability between FTP/Telnet and a variety of heterogeneous computers and operating systems.
  - ✔ consists of a virtual keyboard that generates user-specified characters and a printer that displays specific characters.
  - ✔ Clients and servers can map their local devices to the characteristics and handling conventions of an NVT and can assume that other servers and clients are doing the same

- **Control Connection**
  - ✔ **Commands on control channel**

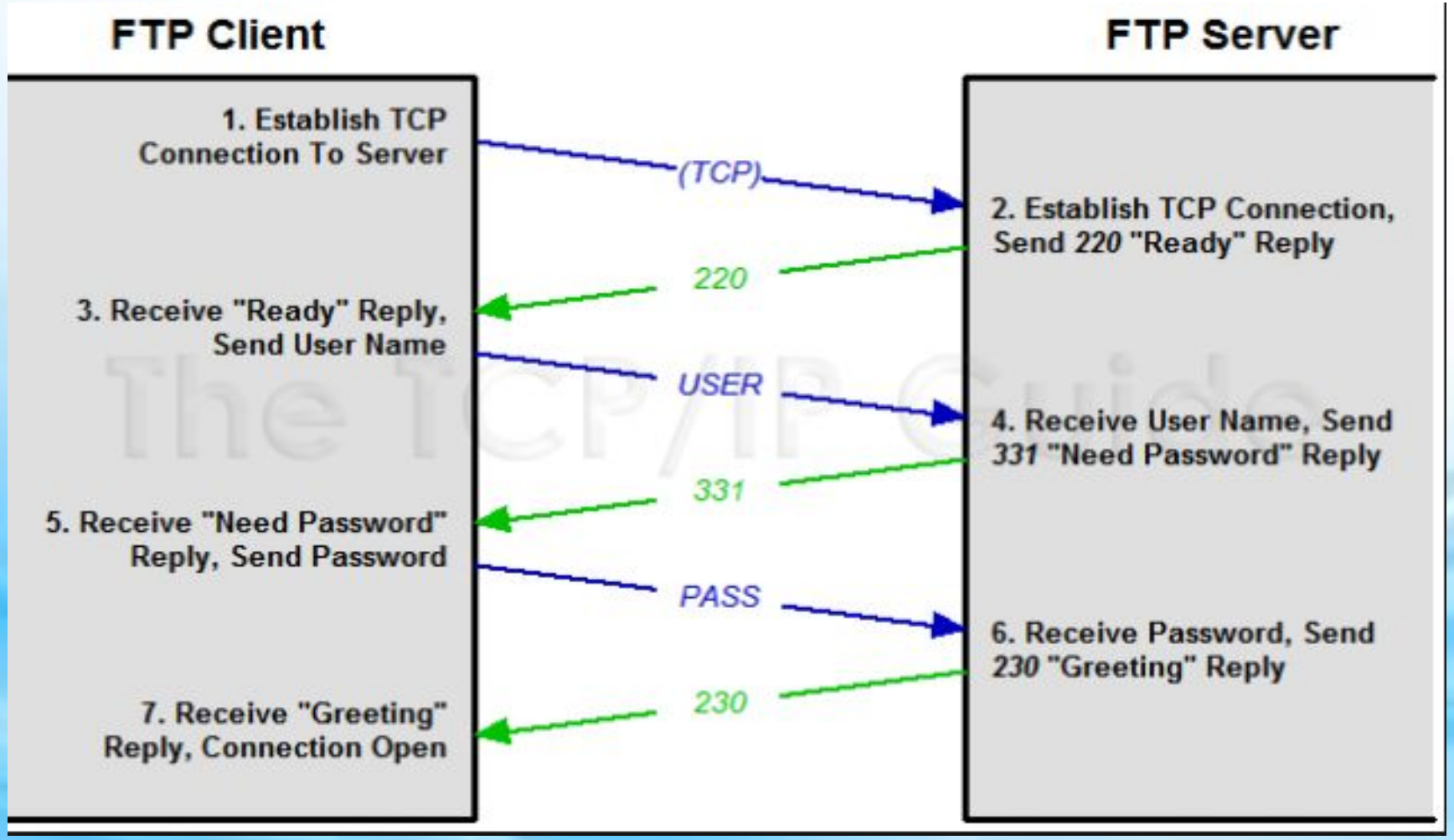
    USER – Send username to the FTP server
    PASS – Send the password (Anonymous servers need email address)
    CWD – Change the working directory on the server
    PASV – To enter the passive mode (To let client connect to the server)
    RETR – To retrieve a remote file from the server
    QUIT – To terminate the connection to the server

  - ✔ Response has two parts: 3 digit code and text

✔ status messages are returned to your display using a **3-digit code**.

Code	What It Means
110	Restart the marker reply
120	Service is ready in nnn minutes
125	Data connection is already open; transfer is starting
150	File starting OK; about to open the data connection
200	Command OK
202	Command was not implemented; it is not used on this system
211	System status, or system help reply
212	Directory status
213	File status
214	Help message
220	Service is ready for a new user
226	Closing the data connection; the requested file action was successful
230	User is logged in

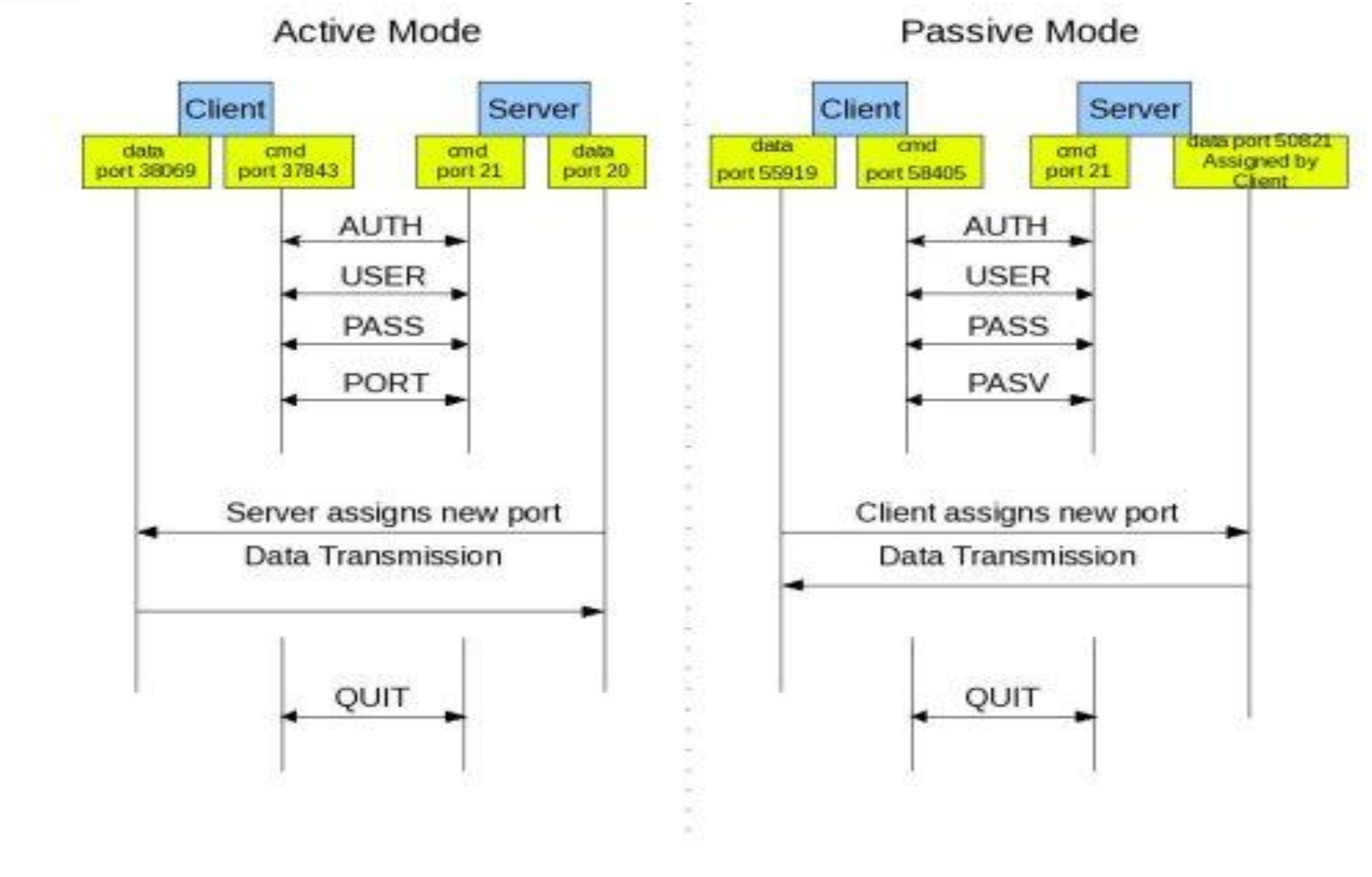
331	Password is required
332	Account is required
425	Cannot open the data connection
426	Connection is closed; the transfer ended abnormally
450	Requested file action was not taken; file busy
451	Requested action ended abnormally; local error in processing
452	Requested action was not taken; insufficient storage exists in system
500	Syntax error; command was unrecognized
501	Syntax error in the parameters or arguments
502	Command was not implemented
503	Bad sequence of commands
504	Command was not implemented for that parameter


FTP Client
FTP Server
1. Establish TCP Connection To Server
(TCP)
2. Establish TCP Connection, Send 220 "Ready" Reply
220
3. Receive "Ready" Reply, Send User Name
USER
4. Receive User Name, Send 331 "Need Password" Reply
331
5. Receive "Need Password" Reply, Send Password
PASS
6. Receive Password, Send 230 "Greeting" Reply
230
7. Receive "Greeting" Reply, Connection Open

- **Data Connection**
  - ✔ Client must define: type of file, structure of data, transmission mode
    - **type of file**: ASCII, EBCDIC, image
    - **structure of data**: file structure, record structure or page structure
    - **transmission mode**: stream mode, block mode, compressed mode
      - **Stream mode**
        - It is the default mode.
        - File is transmitted as continuous stream of bytes to TCP.
        - TCP is responsible for chopping data into segments of appropriate size – Record /Page
      - **Block mode**
        - Data is delivered from FTP to TCP in blocks.
        - Each block is preceded by 3 bytes header.
        - The first byte is called the block descriptor.
        - The second and third byte defines the size of the block in bytes.
      - **Compressed mode**
        - Data is usually compressed if the file to be transmitted is very big.
        - The compression method normally used in Run-length encoding.
        - In a text file, usually spaces (blanks) are removed. In a binary file, null characters are compressed.

- FTP Commands
  - GET (server -> client)
  - PUT (client -> server)
  - MPUT
  - MGET

- Active an Passive mode



C) Electronic Mail (e-mail)

- E-mail (electronic mail) is the **exchange of computer- stored messages** by telecommunication.
- **Client –server** is implemented in a different way, through an intermediate machine(server)
- Architecture
  - An e-mail system consists of 3 subsystems as under:

(i) User agents (UA)

(ii) Message Transfer Agents (MTA)

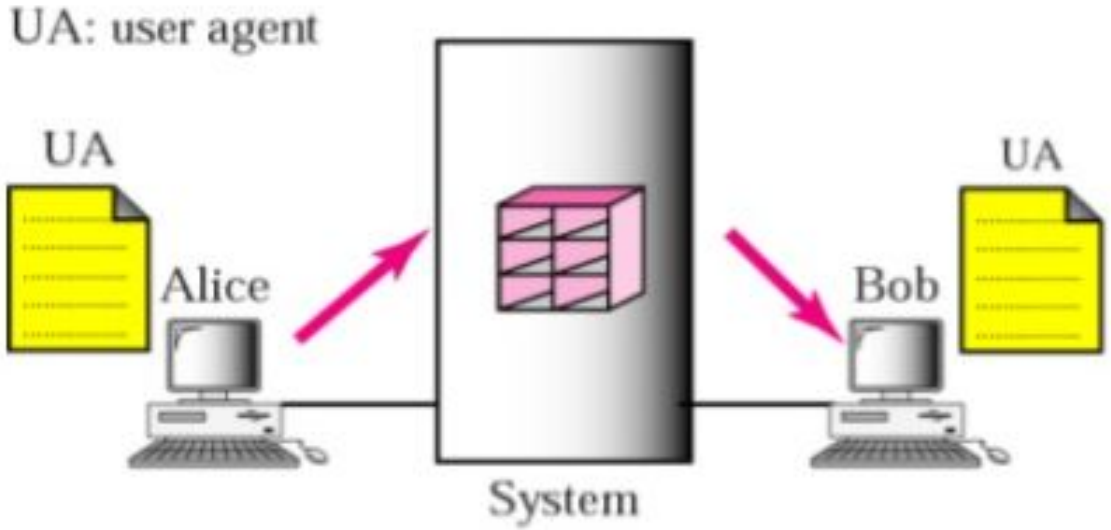
used for transferring (pushing) mails between end computers and mail servers

(iii) MAA (Message Access Agent)

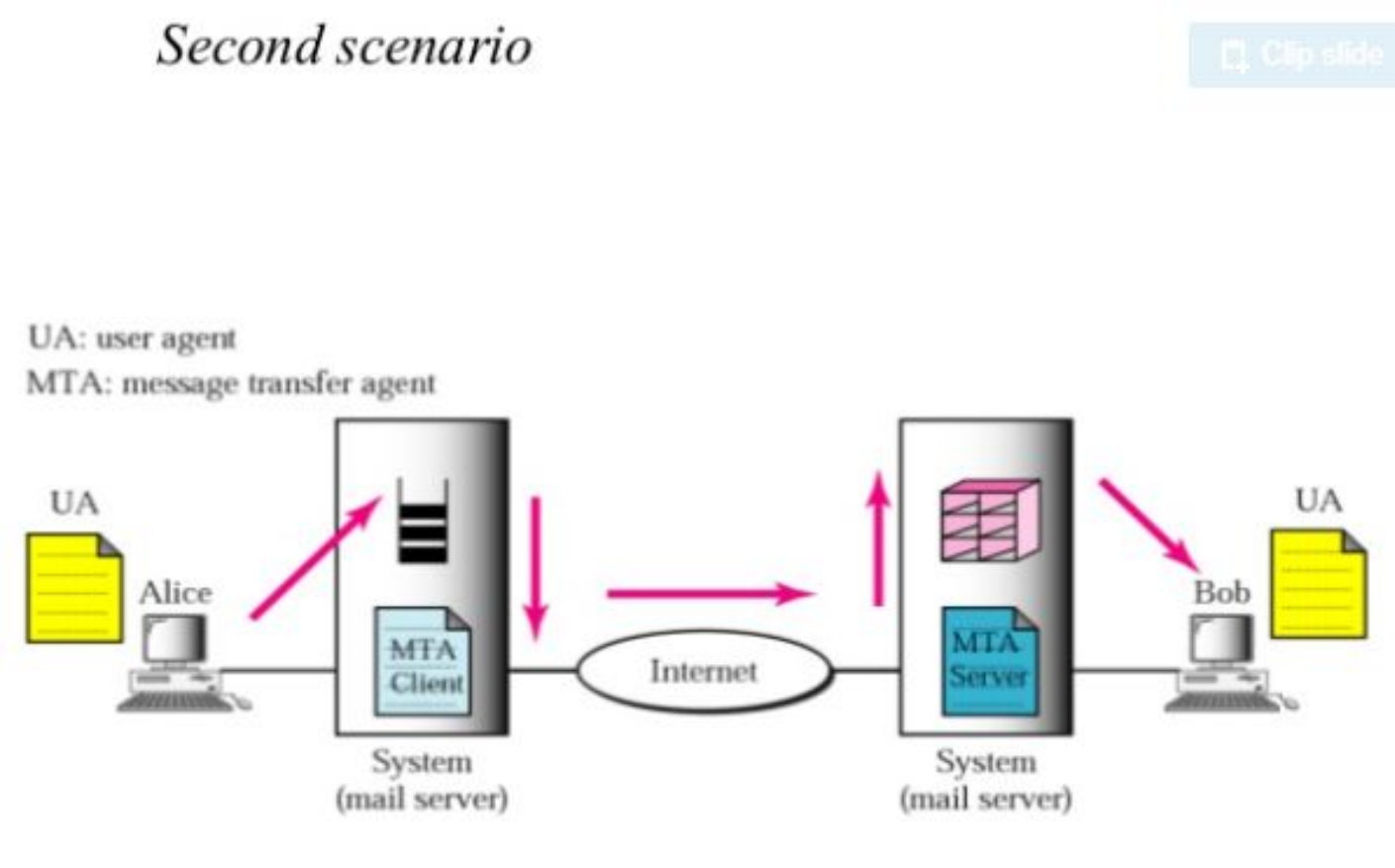
protocols like POP3/IMAP are used for retrieving (pulling) incoming mails from the local mail servers.

- When the sender and the receiver of an email are on the same system, we need only two user agents.

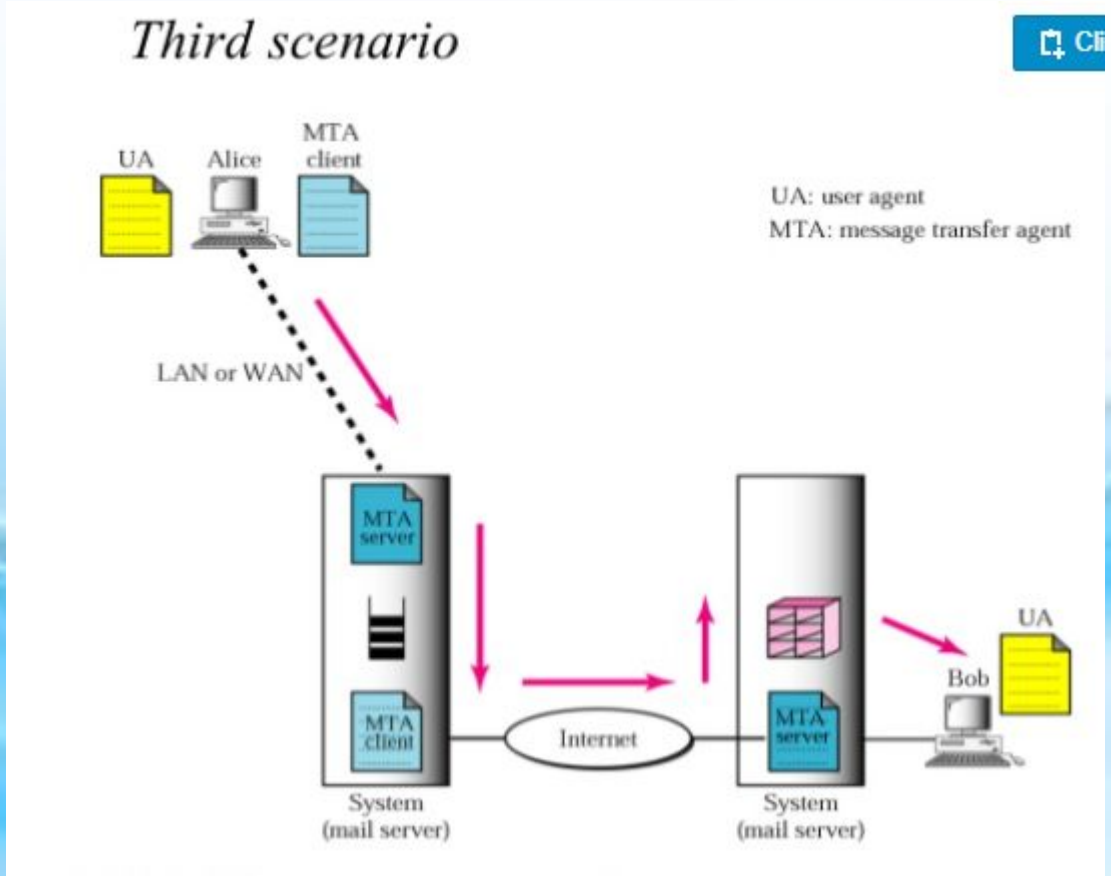
First scenario



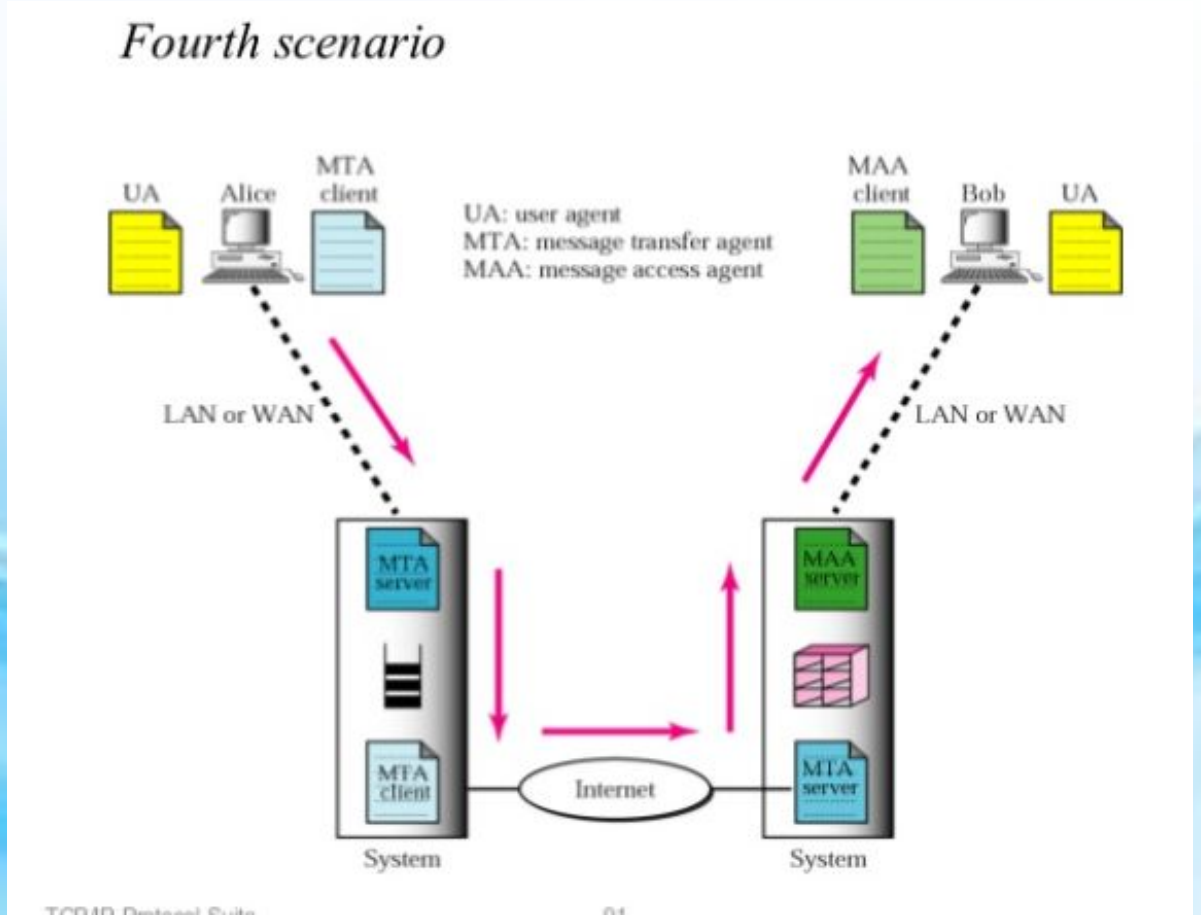
When the sender and the receiver of an email are on different systems, we need two UAs and a pair of MTAs (client and server)

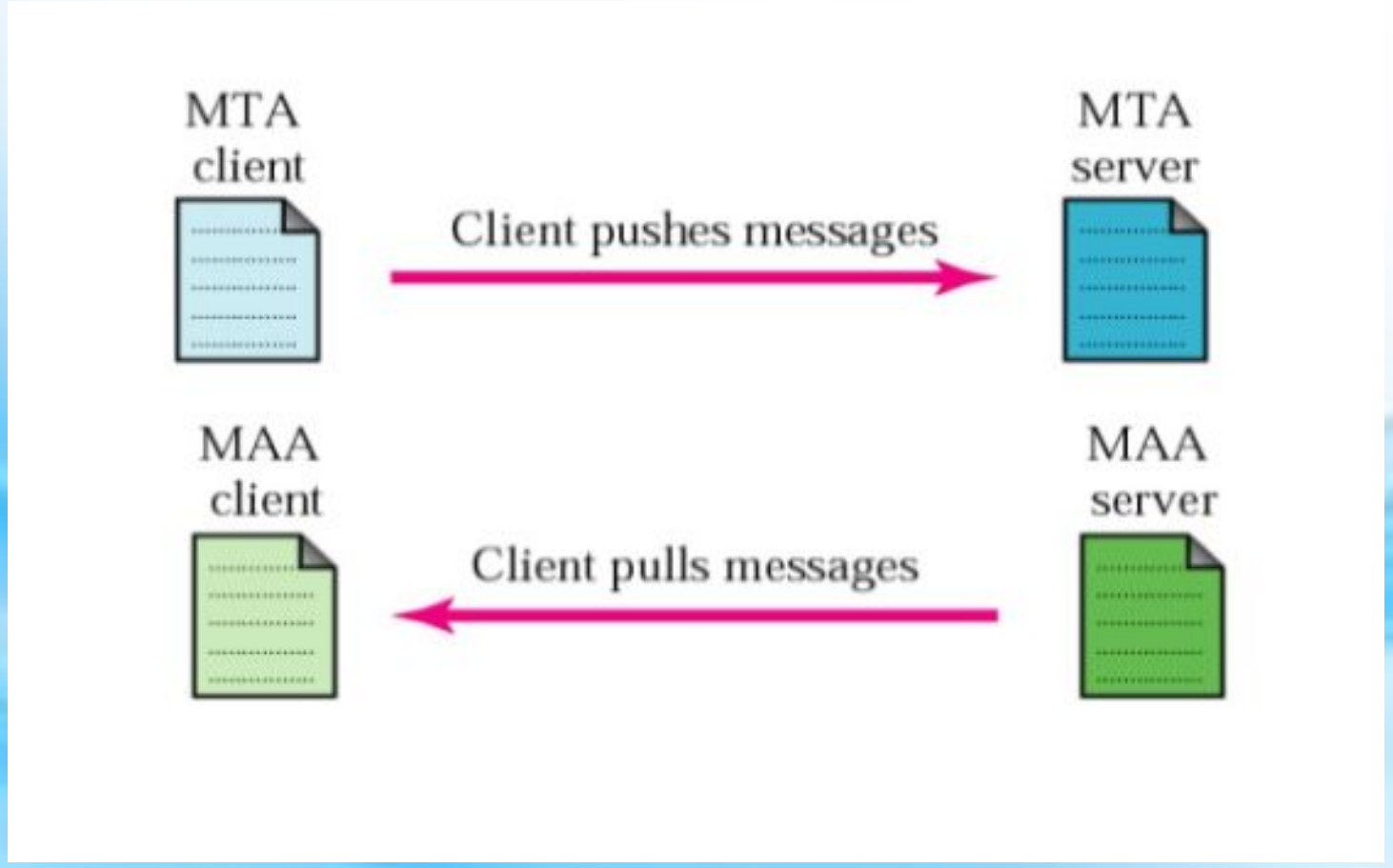


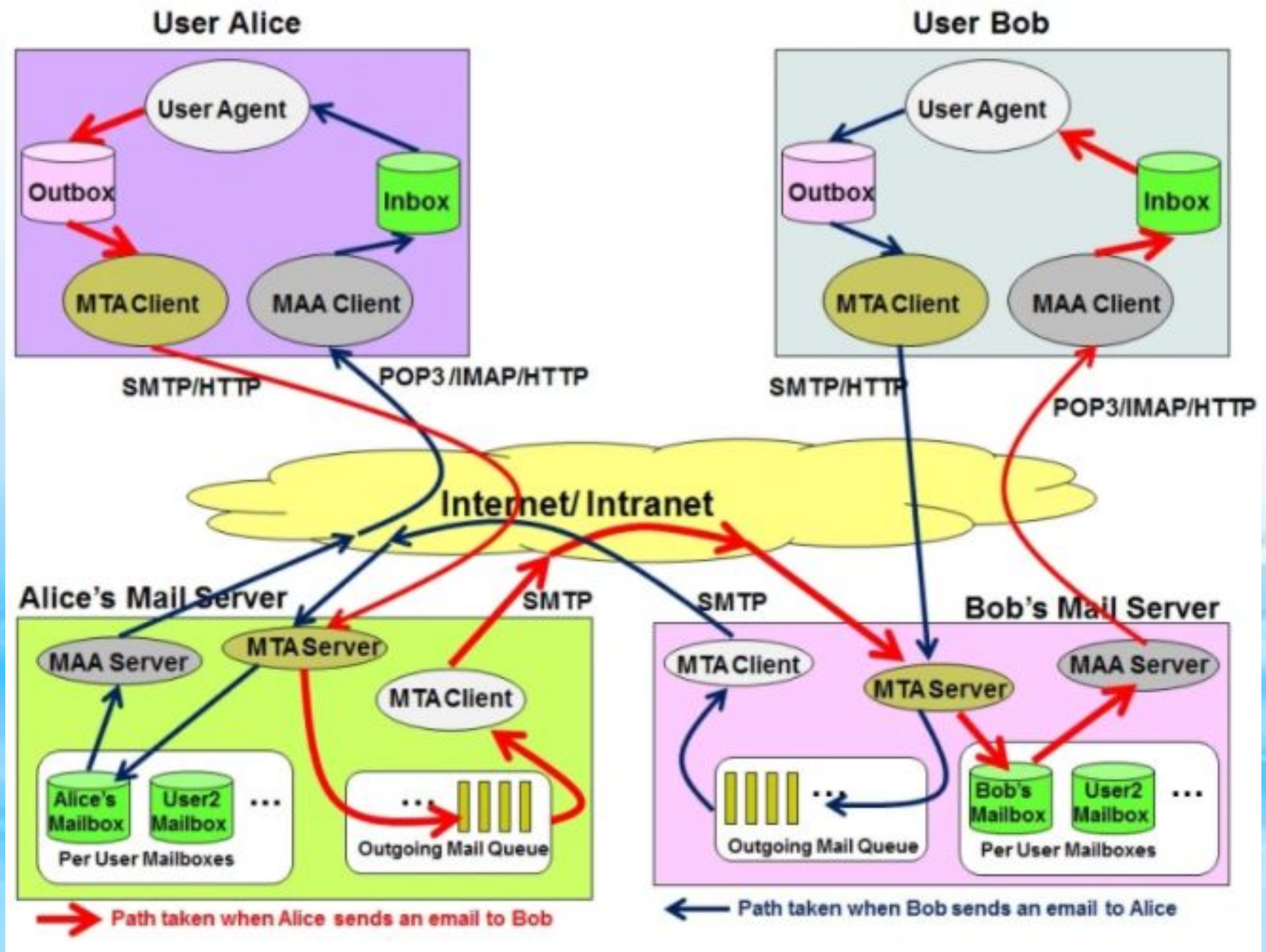
When the sender is connected to the mail server via a LAN or a WAN, we need two UAs and two pairs of MTAs (client and server).



When both sender and receiver are connected to the mail server via a LAN or a WAN, we need two UAs, two pairs of MTAs (client and server), and a pair of MAAs (client and server). This is the most common situation today.




MTA
client
Client pushes messages
MTA
server
MAA
client
Client pulls messages
MAA
server

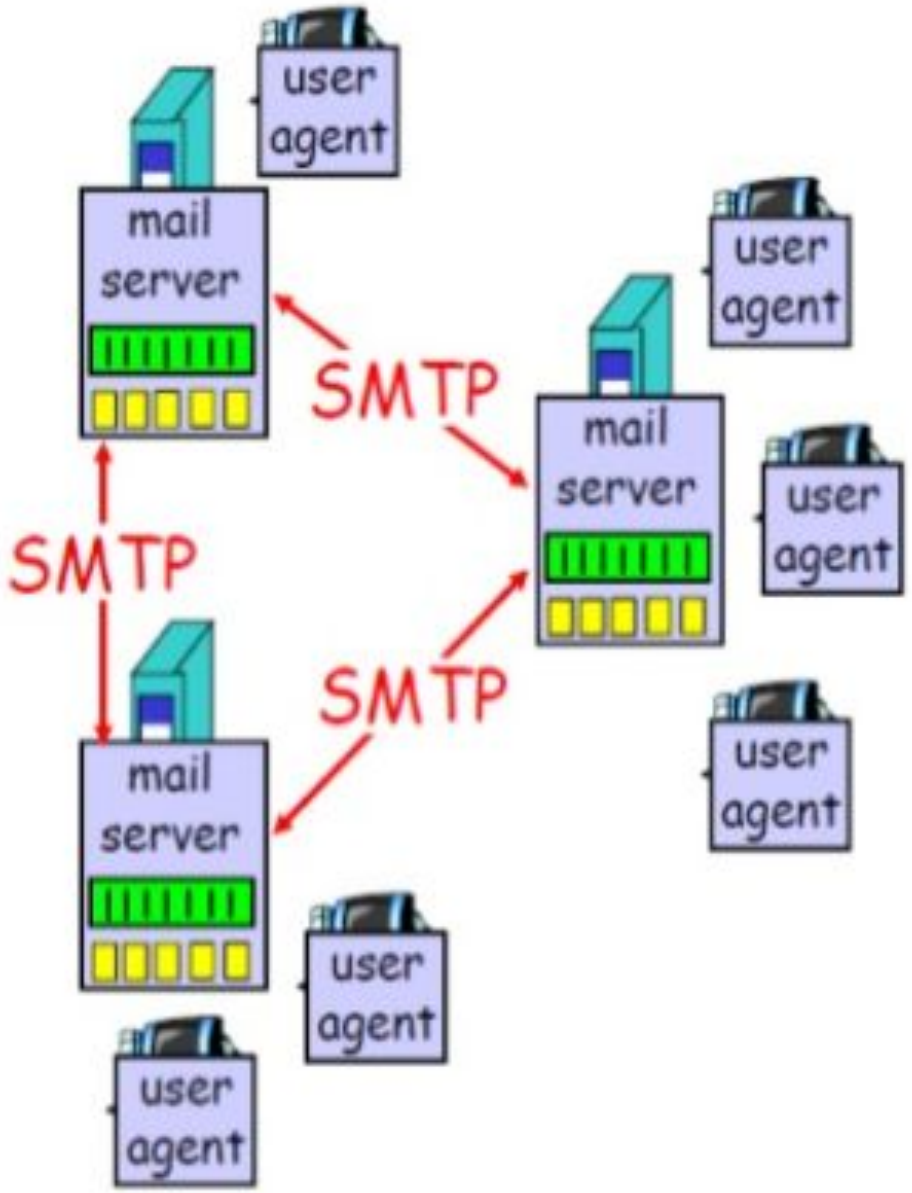

User Alice
User Agent
Outbox
Inbox
MTA Client
MAA Client
SMTP/HTTP
POP3 /IMAP/HTTP
User Bob
User Agent
Outbox
Inbox
MTA Client
MAA Client
SMTP/HTTP
POP3/IMAP/HTTP
Internet/ Intranet
Alice's Mail Server
MAA Server
MTA Server
MTA Client
Alice's Mailbox
User2 Mailbox
Per User Mailboxes
Outgoing Mail Queue
SMTP
SMTP
Bob's Mail Server
MTA Client
MTA Server
MAA Server
Outgoing Mail Queue
Bob's Mailbox
User2 Mailbox
Per User Mailboxes
Path taken when Alice sends an email to Bob
Path taken when Bob sends an email to Alice

User Agents(UA) :They allow the people to read and send e-mail.

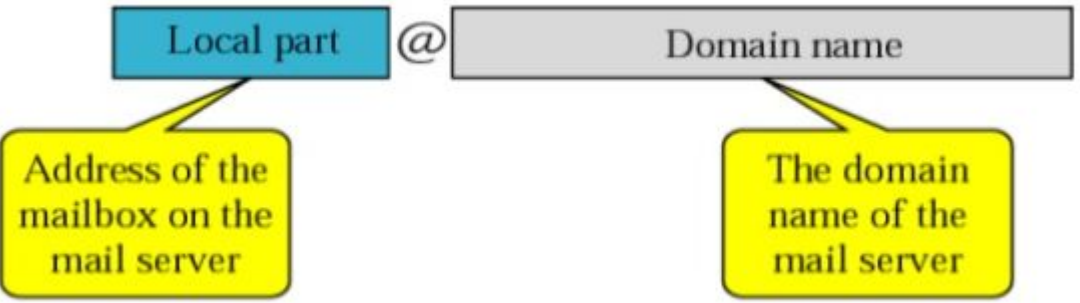
- Listing
- Reading and Replying
- Composition
- Deleting
- Client based (Ms OutLook, Pegasus) or Web based (Gmail, Yahoo)

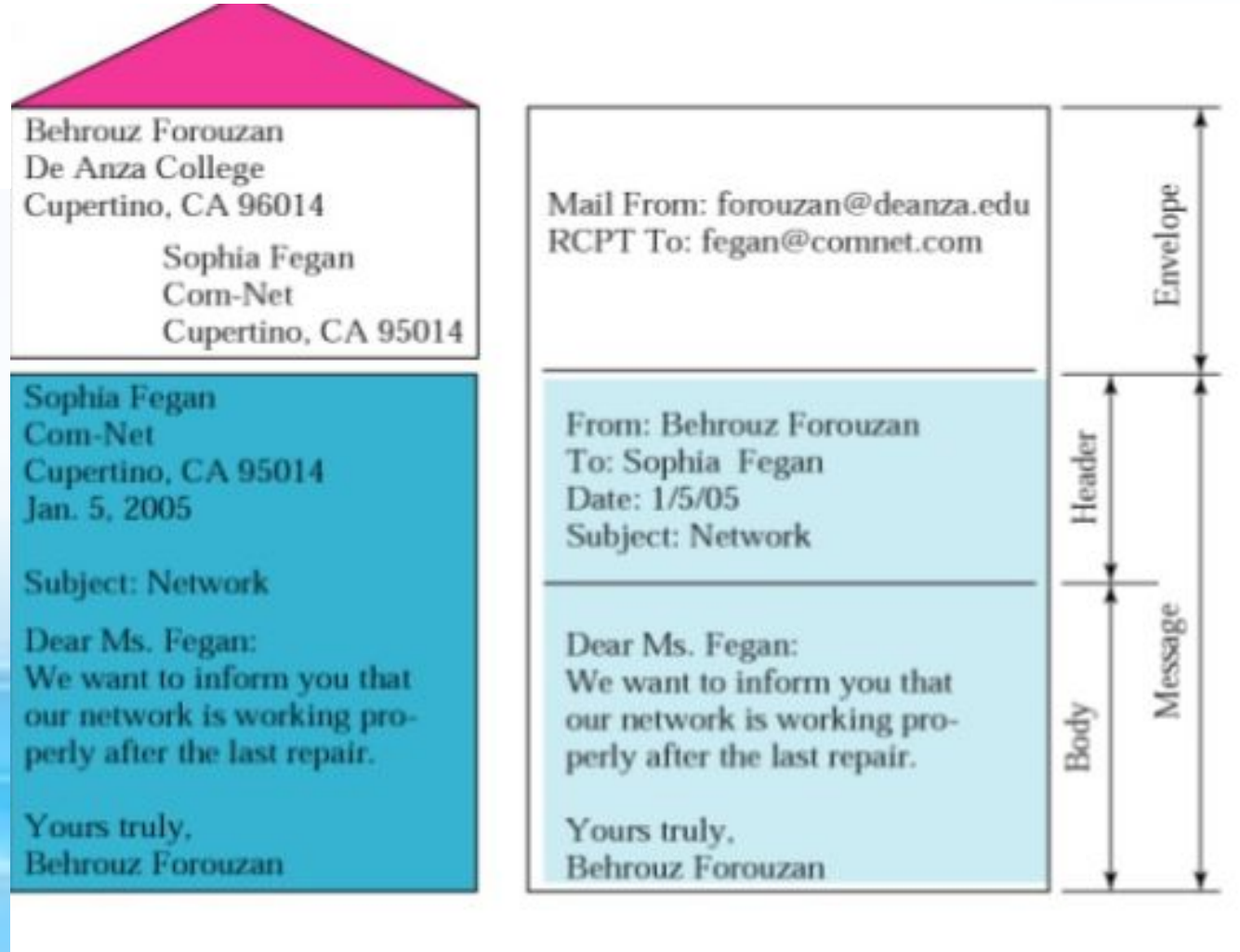
Mail Servers

- mailbox contains incoming messages for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
  - client: sending mail server
  - "server": receiving mail server



mail format


Local part
@
Domain name
Address of the mailbox on the mail server
The domain name of the mail server


Behrouz Forouzan
De Anza College
Cupertino, CA 96014
Sophia Fegan
Com-Net
Cupertino, CA 95014
Sophia Fegan
Com-Net
Cupertino, CA 95014
Jan. 5, 2005
Subject: Network
Dear Ms. Fegan:
We want to inform you that our network is working properly after the last repair.
Yours truly,
Behrouz Forouzan
Mail From: forouzan@deanza.edu
RCPT To: fegan@comnet.com
From: Behrouz Forouzan
To: Sophia Fegan
Date: 1/5/05
Subject: Network
Dear Ms. Fegan:
We want to inform you that our network is working properly after the last repair.
Yours truly,
Behrouz Forouzan
Envelope
Header
Body
Message

Protocols

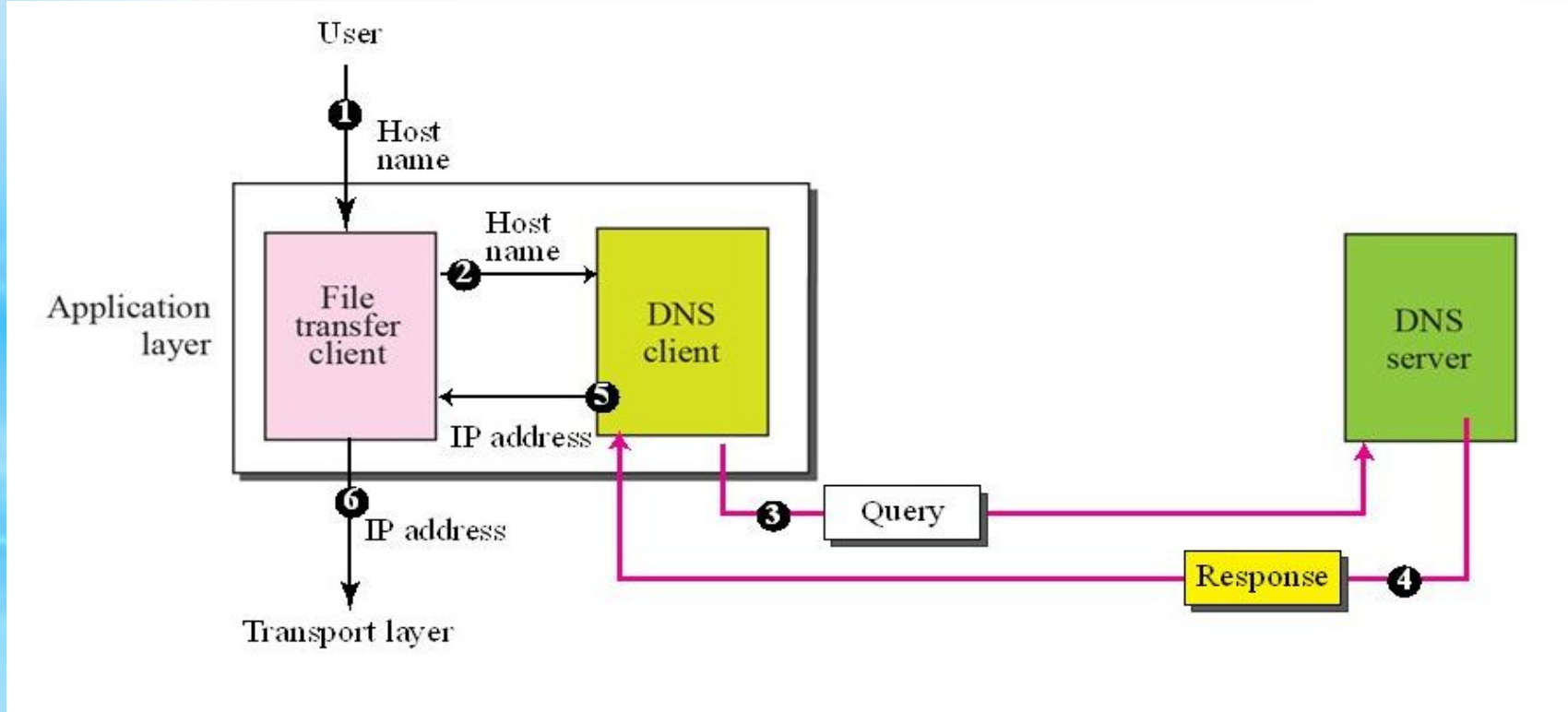
- Client server protocol

- SMTP (Simple Mail Transfer Protocol)
  - protocol for transferring e-mail across the Internet.
  - Port 25
- IMAP (Internet Message Access Protocol)
  - Remote file server
  - email protocol that stores email messages on a mail server, but allows the end user to view and manipulate the messages as though they were stored locally on the end user's computing device(s).
  - support multiple logins, different devices
  - Secure Sockets Layer (SSL) encryption for IMAP.
  - Port 143
  - eg: Gmail

- POP (Post Office Protocol)
  - Store and forward
  - mail is saved for the end user in a single mailbox on the server and moved to the end user's device when the mail client opens.
  - delete mail on the server as soon as the user has downloaded it.
  - Port 110
  - eg: Outlook

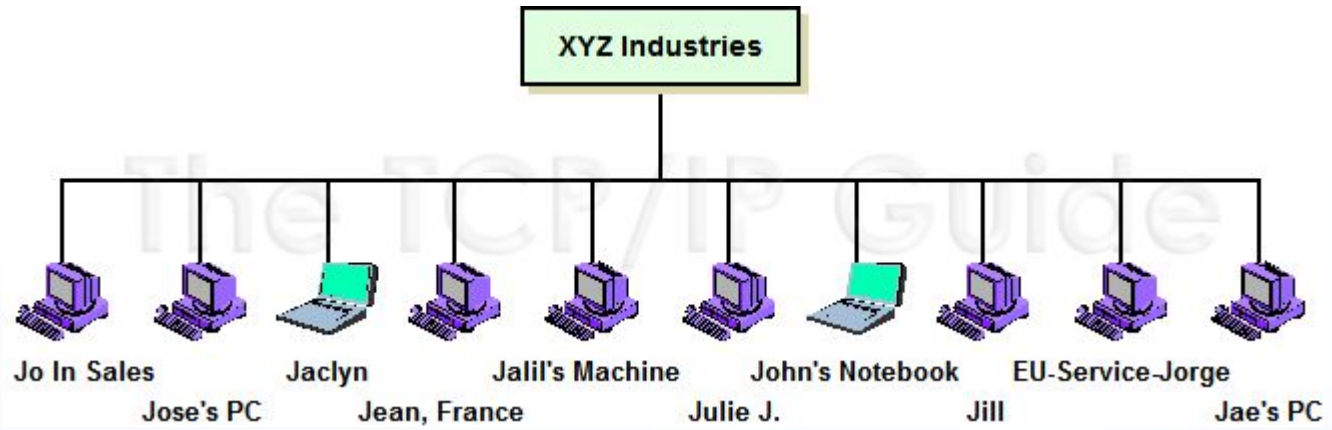
D) Domain Name System (DNS)

- TCP/IP uses IP addresses to identify host
- Humans prefer to use names instead of numeric addresses
- Need for a directory system to name to address
- directory system of DNS is distributed
- Port -53

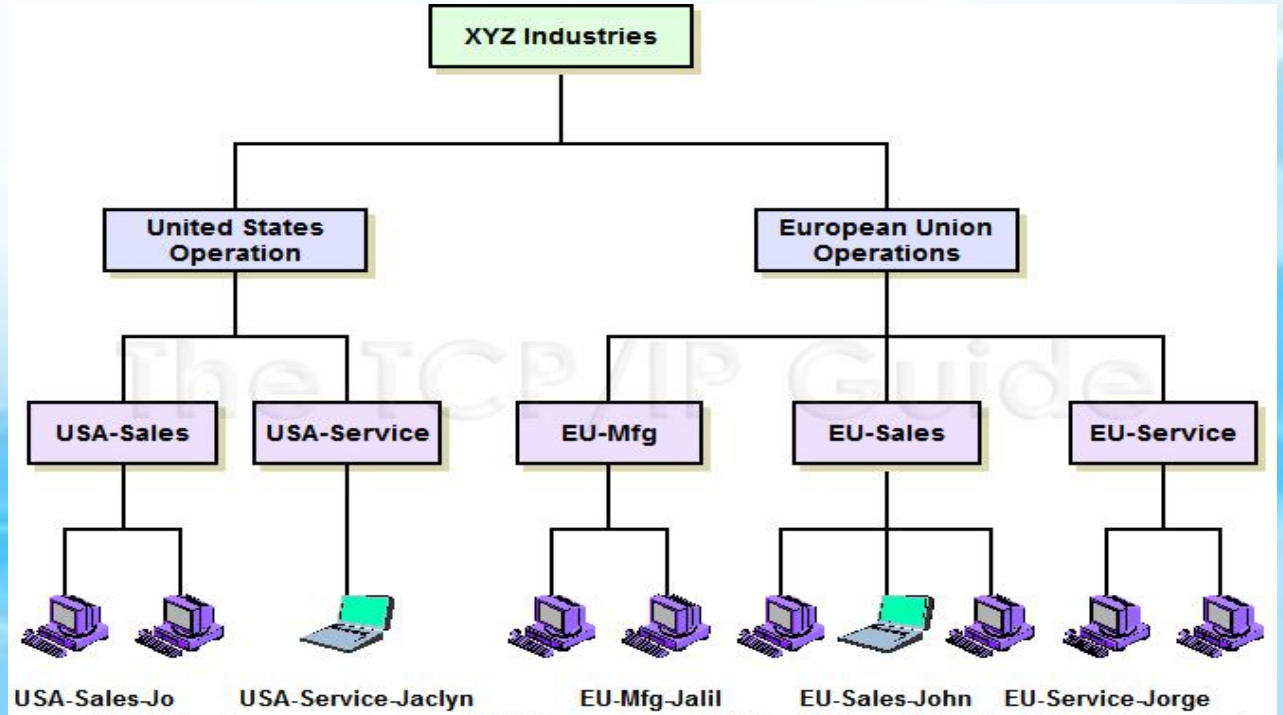


2 connections
: Obtain IP
: service

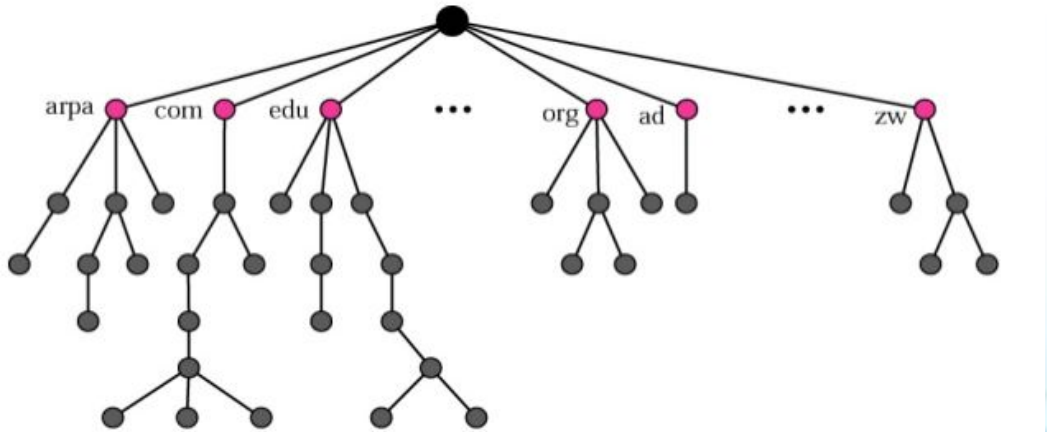
- **Name space**
  - Names must be unique, all registered with ICANN(Internet Corporation for Assigned Name and Numbering)
  - Organization name space :
    - flat



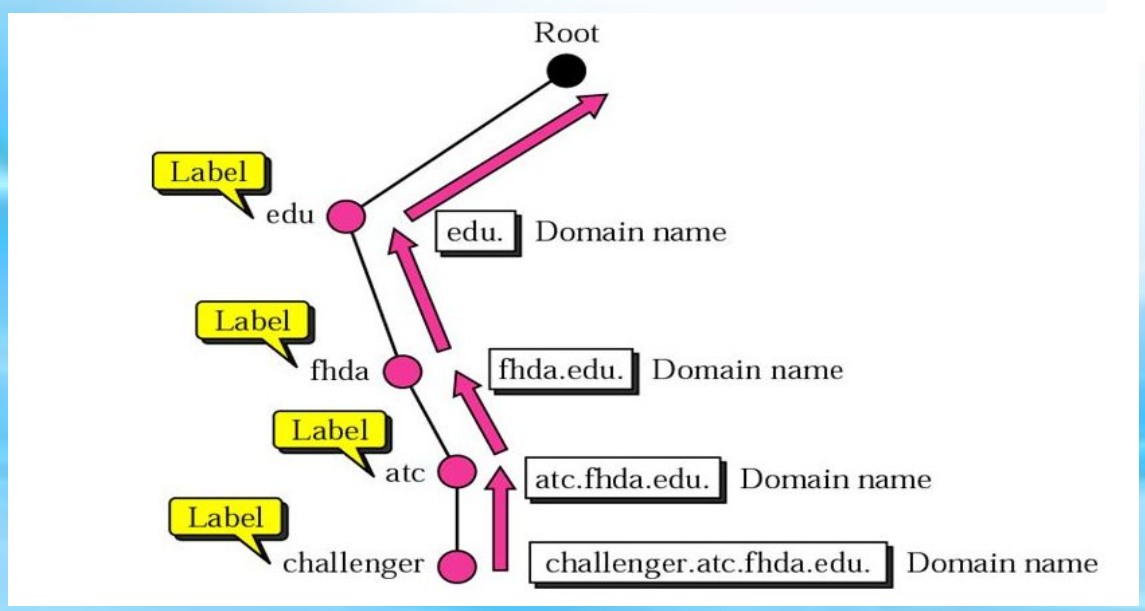
    - Hierarchical

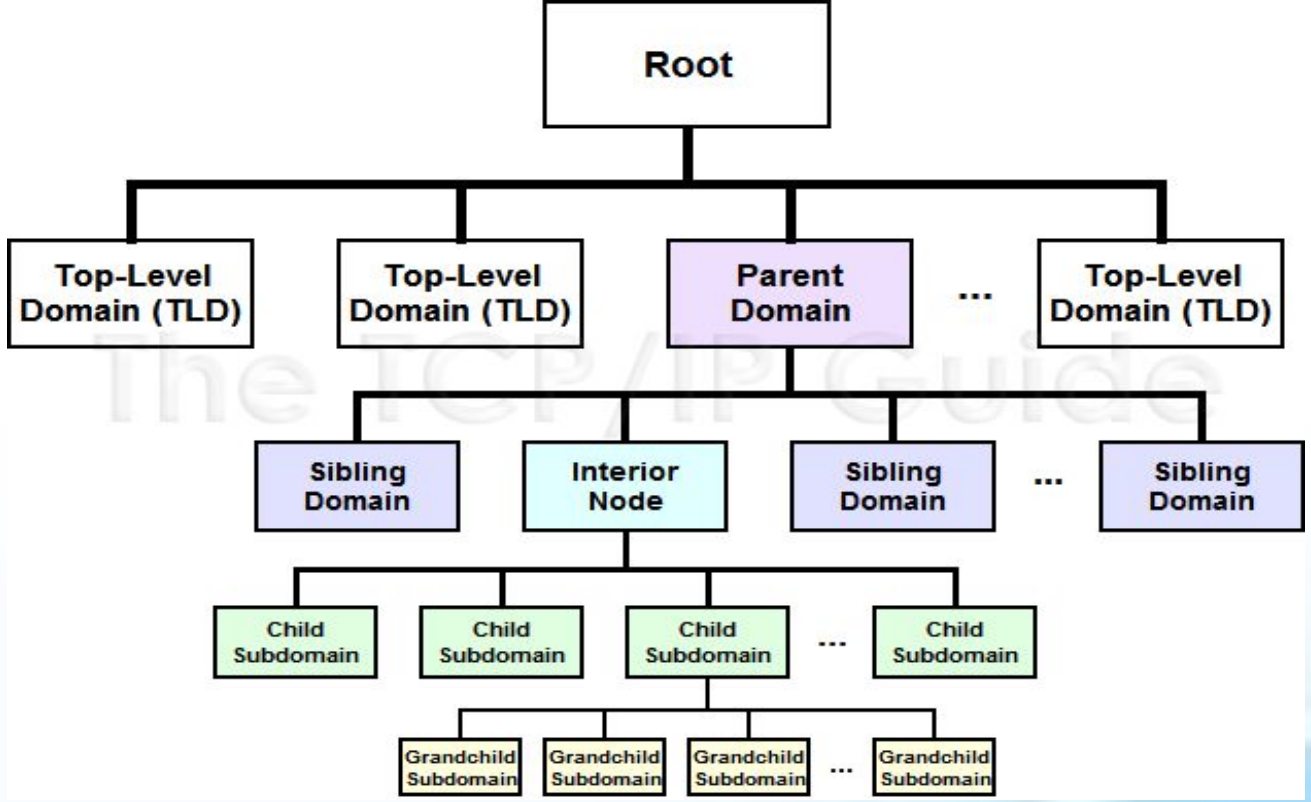


- **Hierarchical Name Space**
  - Root , Top level domain
  - Domain Name Space: 128 levels : 0 (root)-127 levels
    - Label : 63 charater, root : null string

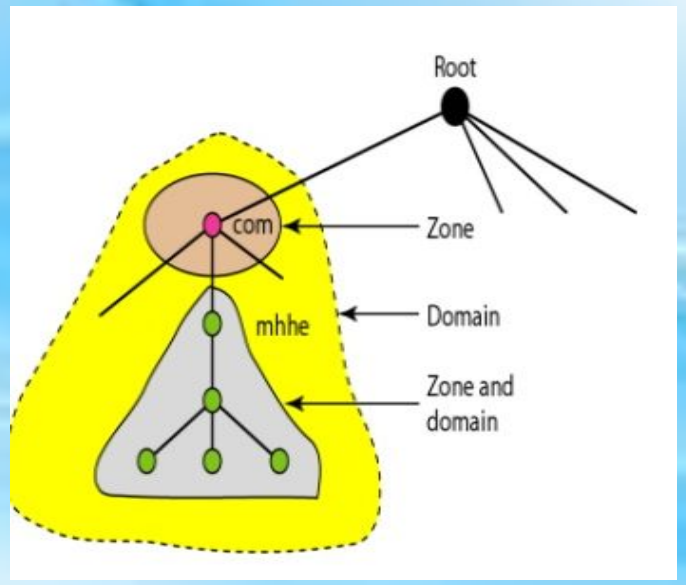


    - Domain Name :
      - Fully qualified domain name (**FQDN)**
      - Partially qualified domain name (P**QDN)**



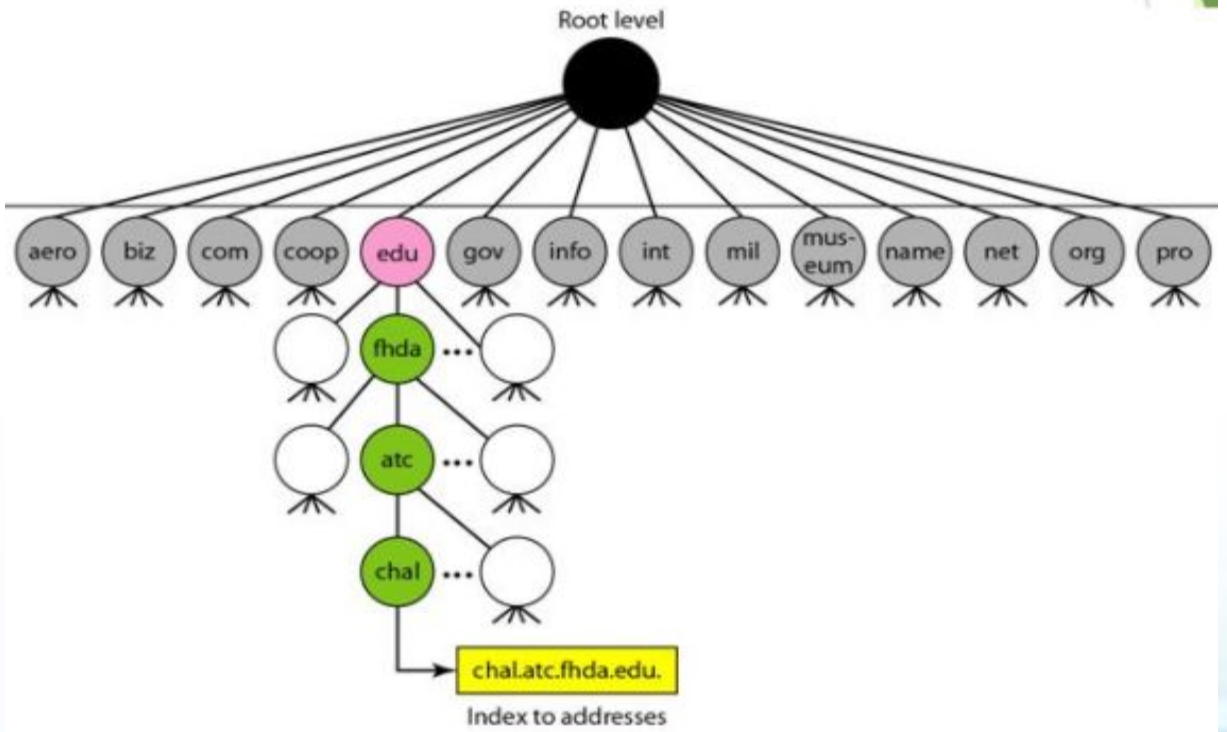


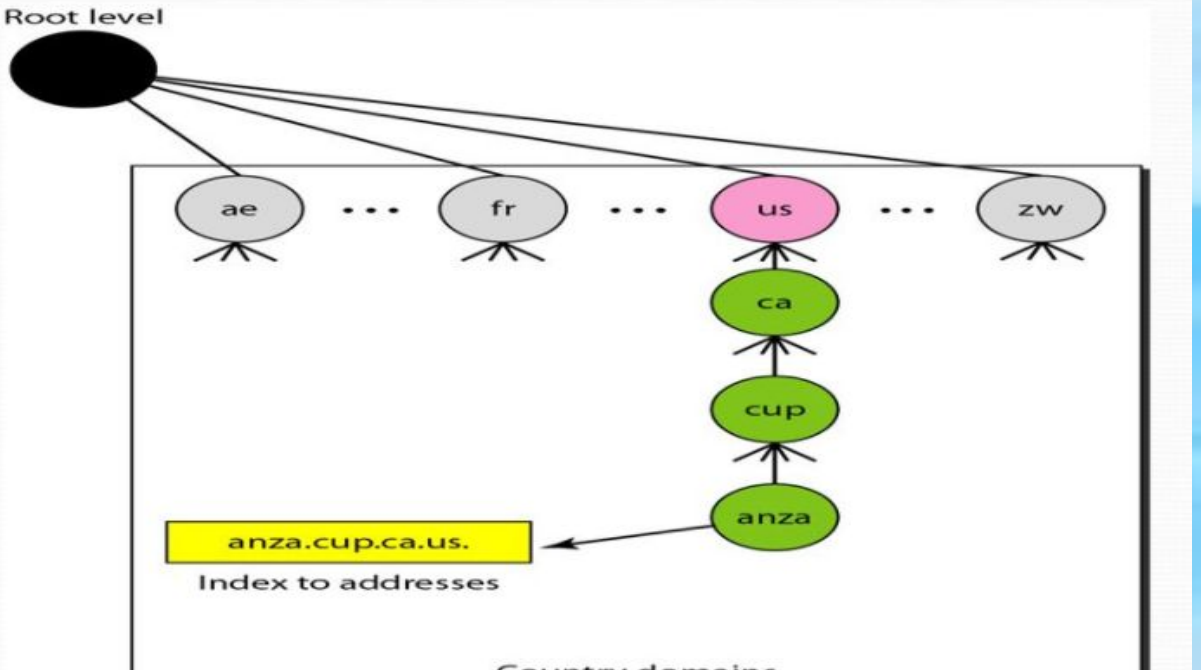
- Domain, sub domains
- Namespaces are distributed
- Hierarchy of Name Space
  - Distribute the information amongst many computers called **DNS server**



- Each server is responsible for it's **zone**
  - Zone file
  - Region over which server has the responsibility and authority
  - Is a part of the entire tree
  - Server can divide domain into smaller domains
- Root Server

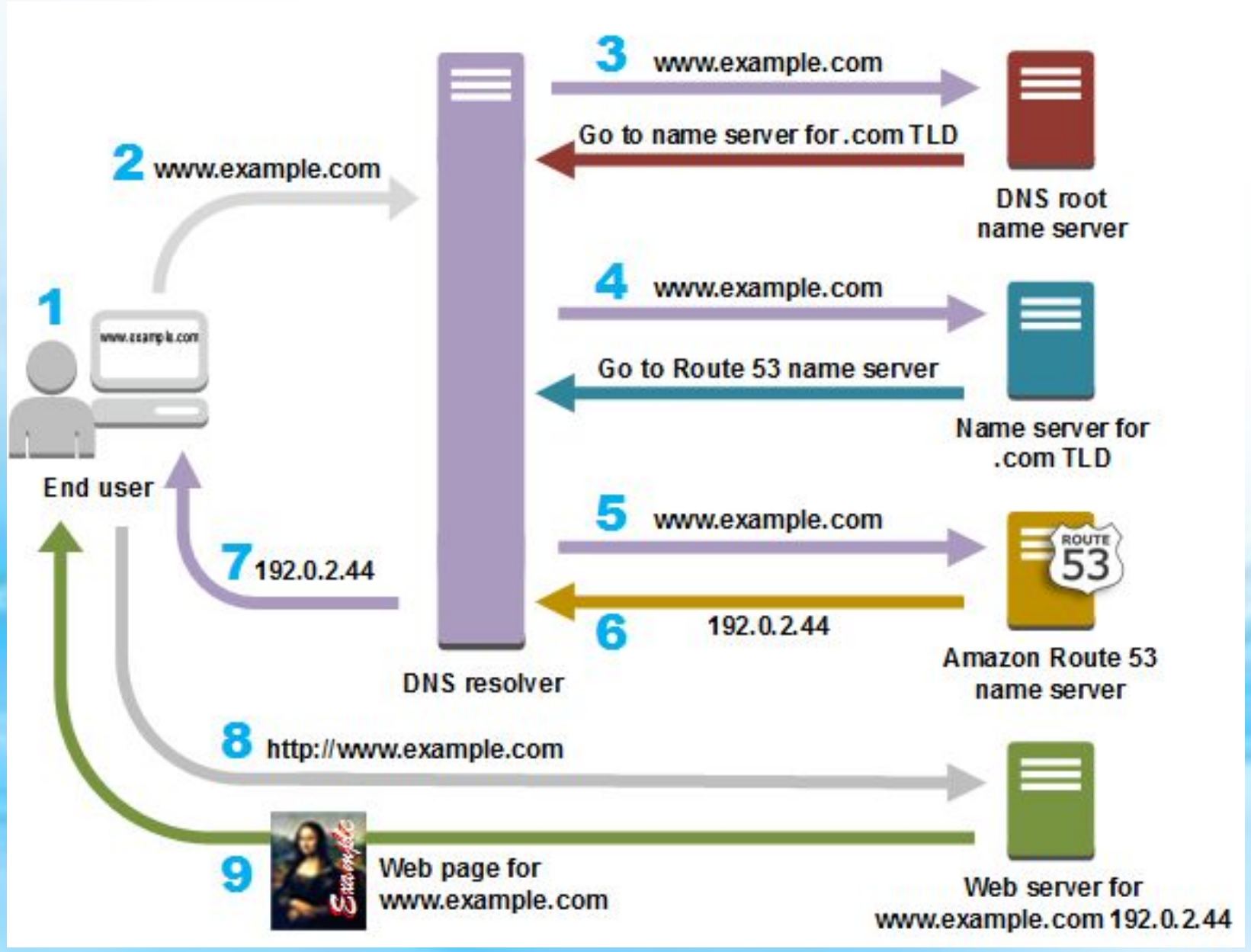
- Primary and Secondary Servers
  - Primary DNS servers contain all relevant resource records and handle DNS queries for a domain. By contrast, secondary DNS servers contain zone file copies that are read-only, meaning they cannot be modified.
  - A server can be primary in a zone and secondary in another zone
- Domain
  - Generic
  - Country





- Resolution : mapping name to address is called name address resolution
  - Resolver : access the closest DNS server with mapping request
  - Techniques
    - Recursive+
    - Iterative
    - Caching : unauthoritative
      : probamatic: outdated mapping

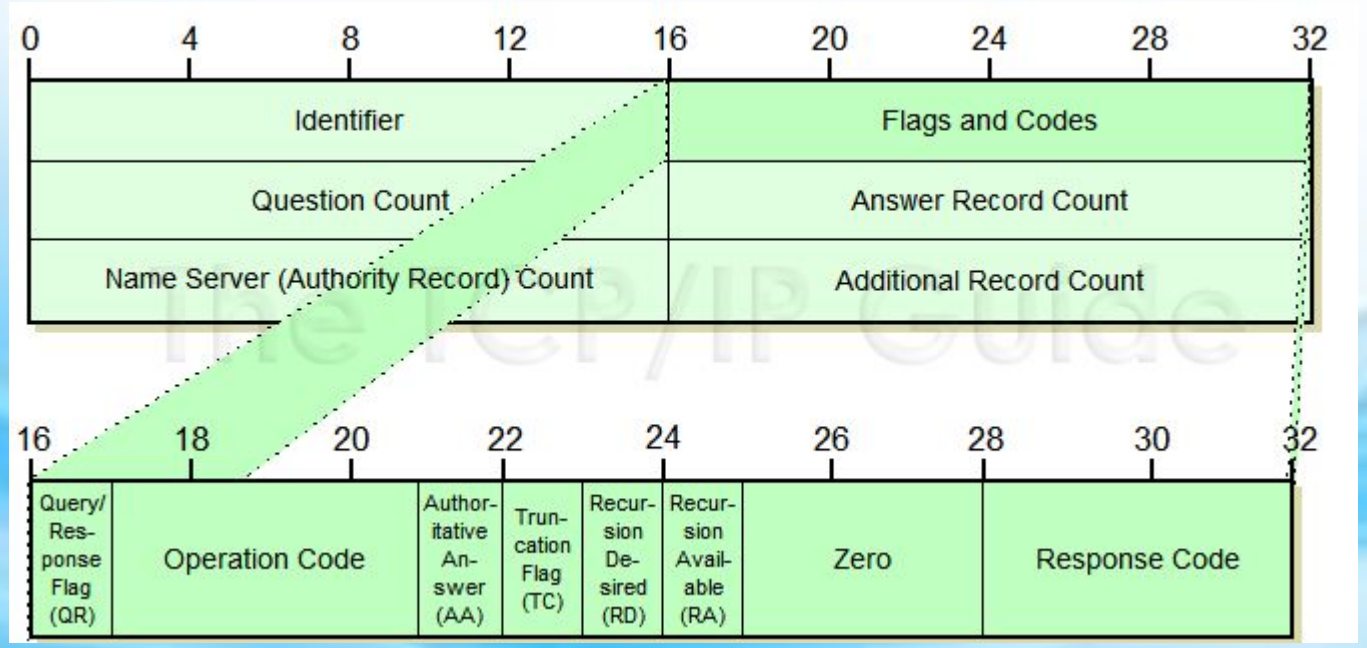
How does DNS resolution take place



- Resource Records
  - 5 tuple structure: (Domain Name, Type, Class, TTL, Value)

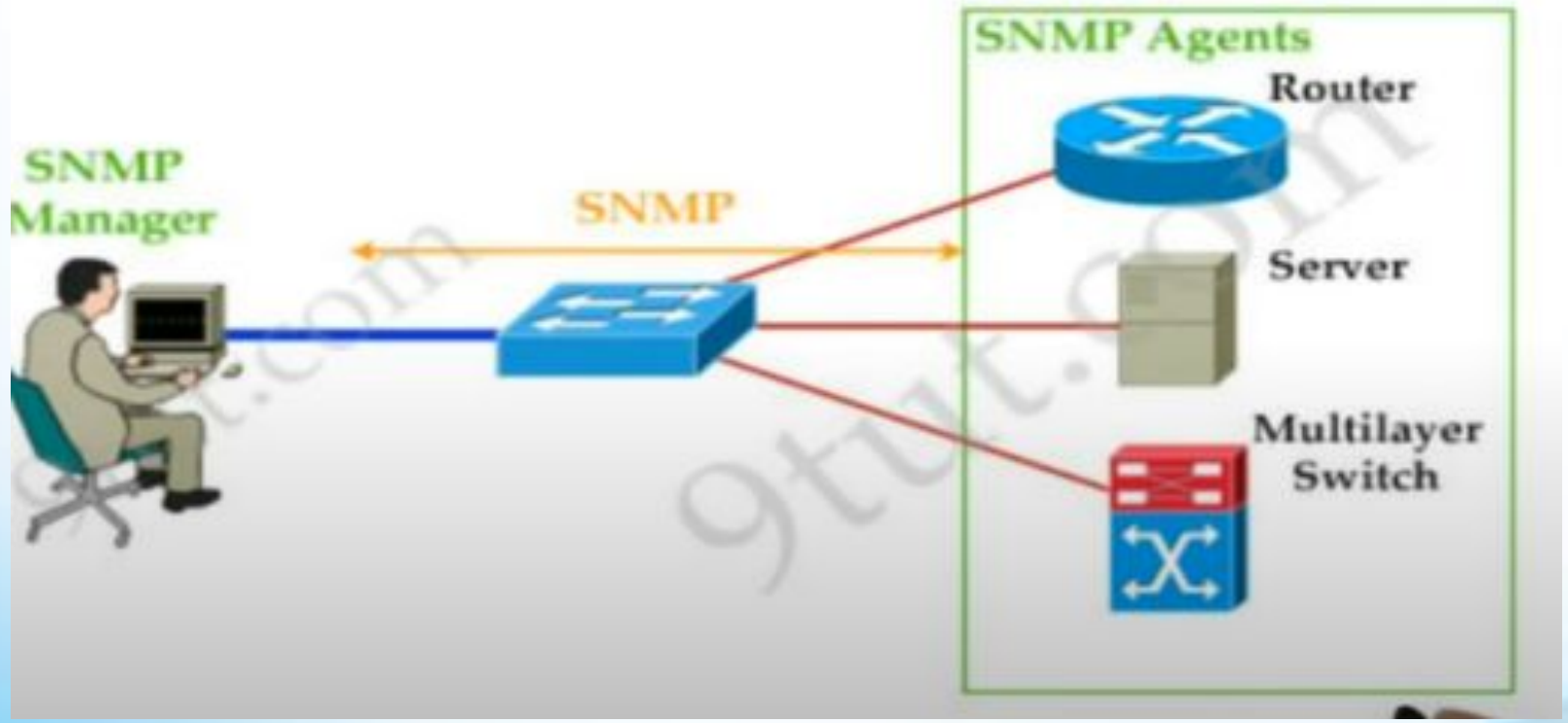
Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

- DNS Message


0
4
8
12
16
20
24
28
32
Identifier
Flags and Codes
Question Count
Answer Record Count
Name Server (Authority Record) Count
Additional Record Count
16
18
20
22
24
26
28
30
32
Query/ Res- ponse Flag (QR)
Operation Code
Author- itative An- swer (AA)
Trun- cation Flag (TC)
Recur- sion De- sired (RD)
Recur- sion Avail- able (RA)
Zero
Response Code

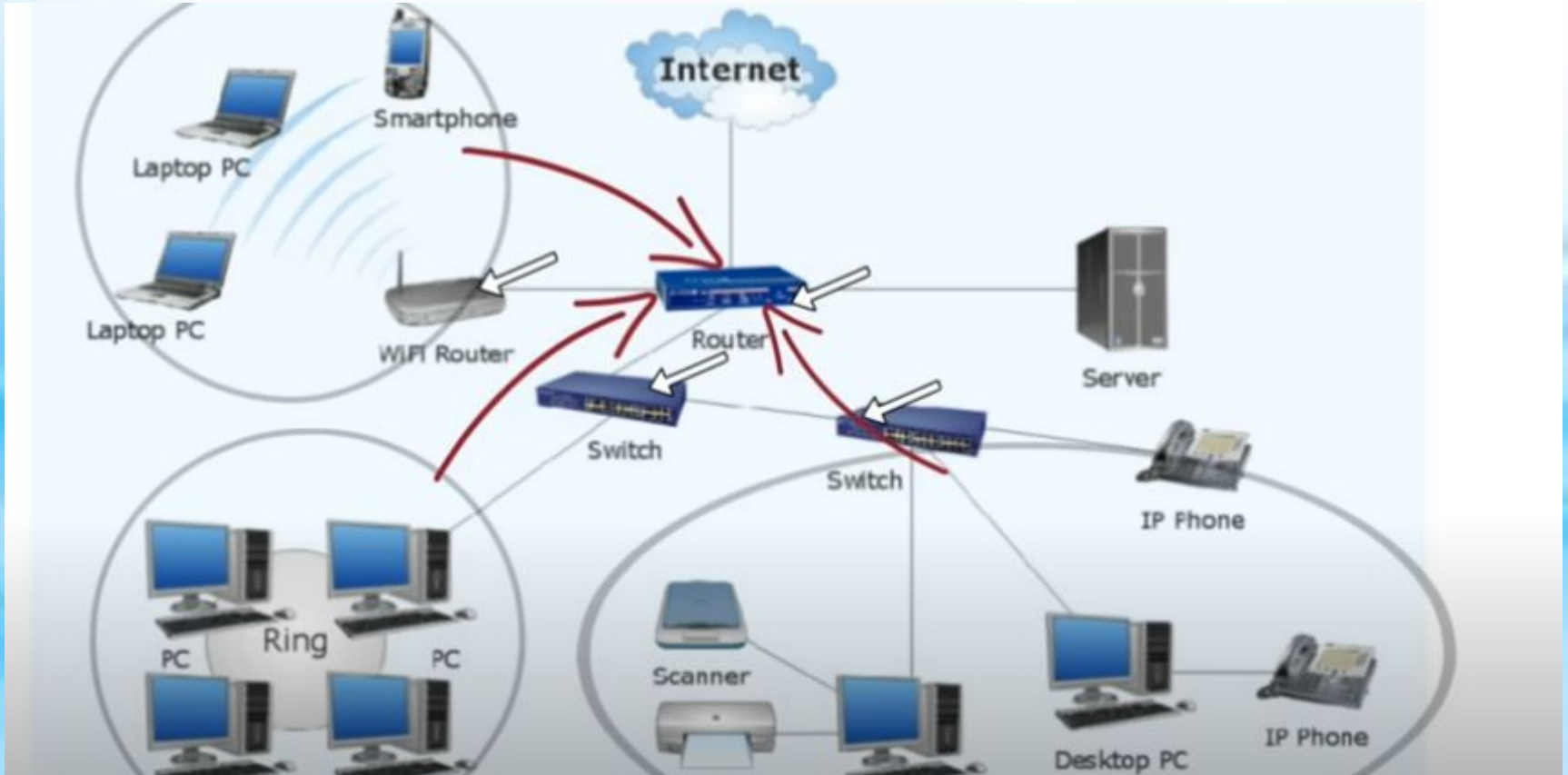
E) SNMP

- The Simple Network Management Protocol (SNMP) is a framework for **managing devices** in an network
- it is a standard protocol for **collecting and organizing info** about managed devices on IP Network - hub, switch, router, bridge etc
- It provides a set of fundamental operations for monitoring and maintaining an internet.
- uses UDP services



- consists of
  - **SNMP Manager**: It is a centralised system used to monitor network.
  - **SNMP Agent:** It is a software management software module installed on a managed device. Managed devices can be network devices like PC, router, switches, servers etc.
  - **Management Information base (MIB):** MIB consists of information of resources that are to be managed. These information is organised hierarchically. It consists of objects instances which are essentially variables.

Router will get congested, with multiple requests, So SNMP is used



SNMP messages –

- GetRequest –SNMP manager sends this message to **request data from SNMP agent**. It is simply used to retrieve data from SNMP agent. In response to this, SNMP agent responds with requested value through response message.
- GetNextRequest –This message can be sent to discover **what all data is available on a SNMP agent**. The SNMP manager can request for data continuously until no more data is left. In this way, SNMP manager can take knowledge of all the available data on SNMP agent.
- GetBulkRequest –This message is used to retrieve l**arge data** at once by the SNMP manager from SNMP agent.

- SetRequest – It is used by SNMP manager to **set the value of an object instance on the SNMP agent.**
  - the value could be updated as compared to Response message.
- Response – It is a message **send from agent upon a request from manager**. When sent in response to Get messages, it will contain the data requested. When sent in response to Set message, it will contain the newly set value as confirmation that the value has been set.
- Trap – These are the message send by the agent without being requested by the manager. It is sent when a **fault** has occurred.

- A MIB or Management Information Base is a formatted text file that resides within the SNMP manager designed to collect information and organize it into a hierarchical format.
- The SNMP manager uses information from the MIB to translate and interpret messages before sending them onwards to the end-user.
- Resources stored within a MIB are referred to as managed objects or management variables.
- The simplest way to think of a MIB is as the central hub of data inside the device.
- The MIB contains all of the performance data that is accessed when loading up a network monitoring tool.

Presentation Layer

Presentation Layer

- The presentation layer is the **sixth** level of the seven layer OSI model.
- It **responds** to service requests from the application layer and **issues service requests** to the session layer.
- Concerned with **syntax and semantics** of the information exchanged between two systems.
- Communication at this layer is **logical** not physical
- **Presentation Format**:
  - In the case of the sender and receiver seeing the same data, the issue is one of agreeing to a message format, called a presentation format .
  - The presentation layer may represent (encode) the data in various ways (e.g., data compression, or encryption), but the receiving peer will convert the encoding back into its original meaning (decode)
- **Providing Services**
  - Data Representation
  - Data Compression
  - Encryption

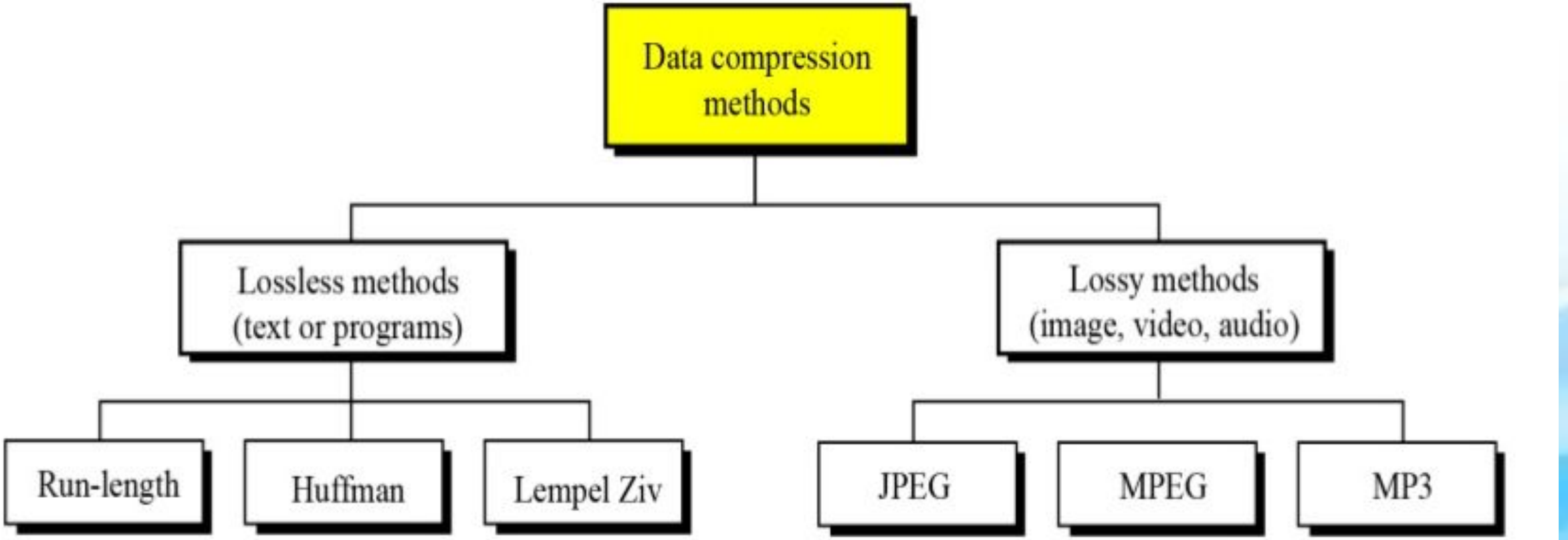
Data Representation

- Different computers have different representations for characters.
- If two dissimilar computers are to exchange text, they must agree on the representation to be used in the exchange. Then one must translate from, or into, the agreed upon representation.
- Converting the complex data structures used by an application (strings, integers, structures, etc.) into a byte stream transmitted across the network.
- Representing information in such a way that communicating peers agree to the format of the data being exchanged. E.g., How many bits does an integer contain?, ASCII or EBCDIC character set?

Data Compression

- Reduces the number of bits contained in the information.
- **Why Compression?**
  - Sometimes programs need to send more data in a timely fashion than the bandwidth of the network supports. (For example, a video stream that needs 10Mbps to transmit on a 1 Mbps network).
  - It's hard to move data on the Internet at >1Mbps.
  - The Internet does not allow applications to use more than their "fair share" of the bandwidth on a congested link.
  - In terms of storage, the capacity of a storage device can be effectively increased with methods that compresses a body of data on its way to a storage device and decompresses it when it is retrieved.
- Need to compress the data at the sender and decompress it at the receiver.

- Types
  - **Lossless Compression** -- data is compressed and can be uncompressed without loss of information. These are referred to as bit-preserving or reversible compression systems.
  - **Lossy Compression** – aim to obtain the best possible fidelity for a given bit-rate or minimizing the bit-rate to achieve a given fidelity measure. Most suited to video and audio compression techniques


Data compression methods
Lossless methods (text or programs)
Lossy methods (image, video, audio)
Run-length
Huffman
Lempel Ziv
JPEG
MPEG
MP3

Comparison between Lossless and Lossy Compression

	Lossy Compression	Lossless Compression
1.	Lossy compression is the method which eliminate the data which is not noticeable.	While Lossless Compression does not eliminate the data which is not noticeable.
2.	In Lossy compression, A file does not restore or rebuilt in its original form.	While in Lossless Compression, A file can be restored in its original form.
3.	In Lossy compression, Data's quality is compromised	But Lossless Compression does not compromise the data's quality

	Lossy Compression	Lossless Compression
4.	Lossy compression reduces the size of data.	But Lossless Compression does not reduce the size of data.
5.	Algorithms used in **Lossy** compression are: Transform coding, Discrete Cosine Transform, Discrete Wavelet Transform, fractal compression etc.	Algorithms used in **Lossless** compression are: Run Length Encoding, Lempel-Ziv-Welch, Huffman Coding, Arithmetic encoding etc.
6.	Lossy compression is used in Images, audio, video.	Lossless Compression is used in Text, images, sound.

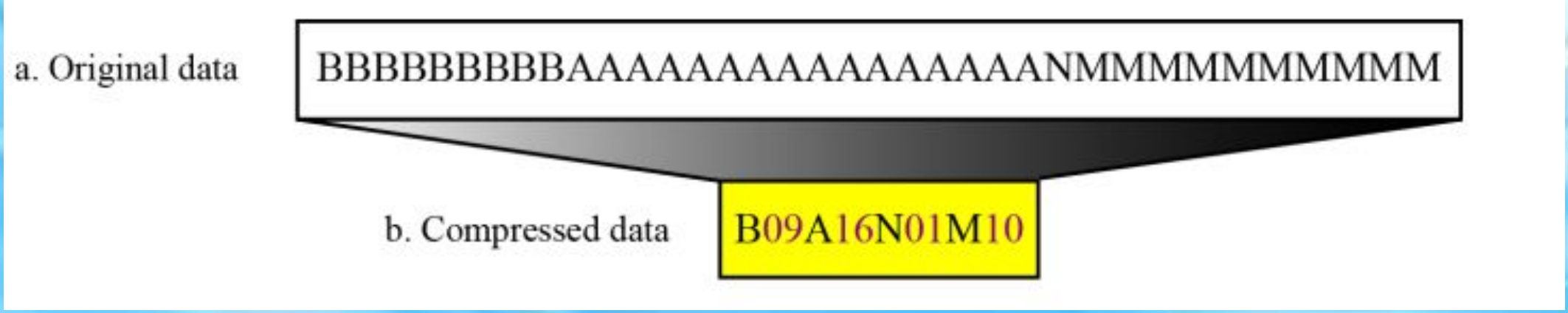
	Lossy Compression	Lossless Compression
7.	Lossy compression has more data-holding capacity.	Lossless Compression has less data-holding capacity than Lossy compression technique.
8.	Lossy compression is also termed as irreversible compression.	Lossless Compression is also termed as reversible compression.

Lossless Coding

- In lossless methods, original data and the data after compression and decompression are exactly the same.
- Redundant data is removed in compression and added during decompression.
- Lossless methods are used when we can’t afford to lose any data: legal and medical documents, computer programs

A) Run-length Coding

- Simplest method of compression.
- How: replace consecutive repeating occurrences of a symbol by 1 occurrence of the symbol itself, then followed by the number of occurrences



B) Dictionary Coding

- Invented by Lempel-Ziv-Welch, referred to as LZW
- Create a dictionary(a table) of strings
- Dictionary is dynamically created
- It is created by the sender and receiver, not sent from sender to the receiver
- 2 types
  - Static dictionary
  - Adaptive dictionary :Lempel-Ziv (LZ77, LZ78), LZW

Static Dictionary Example

- All of the adaptive methods are *one-pass* methods; only one scan of the message is required.
- We have prior knowledge of the message

Colleagues		
name	street	city
peter	narrowstreet	new york
steve	macstreet	cuppertino
mike	longstreet	saarbruecken
tim	unistreet	saarbruecken
hans	msstreet	new york
jens	meerweinstreet	cuppertino
frank	narrowstreet	new york
olaf	macstreet	saarbruecken
stefan	unistreet	saarbruecken
alekh	unistreet	saarbruecken
felix	macstreet	new york
jorge	narrowstreet	saarbruecken

Colleagues2		
name	street	cityID
peter	narrowstreet	0
steve	macstreet	1
mike	longstreet	2
tim	unistreet	2
hans	msstreet	0
jens	meerweinstreet	1
frank	narrowstreet	0
olaf	macstreet	2
stefan	unistreet	2
alekh	unistreet	2
felix	macstreet	0
jorge	narrowstreet	2

Cities_Dictionary	
cityID	city
0	new york
1	cuppertino
2	saarbruecken

Colleagues3		
name	streetID	cityID
peter	0	0
steve	1	1
mike	2	2
tim	3	2
hans	4	0
jens	5	1
frank	0	0
olaf	1	2
stefan	3	2
alekh	3	2
felix	1	0
jorge	0	2

Cities_Dictionary	
cityID	city
0	new york
1	cuppertino
2	saarbruecken

Streets_Dictionary	
streetID	streets
0	narrowstreet
1	macstreet
2	longstreet
3	unistreet
4	msstreet
5	meerweinstreet

Adaptive Dictionary

Lempel-Ziv Encoding: LZ77 & LZ78

- An innovative, radically different method was introduced in1977 by Abraham Lempel and Jacob Ziv.
- This technique (called Lempel-Ziv) actually consists of two considerably different algorithms, LZ77 and LZ78.
- Due to patents, LZ77 and LZ78 led to many variants:

		LZH	LZB	LZSS	LZR	LZ77 Variants
LZFG	LZJ	LZMW	LZT	LZC	LZW	LZ78 Variants

- The **zip** and **unzip** use the LZH technique while UNIX's **compress** methods belong to the LZW and LZC classes.

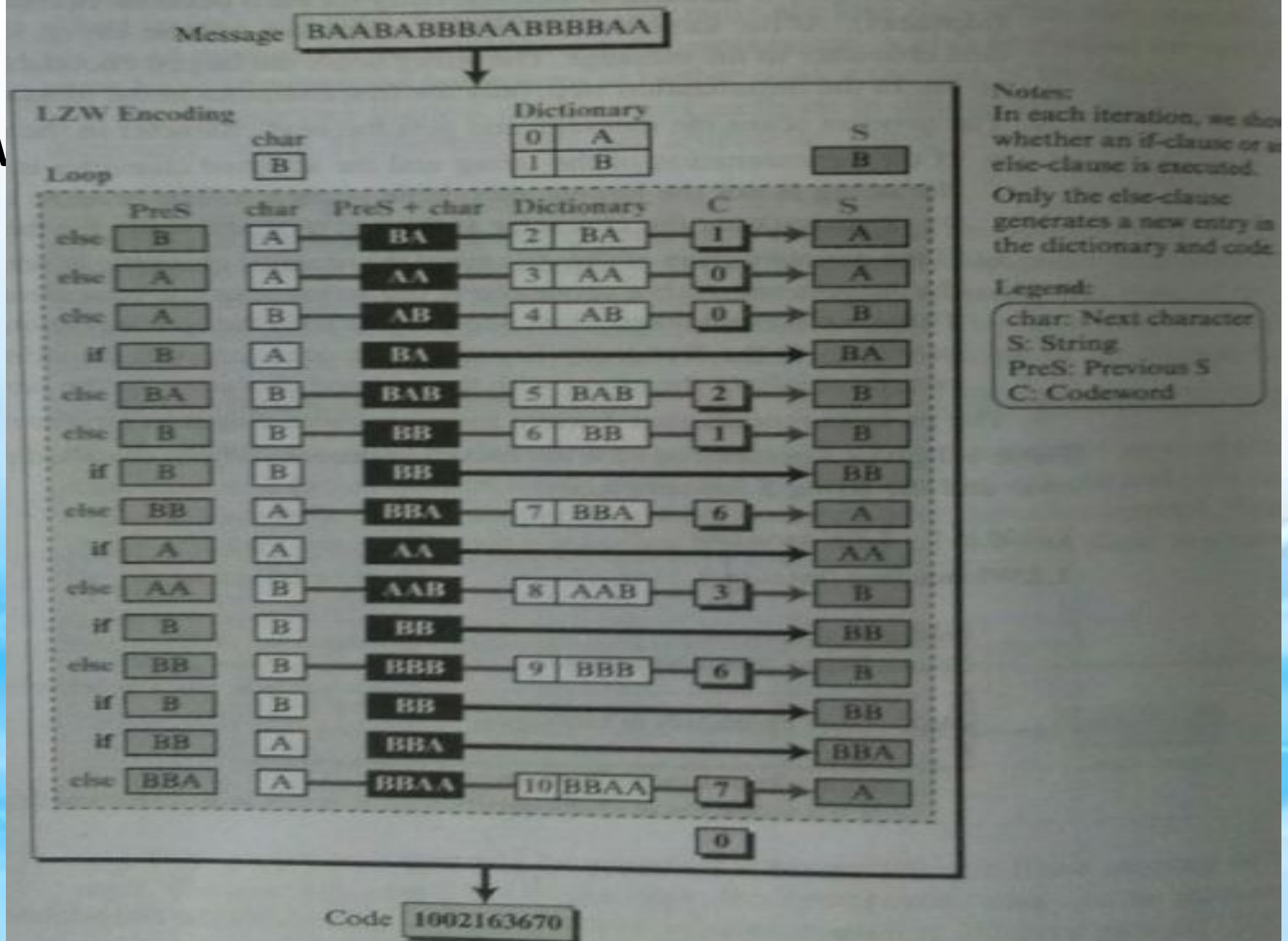
LZW Encoding

abababababab

Index	Entity	Encoder o/p
1	a	
2	b	
3	ab	1
4	ba	2
5	aba	3
6	abab	5
7	bab	4
	b	2

Encoder o/p: 123542

LZV


Message
BAABABBBBAABBBBBAA
LZW Encoding
Loop
char
B
Dictionary
0 A
1 B
S
B
PreS
char
PreS + char
Dictionary
C
S
else B A BA 2 BA 1 A
else A A AA 3 AA 0 A
else A B AB 4 AB 0 B
if B A BA BA
else BA B BAB 5 BAB 2 B
else B B BB 6 BB 1 B
if B B BB BB
else BB A BBA 7 BBA 6 A
if A A AA AA
else AA B AAB 8 AAB 3 B
if B B BB BB
else BB B BBB 9 BBB 6 B
if B B BB BB
if BB A BBA BBA
else BBA A BBAA 10 BBAA 7 A
0
Code
1002163670
Notes:
In each iteration, we show whether an if-clause or an else-clause is executed.
Only the else-clause generates a new entry in the dictionary and code.
Legend:
char: Next character
S: String
PreS: Previous S
C: Codeword

Try it yourself

Encode using LZW Technique

wabba/wabba/wabba/wabba/wabba/woo

Assume initial dictonary as follows

Index	Entry
1	/
2	a
3	b
4	o
5	w

Index	Entry
1	/
2	a
3	b
4	o
5	w
6	wa
7	ab
8	bb
9	ba
10	a/

Index	Entry
11	/w
12	wab
13	bba
14	a/w
15	wabb
16	ba/
17	/wa
18	abb
19	ba/w
20	wo

Index	Entry
21	oo
22	o/
23	/wo
24	oo/
25	/woo

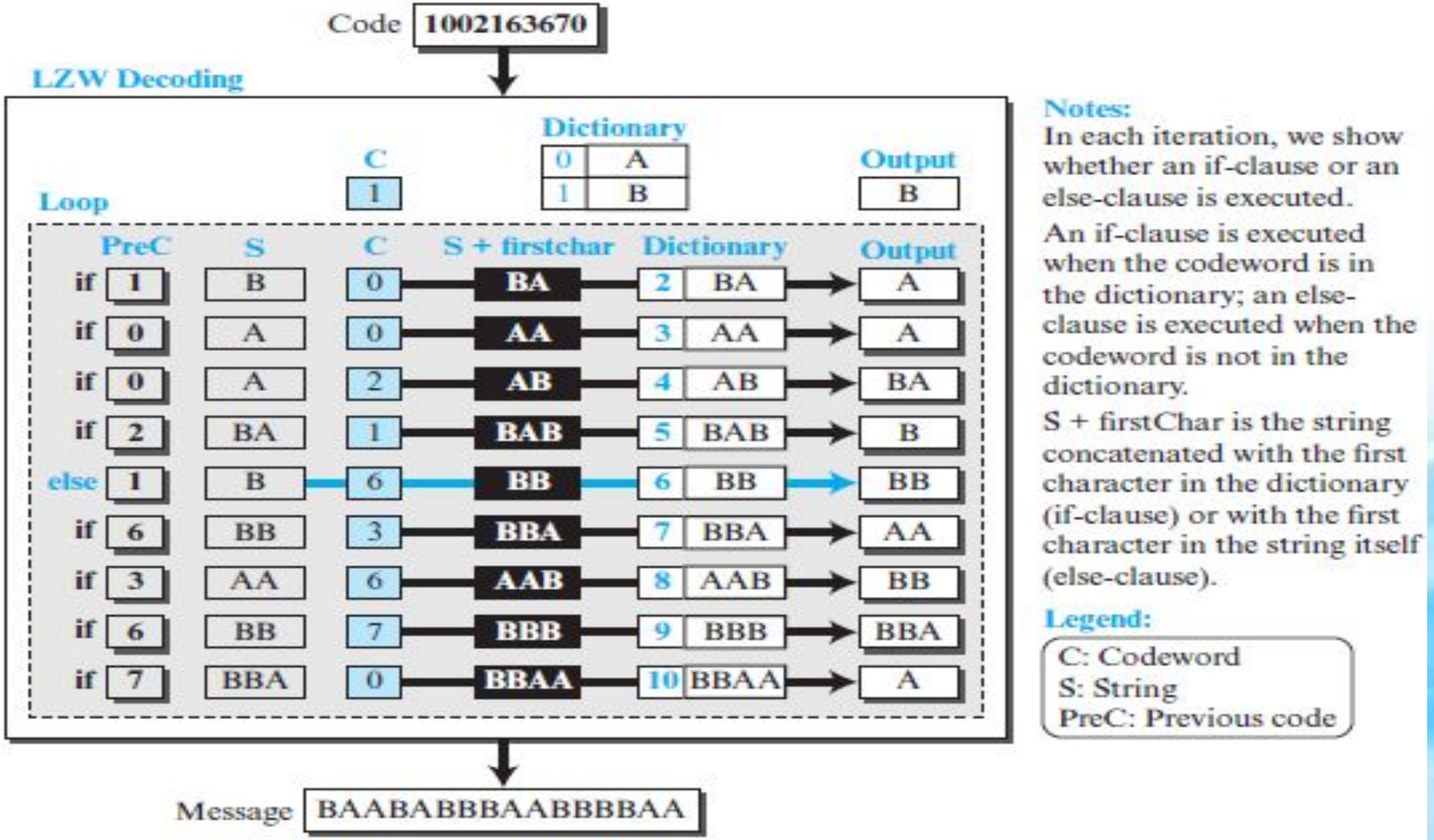
Encoder output sequence: 5 2 3 3 2 1 6 8 10 12 9 11 7 16 5 4 4 11 21 23 4

LZW Decoding

DecodedMessage	B	A	A	BA	B	BB	AA	BB	BBA	A
Encoded Message	1	0	0	2	1	6	3	6	7	0

1002163670

Index	Entity
0	A
1	B
2	BA
3	AA
4	AB
5	BAB
6	BB
7	BBA



Notes:

In each iteration, we show whether an if-clause or an else-clause is executed.

An if-clause is executed when the codeword is in the dictionary; an else-clause is executed when the codeword is not in the dictionary.

S + firstChar is the string concatenated with the first character in the dictionary (if-clause) or with the first character in the string itself (else-clause).

Legend:

C: Codeword
S: String
PreC: Previous code

Message : 123547

Dictionary

Index	Entity
1	a
2	b

Try it yourself

Decode using LZW Technique

3	1	4	6	8	4	2	1	2	5	10	6	11	13	6

Assume initial dictionary as follows

Index	Entity
1	a
2	b
3	r
4	t

DecodedMessage	r	a	t	at	ata	t	b	a	b	ra	tb	at	ba	br	at
Encoded Message	3	1	4	6	8	4	2	1	2	5	10	6	11	13	6

Index	Entity
1	a
2	b
3	r
4	t
5	ra
6	at
7	ta
8	ata
9	atat
10	tb
11	ba
12	ab
13	br

B) Huffman Coding

- Lossless
- Huffman coding is a form of statistical coding
- Not all characters occur with the same frequency!
- Yet all characters are allocated the same amount of space
- assigns variable-length codes to input characters, lengths of the assigned codes are based on the frequencies of corresponding characters.
- The most frequent character gets the smallest **code** and the least frequent character gets the largest **code**.
- All codes have unique **prefixes**

Huffman algorithm

1. Scan text to be compressed and tally occurrence of all characters.
2. Sort or prioritize characters based on number of occurrences in text.
3. Build Huffman code tree based on prioritized list.
4. Perform a traversal of tree to determine all code words.
5. Scan text again and create new file using the Huffman codes.

Building a Tree

Scan the original text

- Consider the following short text:
  - *Eerie eyes seen near lake.*
- Count up the occurrences of all characters in the text

Building a Tree

Scan the original text

Eerie eyes seen near lake.

- What characters are present?

E e r i space y s n a r l k .

Building a Tree

Scan the original text

Eerie eyes seen near lake.

- What is the frequency of each character in the text?

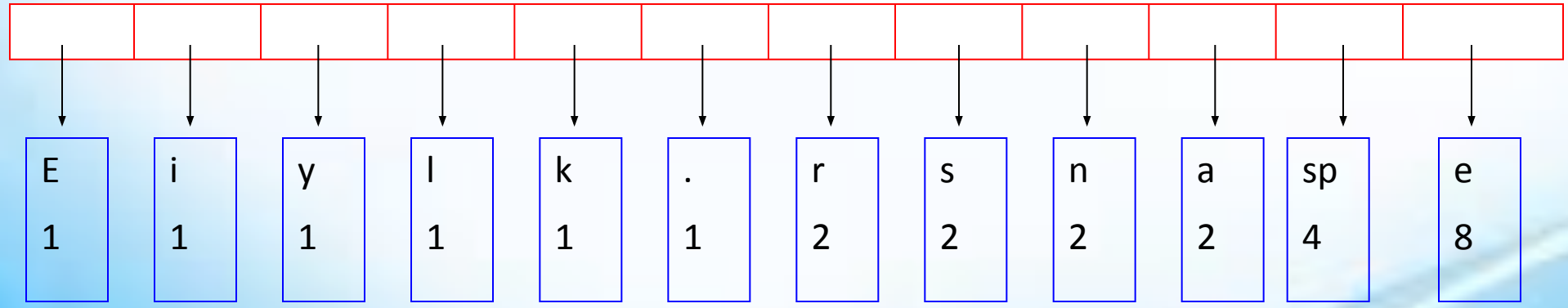
Char	Freq.	Char	Freq.	Char	Freq.
E	1	y	1	k	1
e	8	s	2	.	1
r	2	n	2		
i	1	a	2		
space	4	l	1		

Building a Tree
Prioritize characters

- Create binary tree nodes with character and frequency of each character
- Place nodes in a priority queue
  - The lower the occurrence, the higher the priority in the queue

Building a Tree

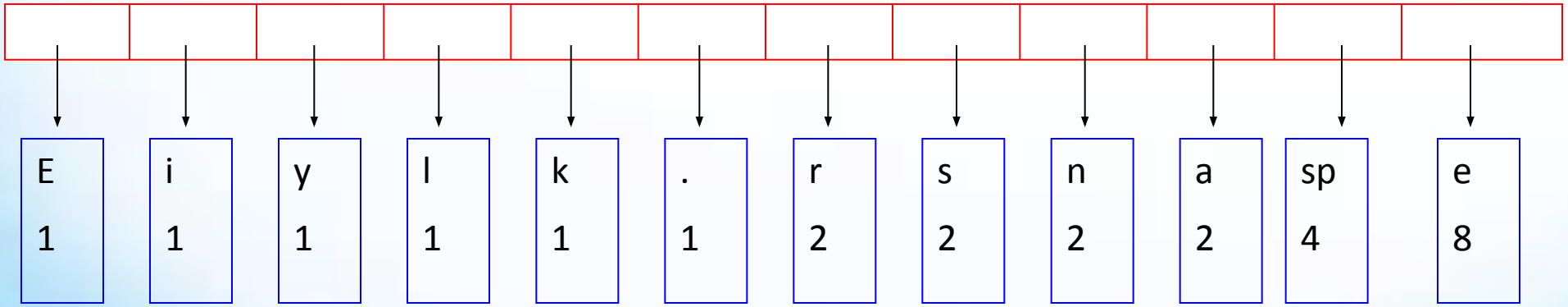
- The queue after inserting all nodes



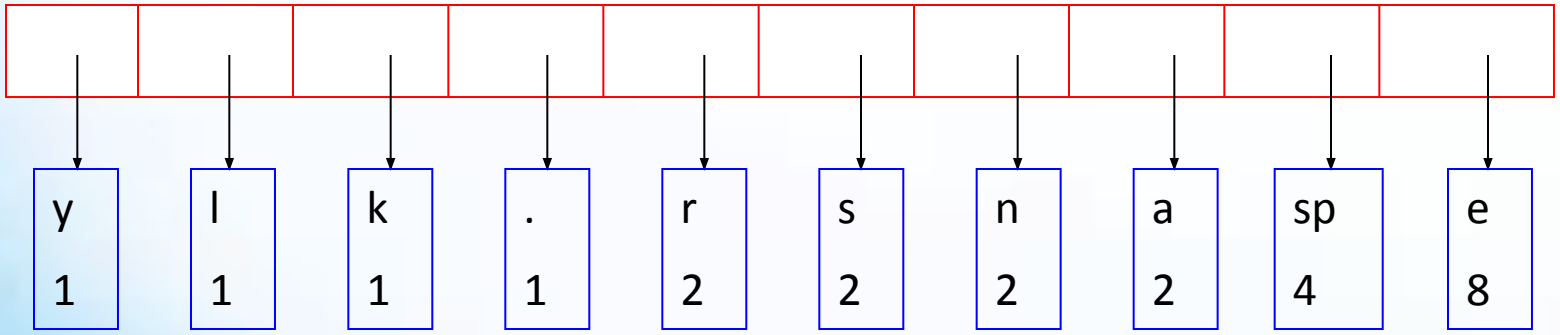
Building a Tree

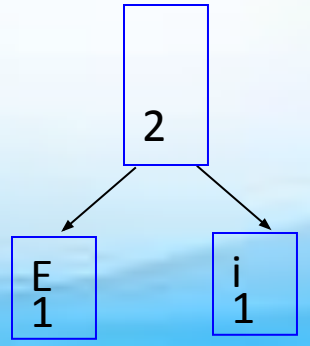
- While priority queue contains two or more nodes
  - Create new node
  - Dequeue node and make it left subtree
  - Dequeue next node and make it right subtree
  - Frequency of new node equals sum of frequency of left and right children
  - Enqueue new node back into queue

Building a Tree

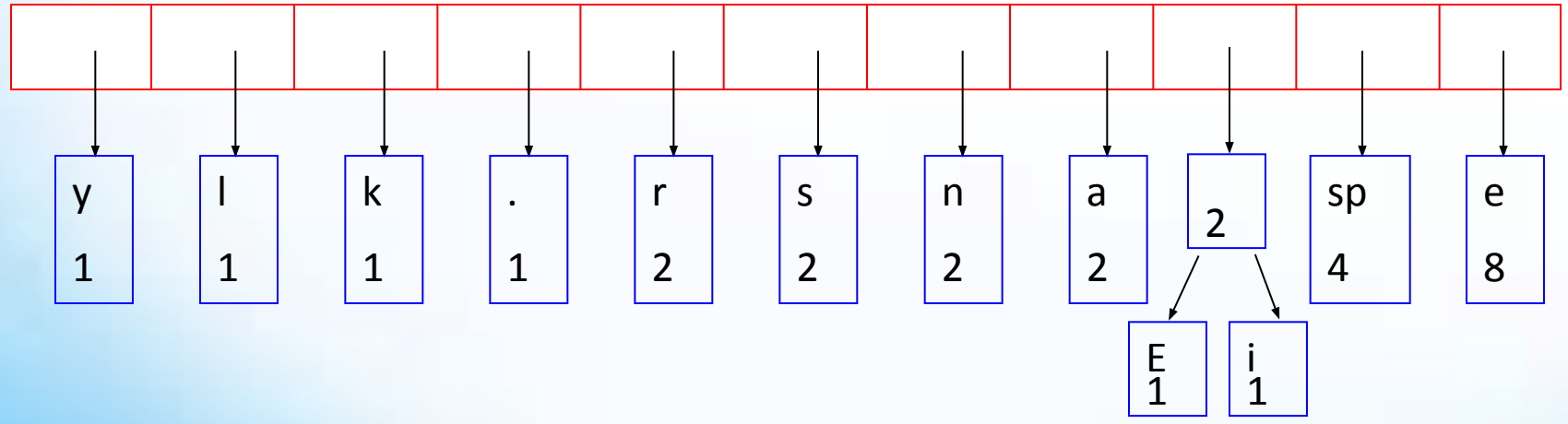


Building a Tree

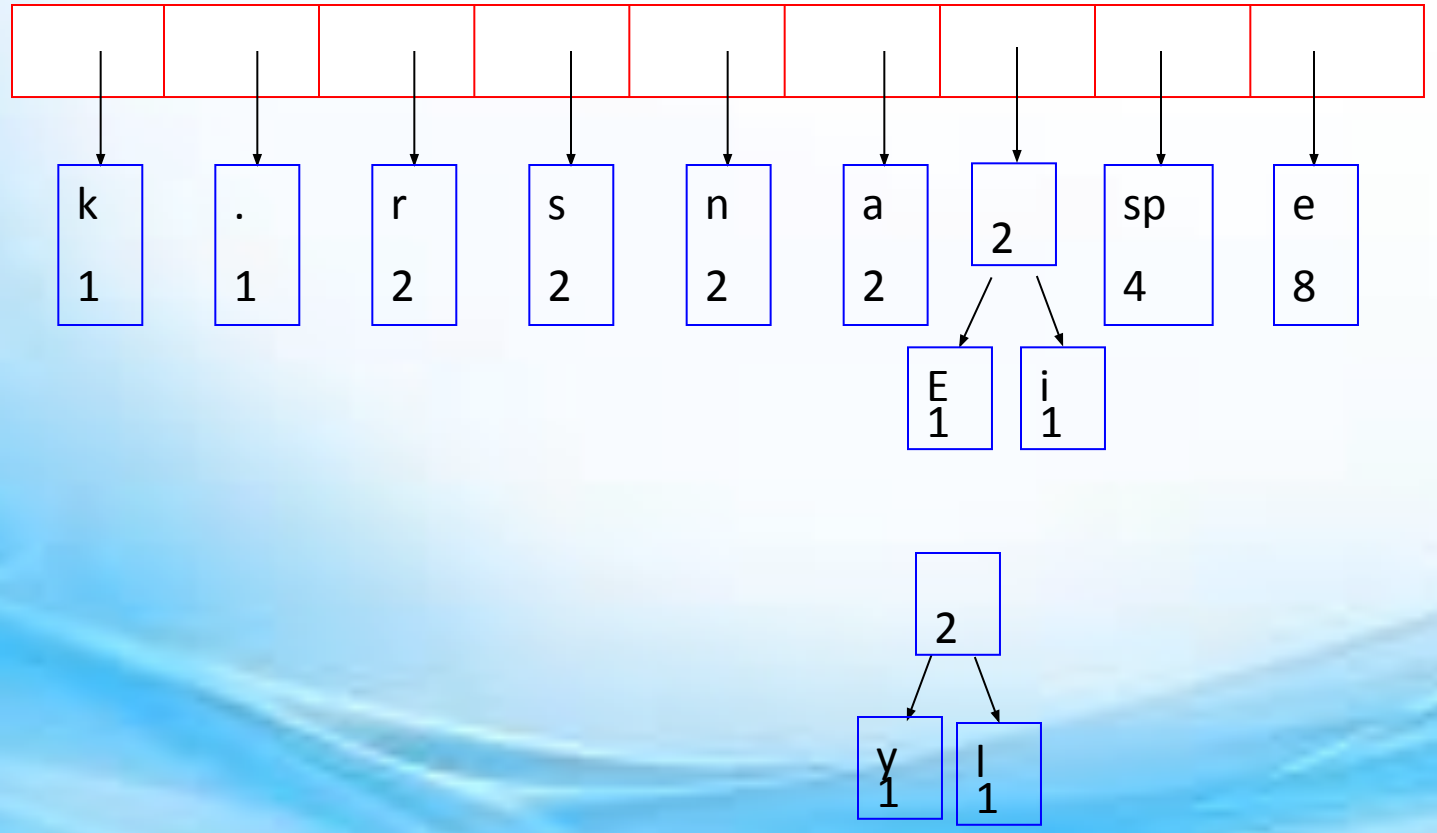




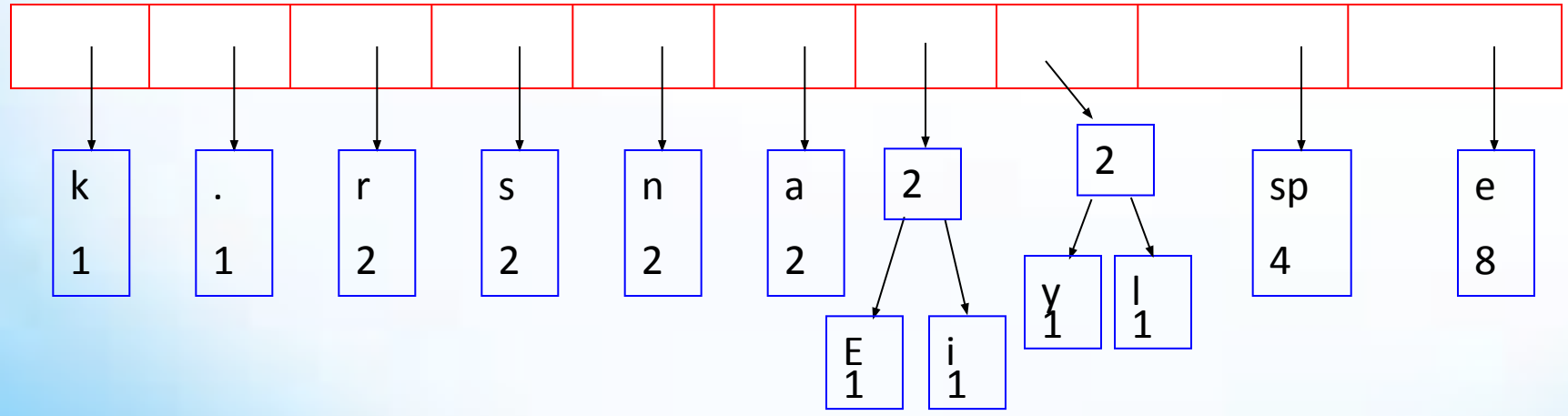
Building a Tree



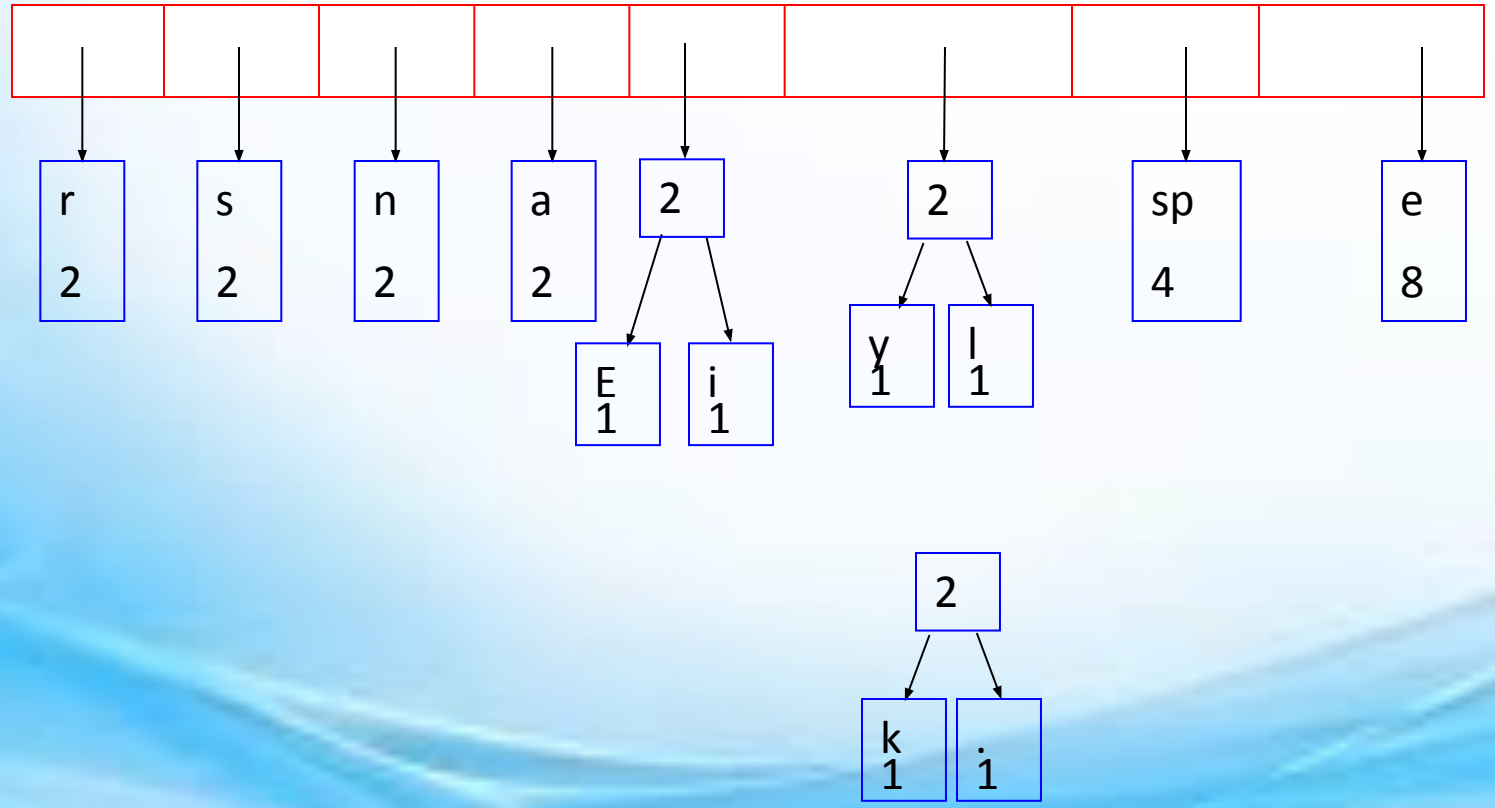
Building a Tree



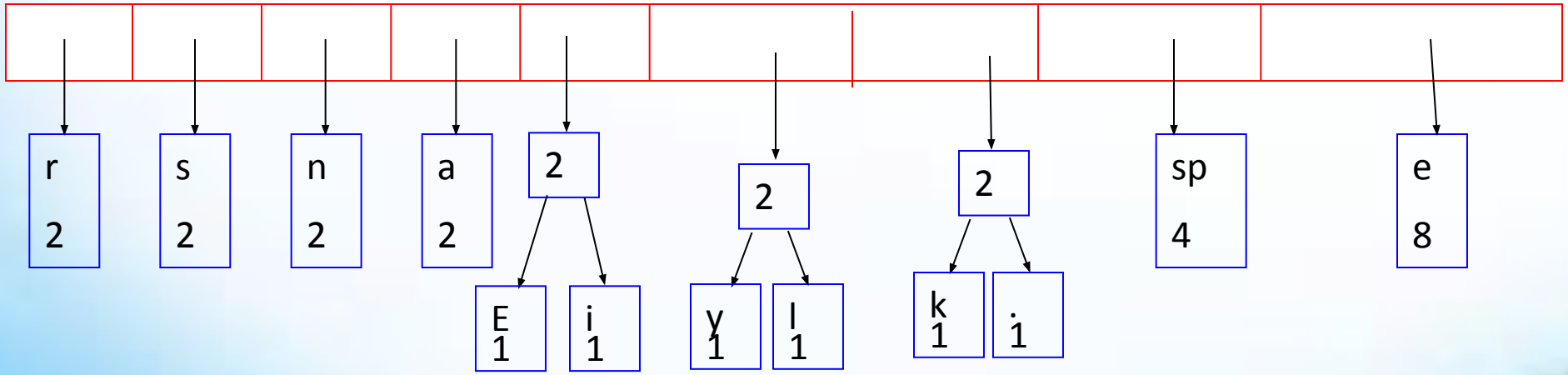
Building a Tree



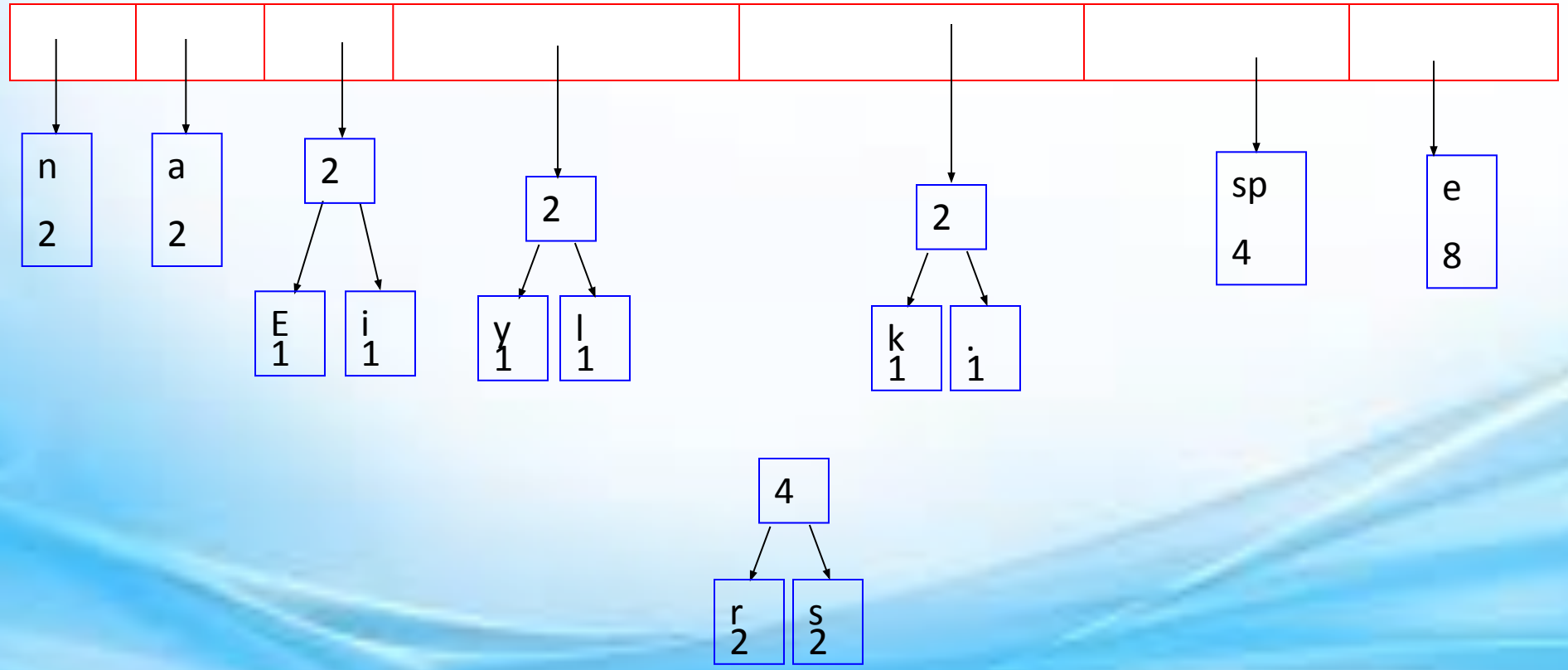
Building a Tree



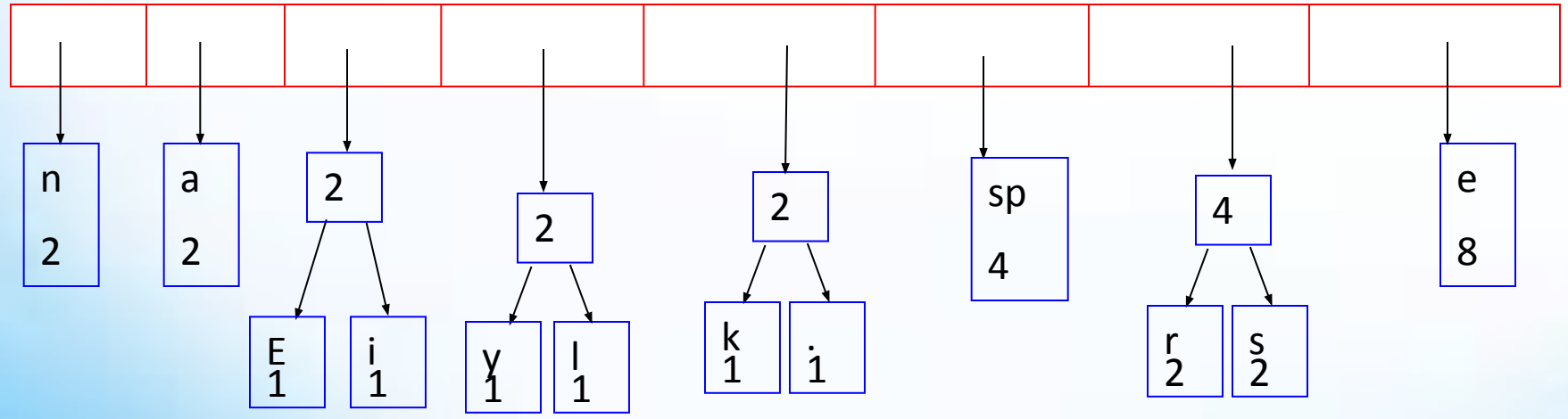
Building a Tree



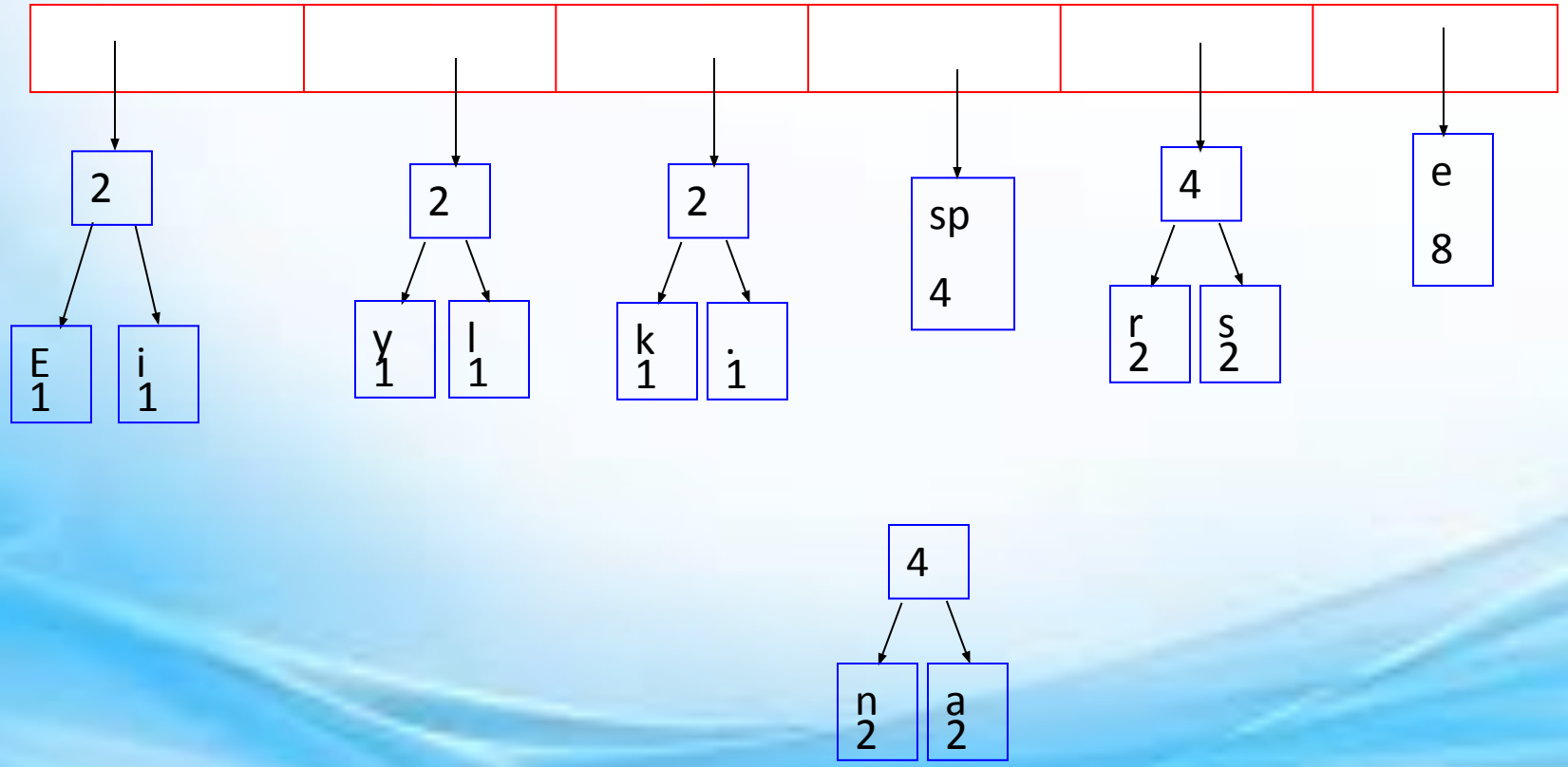
Building a Tree



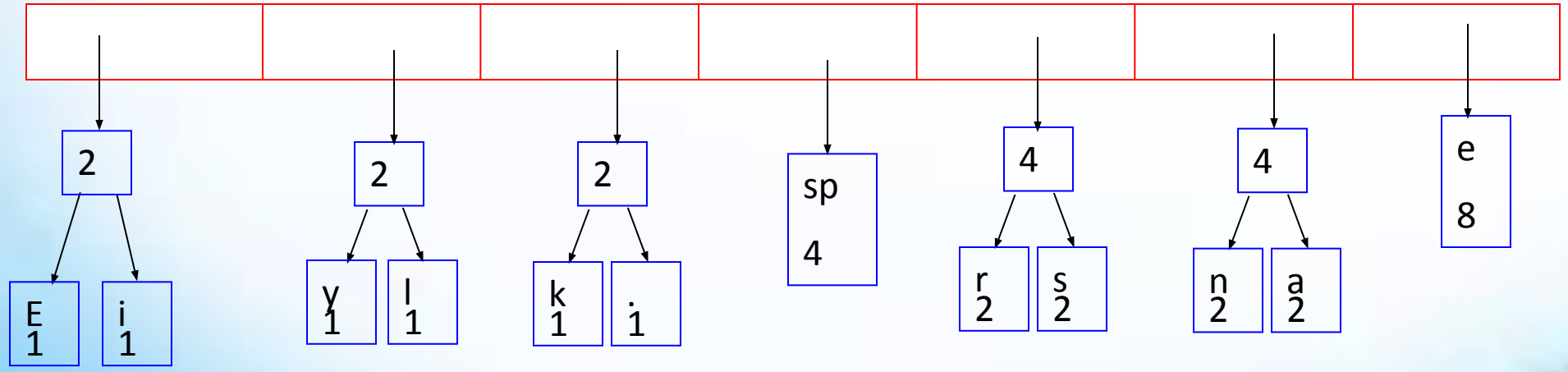
Building a Tree



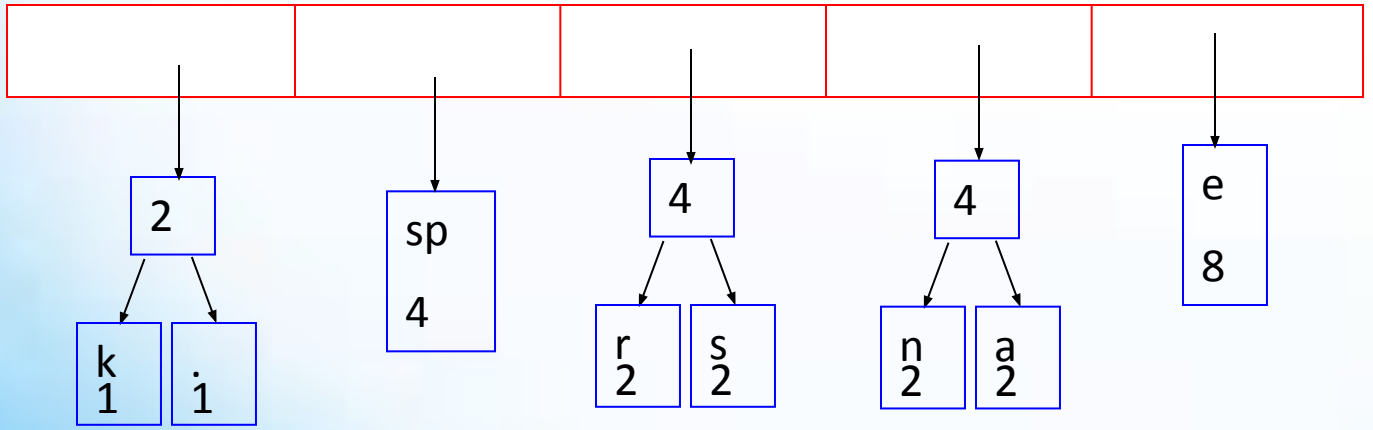
Building a Tree

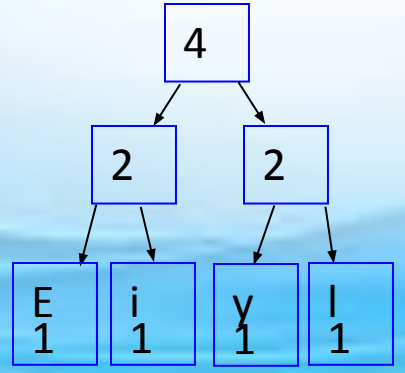


Building a Tree

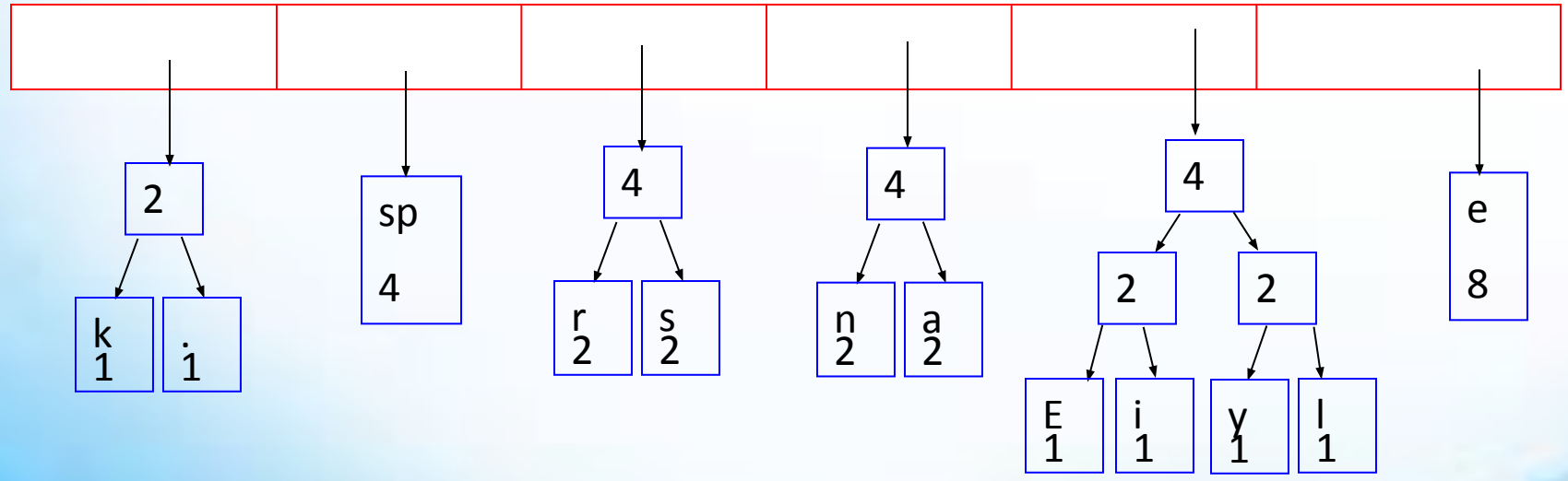


Building a Tree

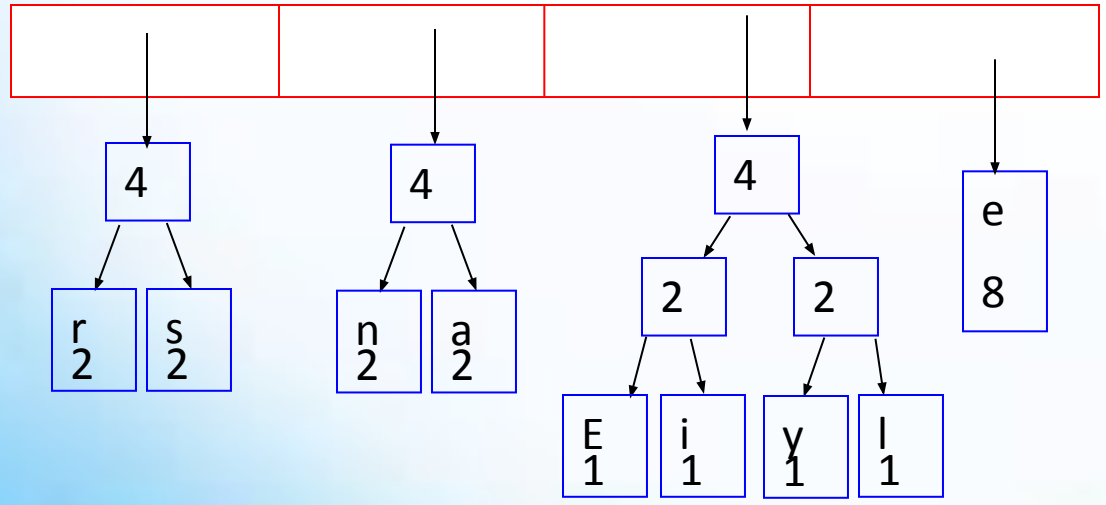


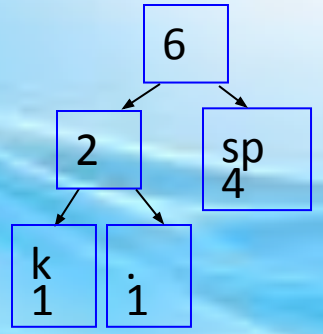


Building a Tree

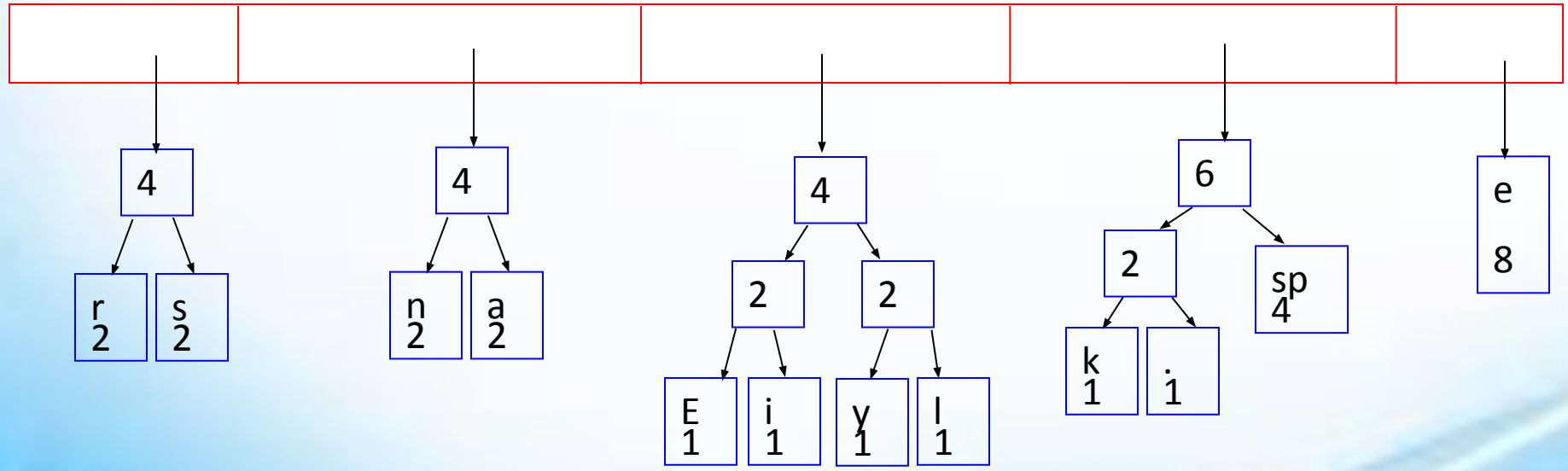


Building a Tree



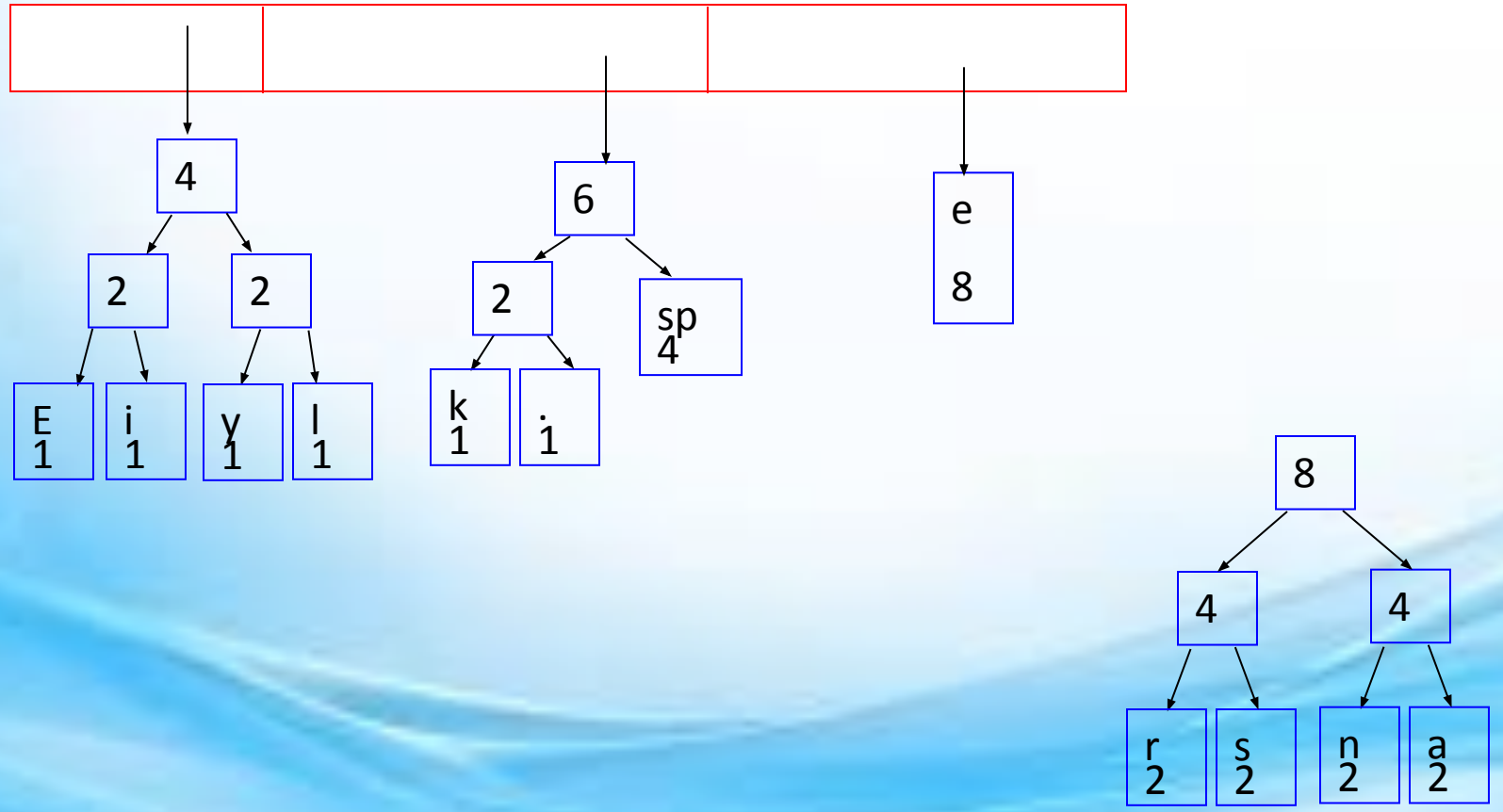


Building a Tree

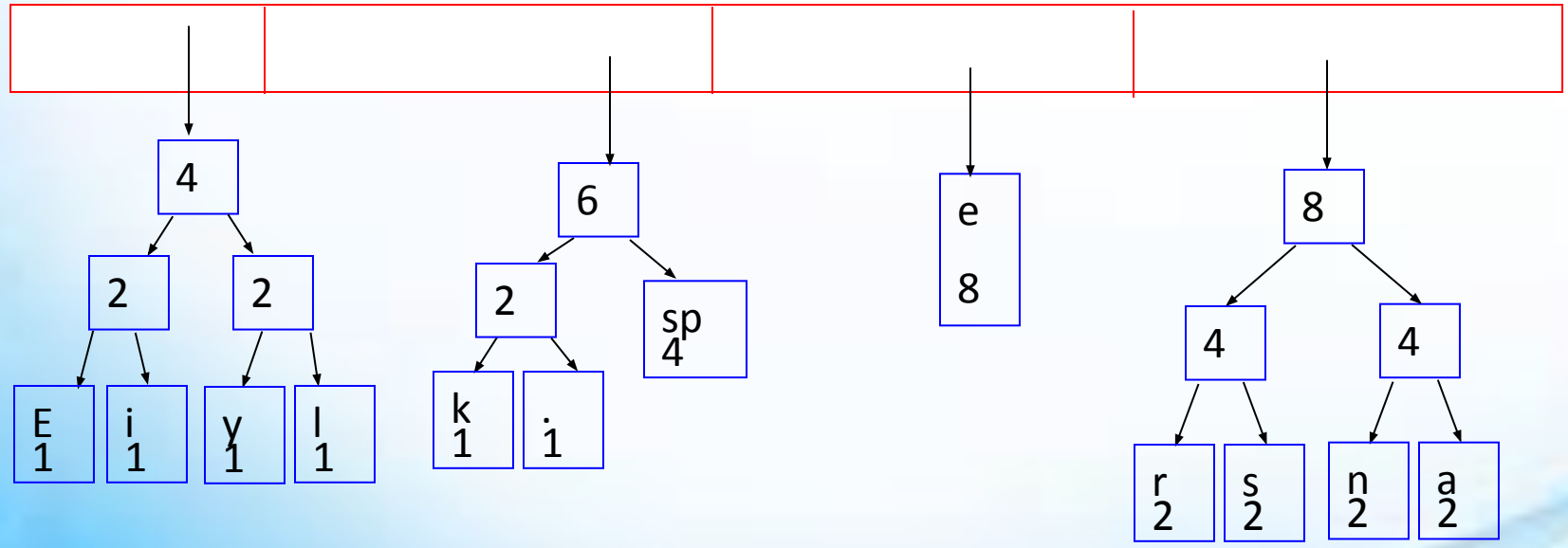


What is happening to the characters with a low number of occurrences?

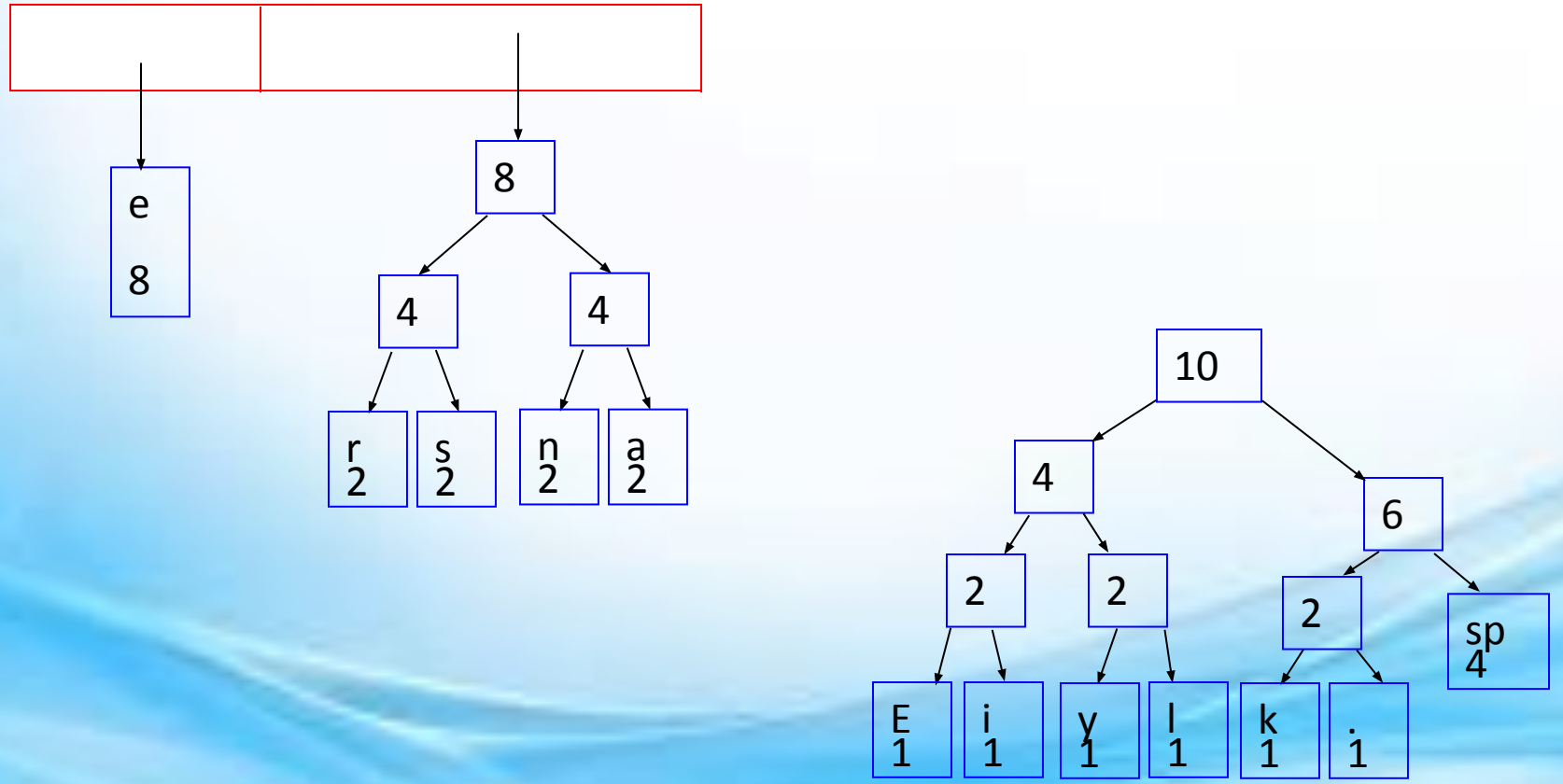
Building a Tree



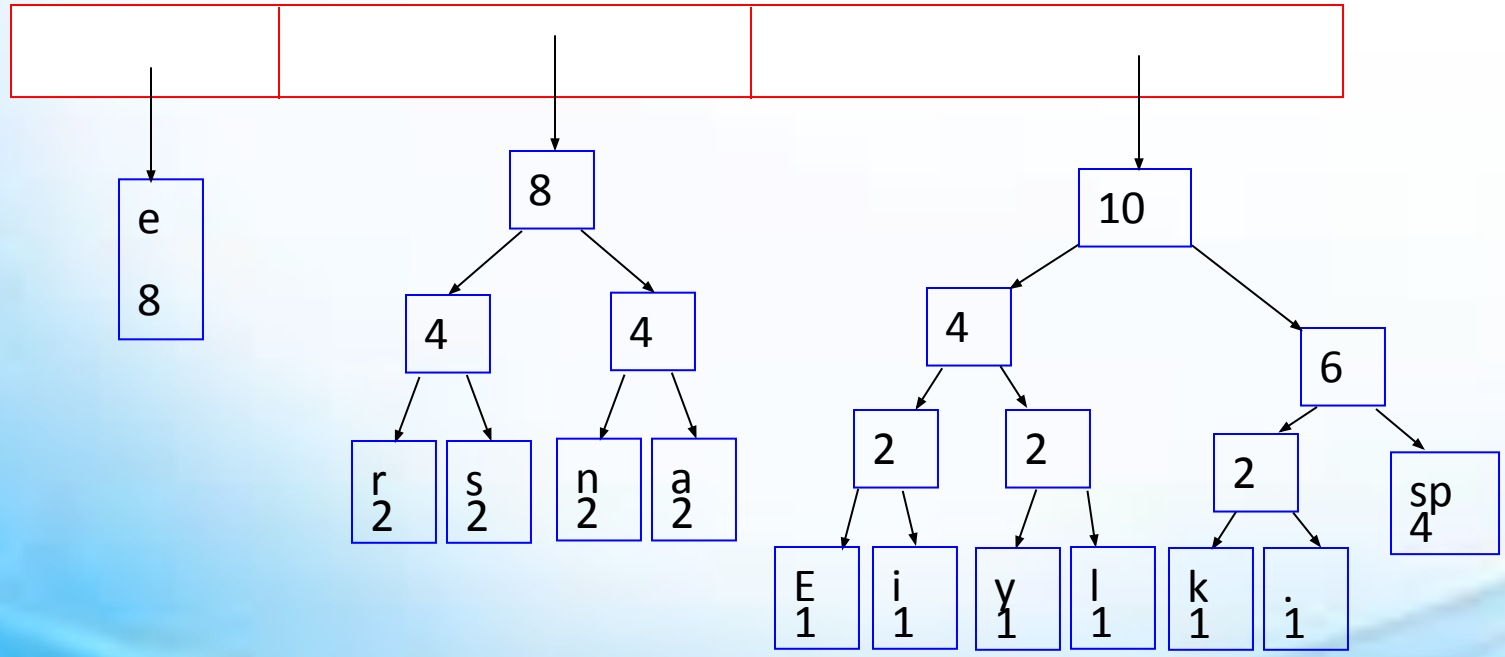
Building a Tree



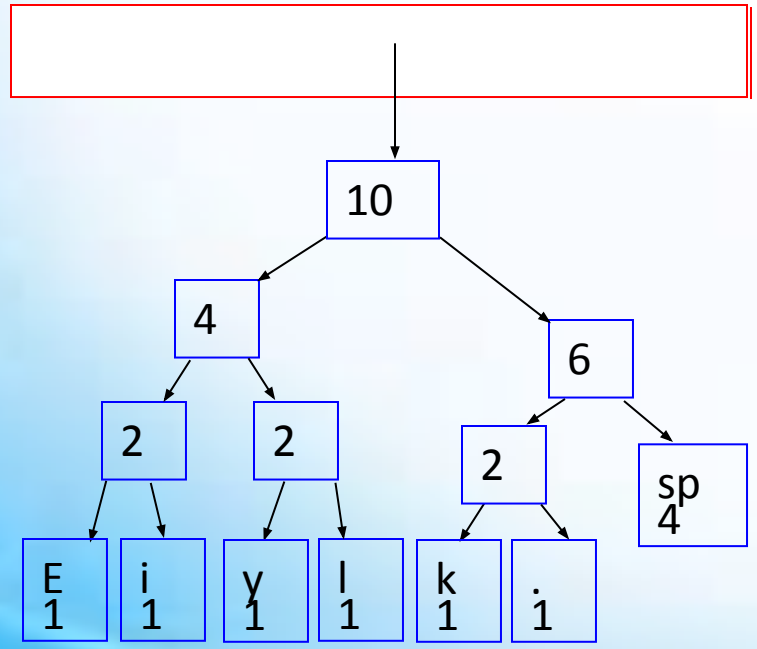
Building a Tree

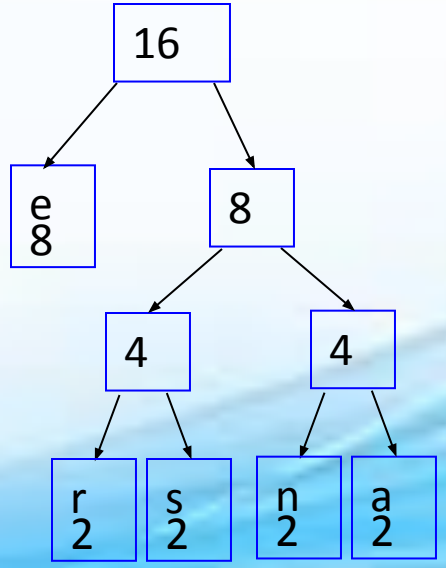


Building a Tree

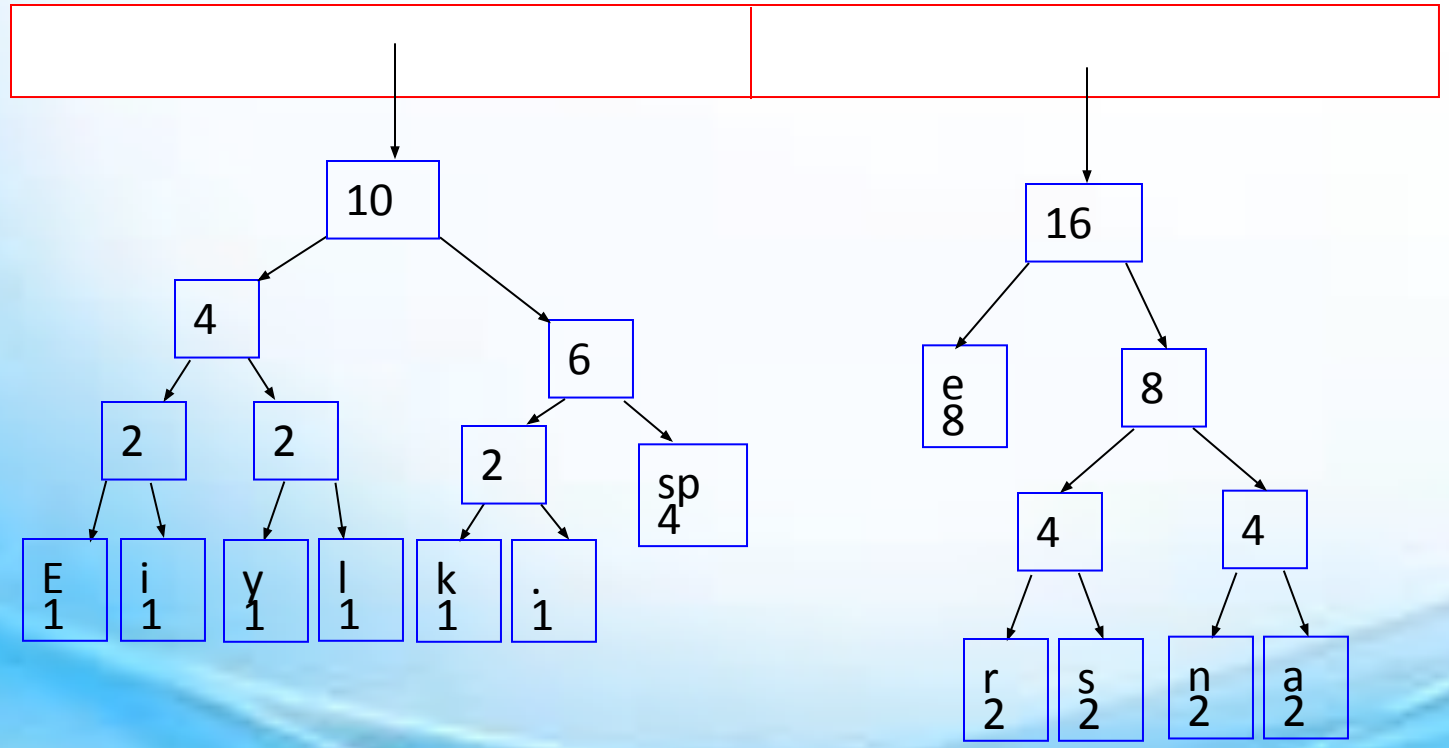


Building a Tree

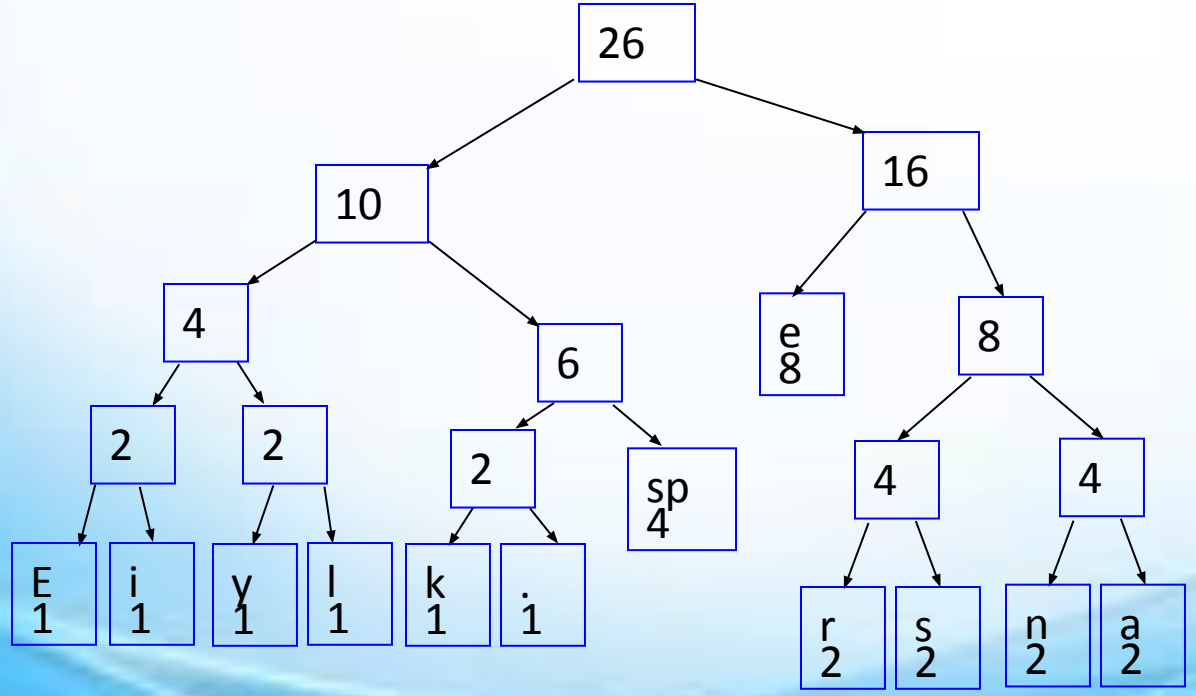




Building a Tree



Building a Tree

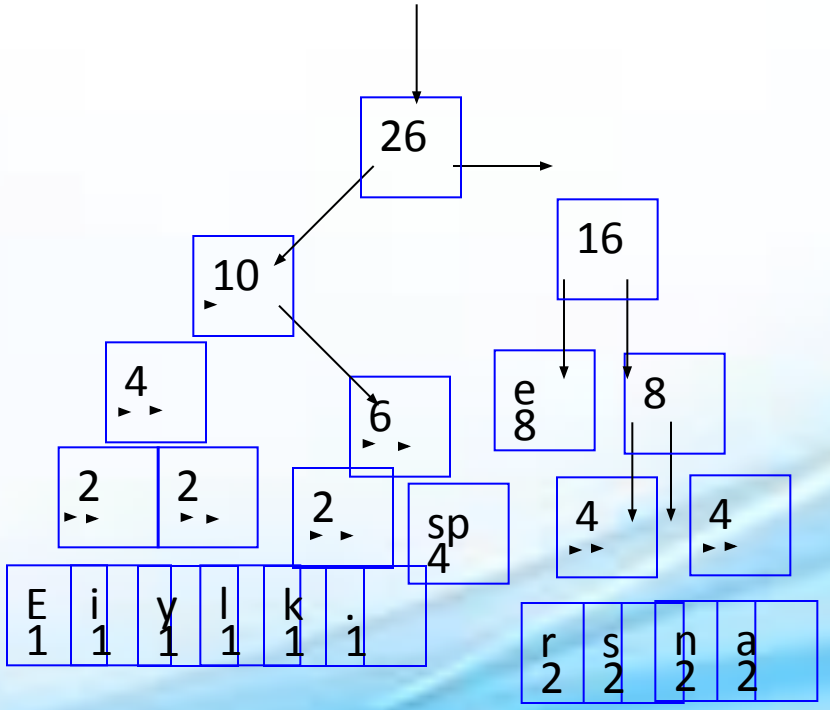


Building a Tree

Dequeue the single node left in the queue.

This tree contains the new code words for each character.

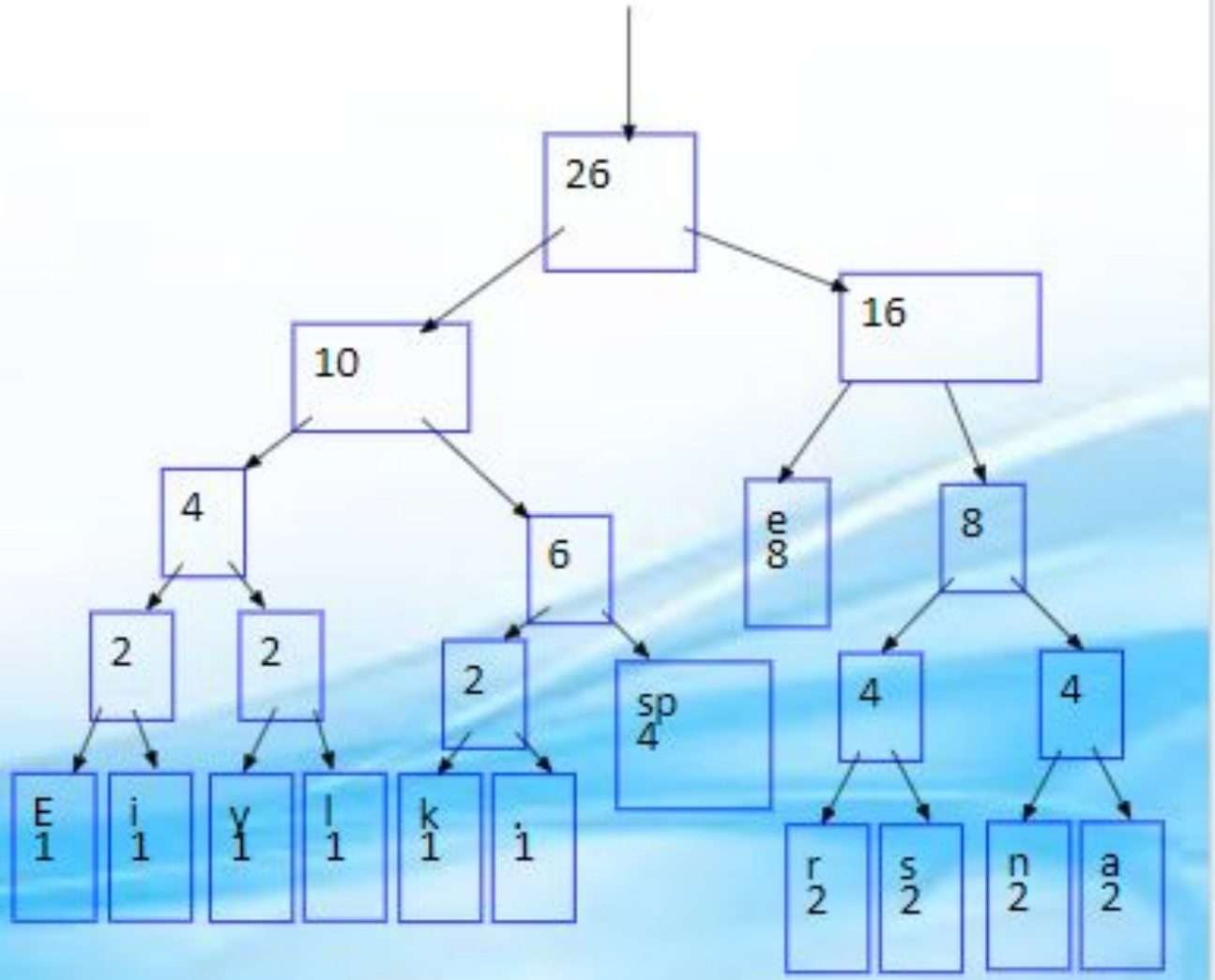
Frequency of root node should equal number of characters in text.



Eerie eyes seen near lake. □ 26 characters

Encoding the File
Traverse Tree for Codes

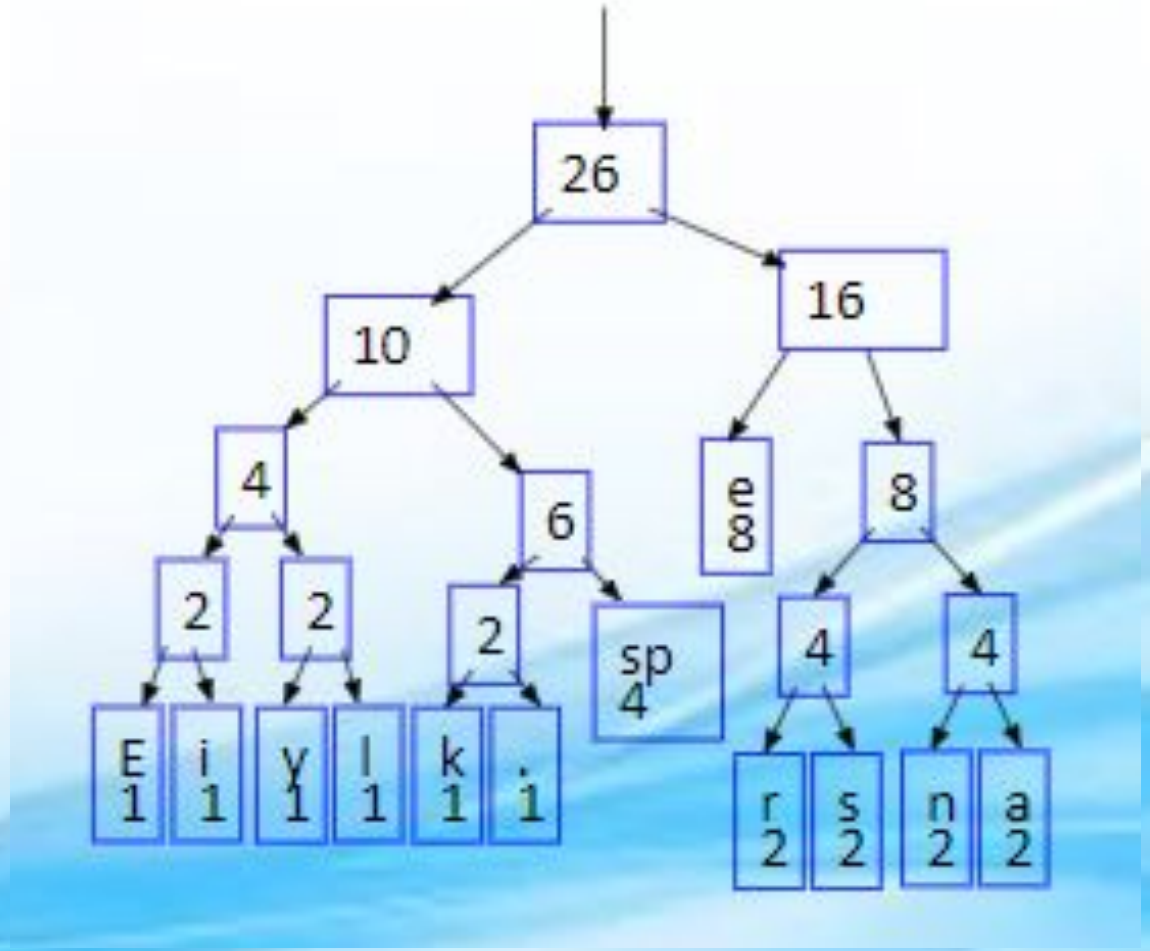
- Perform a traversal of the tree to obtain new code words
- Going left is a 0 going right is a 1
- code word is only completed whe a leaf node is reached



Encoding the File

Coding Table -Traverse Tree for Codes

Char	Code
E	0000
i	0001
y	0010
l	0011
k	0100
.	0101
space	011
e	10
r	1100
s	1101
n	1110
a	1111



Encoding the File

- Rescan text and encode file using new code words

Eerie eyes seen near lake.

```
000010110000011001110001010111010011111010111111000110011111110100100101
```

Char	Code
E	0000
i	0001
y	0010
l	0011
k	0100
.	0101
space	011
e	10
r	1100
s	1101
n	1110
a	1111

- **Why is there no need for a separator character?**

Encoding the File

Results

- Have we made things any better?
- 73 bits to encode the text
- ASCII would take 8 * 26 = 208 bits

```
00001011000001100111000101011011010011111010111100011001111110100100101
```

If modified code used 4 bits per character are needed.

Total bits 4 * 26 = 104.

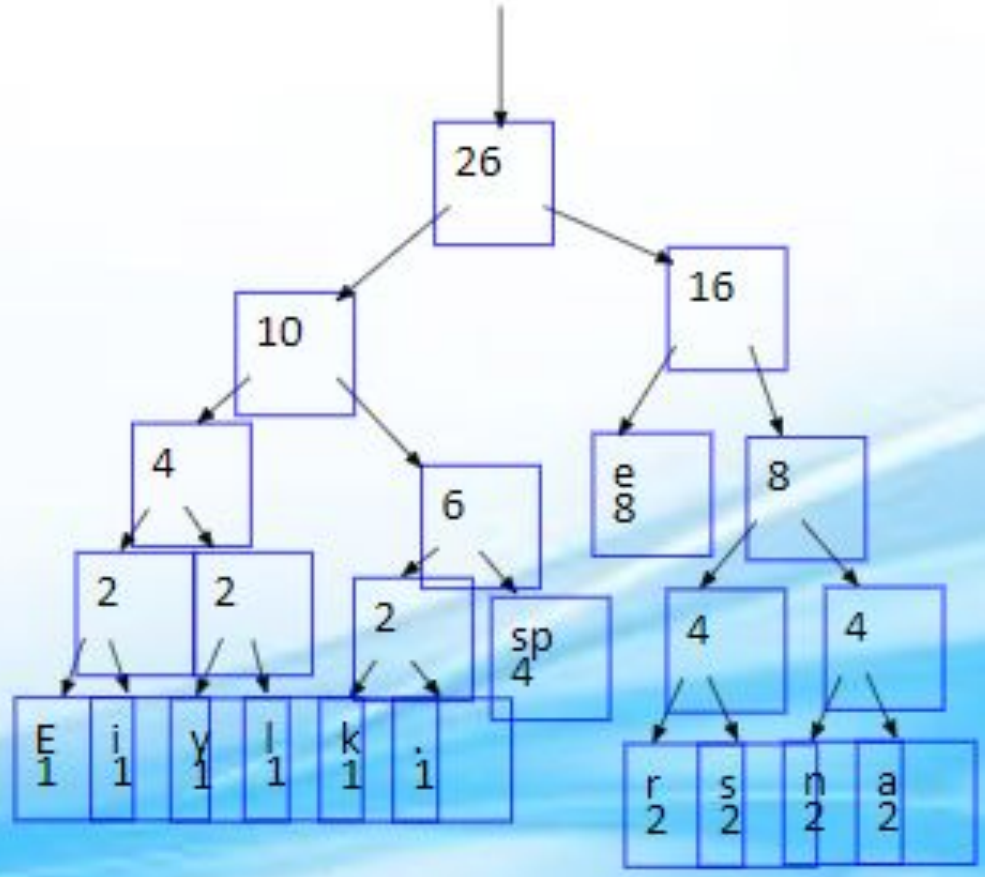
Decoding the File

- How does receiver know what the codes are?
- Tree constructed for each text file.
  - Considers frequency for each file
  - Big hit on compression, especially for smaller files
- Tree predetermined
  - based on statistical analysis of text files or file types
- Data transmission is bit based versus byte based

Decoding the File

- Once receiver has tree it scans incoming bit stream
- 0 ⇒ go left
- 1 ⇒ go right

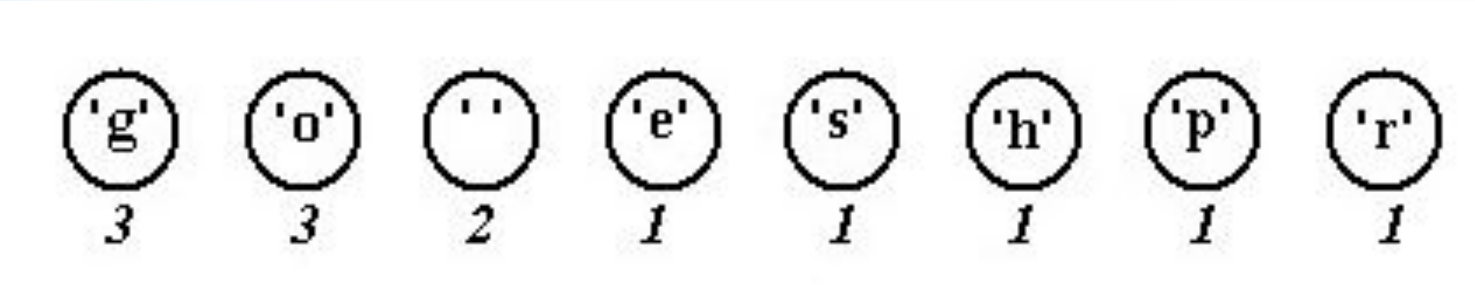
```
00001011000001100111000101011
01101001111101011111100011001
11111010010010 1
```

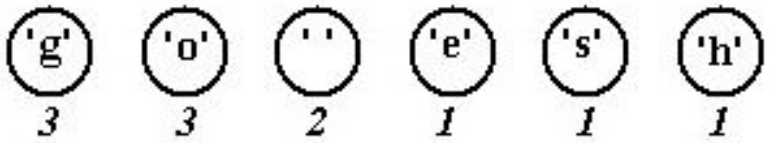


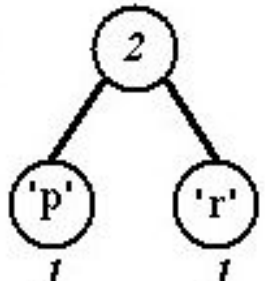
Summarizing, to find number of bits for encoding a given message

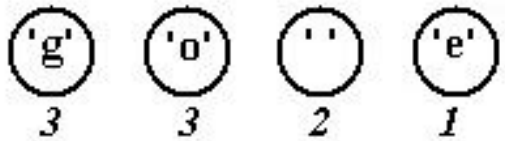
1. First calculate frequency of characters if not given
2. Generate Huffman Tree
3. Calculate number of bits using frequency of characters and number of bits required to represent those characters.

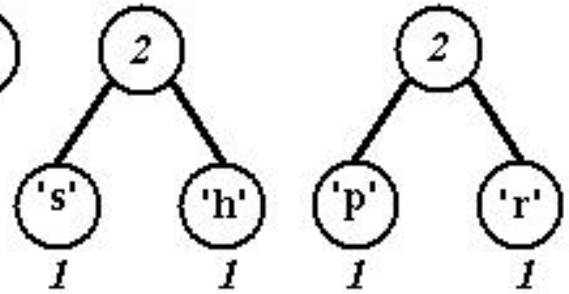
How many bits may be required for encoding the message "go go gophers"?



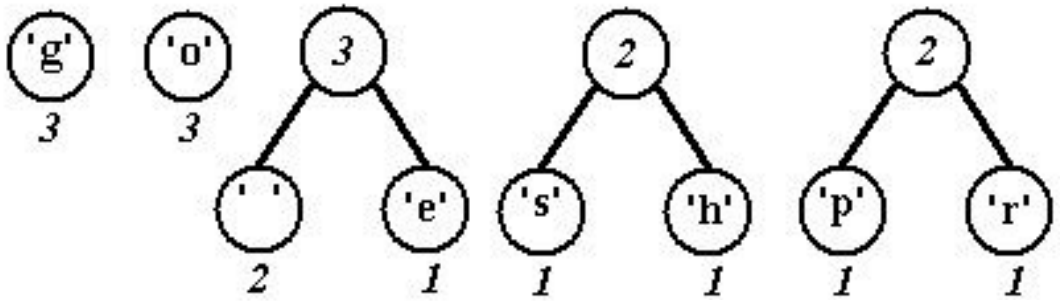

'g' 3
'o' 3
' ' 2
'e' 1
's' 1
'h' 1

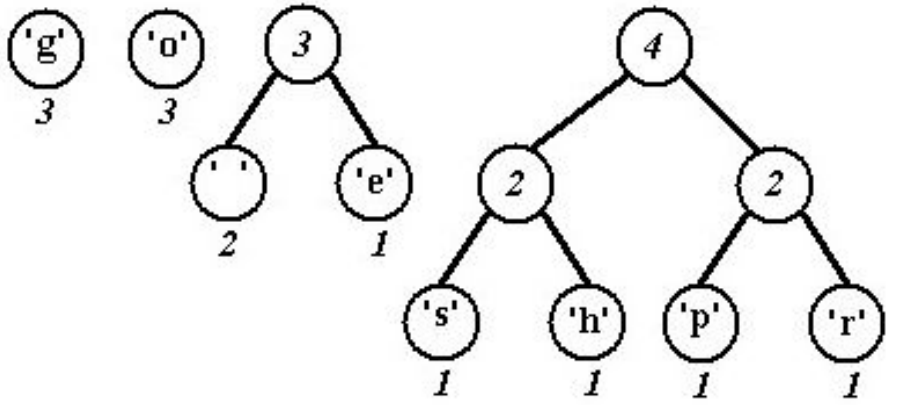

2
'p' 1
'r' 1

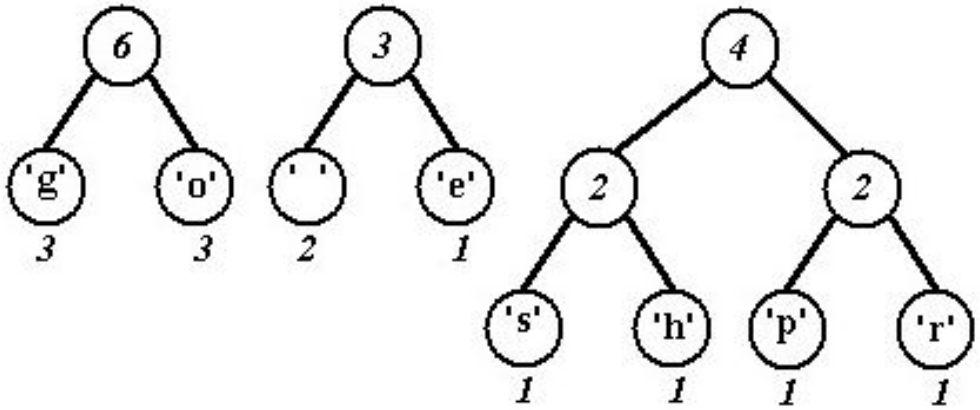

'g' 3
'o' 3
' ' 2
'e' 1

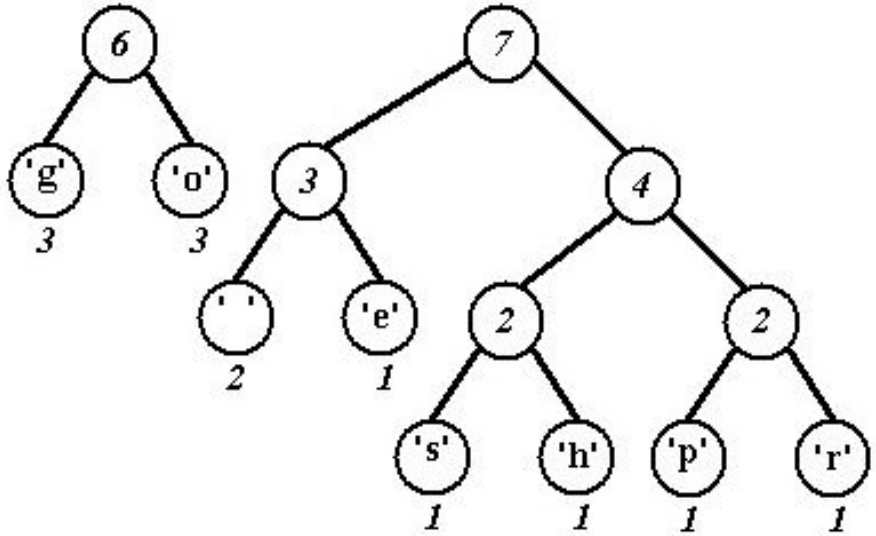

2
's' 1
'h' 1

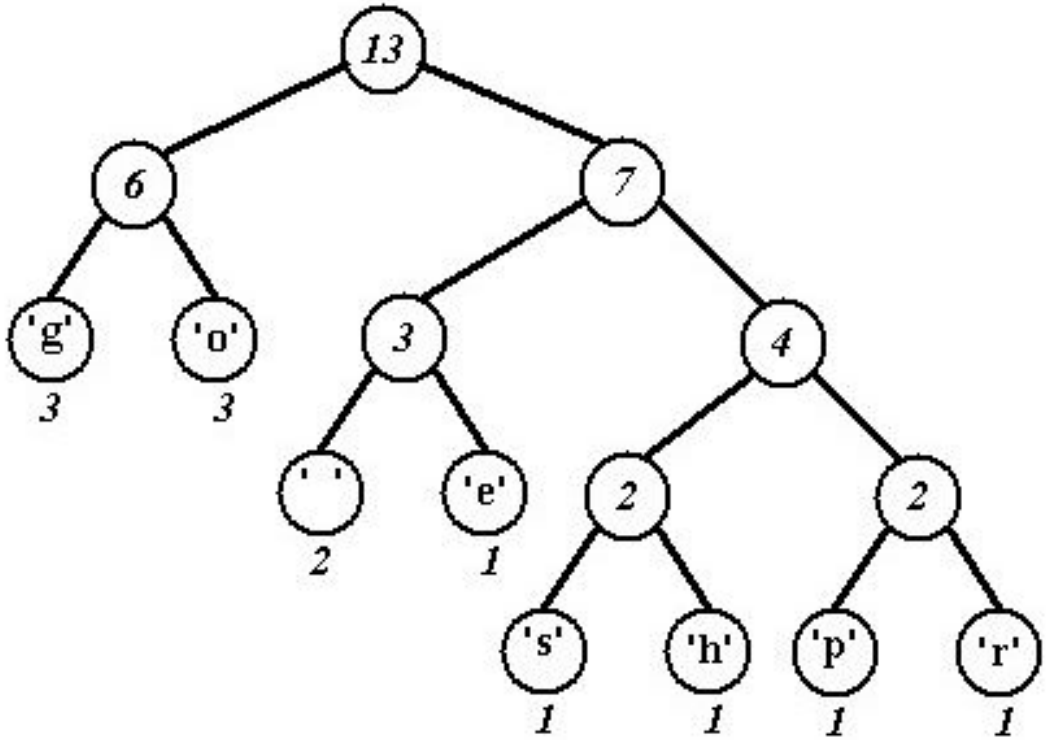
2
'p' 1
'r' 1


'g' 3
'o' 3
3
' ' 2
'e' 1
2
's' 1
'h' 1
2
'p' 1
'r' 1


'g' 3
'o' 3
3
' ' 2
'e' 1
4
2
's' 1
'h' 1
2
'p' 1
'r' 1


6
'g'
'o'
3
3
3
' '
'e'
2
1
4
2
2
's'
'h'
'p'
'r'
1
1
1
1


6
7
'g'
'o'
3
4
3
3
' '
'e'
2
2
2
1
's'
'h'
'p'
'r'
1
1
1
1


13
6
7
'g'
'o'
3
4
3
3
' '
'e'
2
2
2
1
's'
'h'
'p'
'r'
1
1
1
1

Character encoding

37 bits are used to encode "go go gophers"

Character	Binary Code	No of bits	Frequency
g	00	2	3
o	01	2	3
p	1110	4	1
h	1101	4	1
e	101	3	1
r	1111	4	1
s	1100	4	1
space	100	3	2

Average code length (*Entropy*)

= ∑ (frequencyi x code lengthi) / ∑ (frequencyi)

= { (3 x 2) + (3 x 2) + (1 x 4) + (1 x 4) + (1 x 3) + (1 x 4) + (1 x 4)
+(3 x 2) } / (3 +3+1+1+1+1+1+2)

= **2.846**

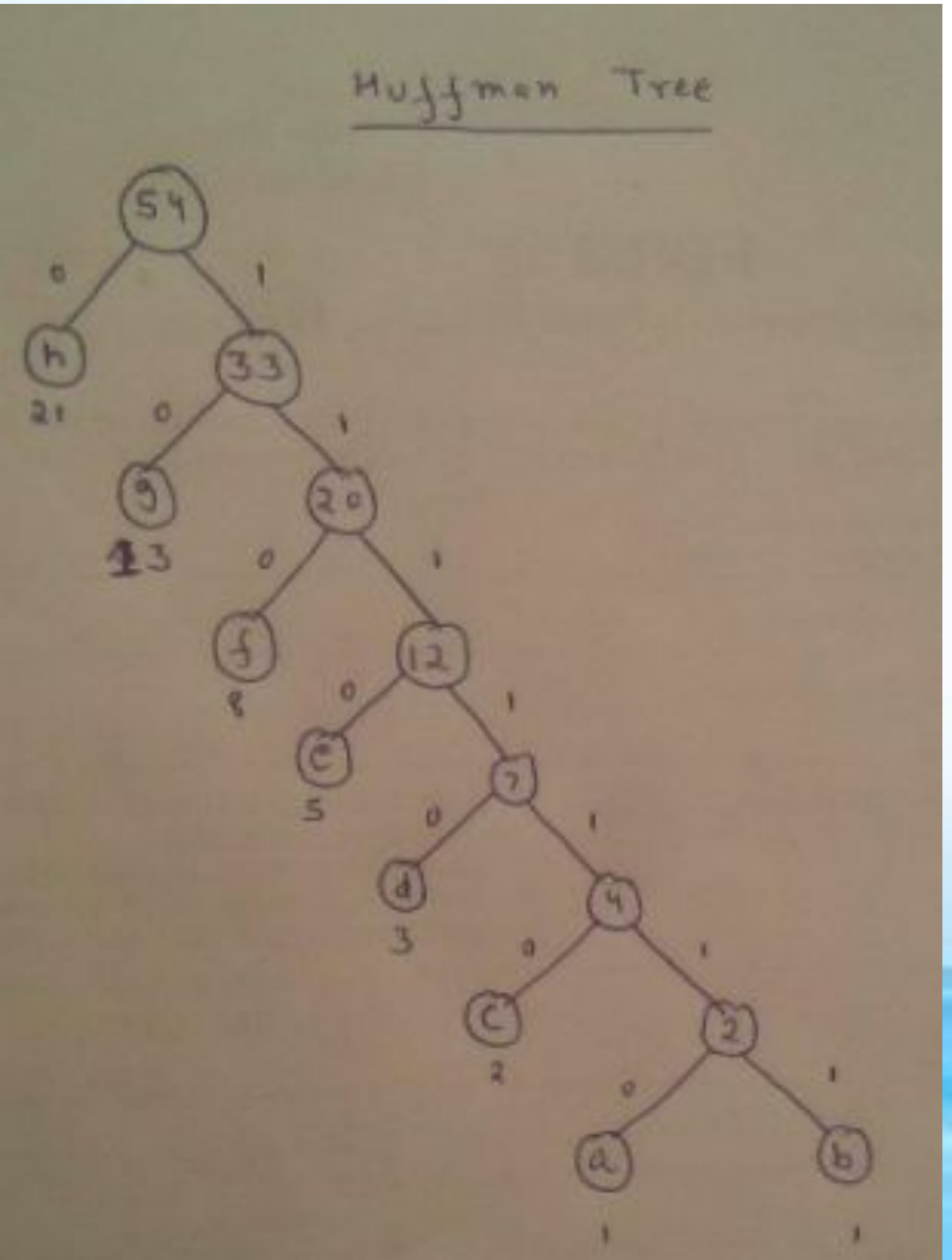
Total number of bits

= Total number of characters in the message x Average code length per character

= 13 x 2.846 = 36.998 ≅ **37**

Decode message 110111100111010 for characters having frequency

a : 1, b : 1, c : 2, d : 3, e : 5, f : 8, g : 13, h : 21



Sol:1. Construct Tree based on frequency

2. Traverse the tree to get codes as follows

a – 1111110

b – 1111111

c – 111110

d – 11110

e – 1110

f – 110

g – 10

h – 0

3. Decode message by substituting

110 11110 0 1110 10

f d h e g

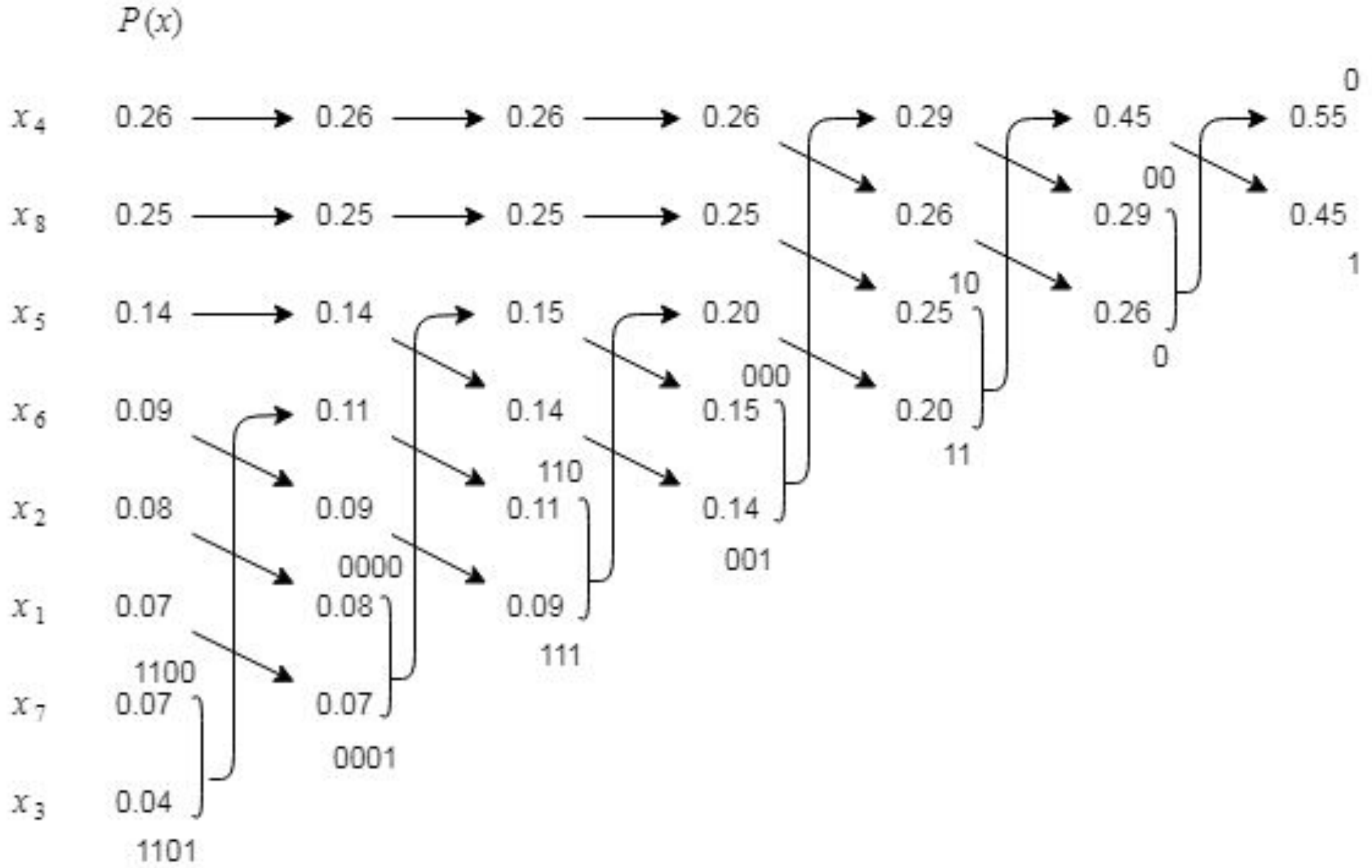
Example 21

Construct Huffman code for the given symbols

{x1, x2 , ... x8} with probabilities

P(x) = {0.07,0.08,0.04,0.26,0.14,0.09,0.07,0.25}.

Find the code efficiency.


P(x)
x4
x8
x5
x6
x2
x1
x7
x3
0.26
0.25
0.14
0.09
0.08
0.07
0.07
0.04
0.26
0.25
0.14
0.11
0.09
0.08
0.07
0.26
0.25
0.15
0.14
0.11
0.09
0.26
0.25
0.20
0.15
0.14
0.29
0.26
0.25
0.20
0.45
0.29
0.26
0.55
0.45
0
00
10
0
000
11
110
001
0000
111
1100
0001
1101
1

Symbol	f(x)	Codeword	length
x_4	0.26	01	2
x_8	0.25	10	2
x_5	0.14	001	3
x_6	0.09	001	3
x_2	0.08	0000	4
x_1	0.07	0001	4
x_7	0.07	1100	4
x_3	0.04	1101	4

111

$$\text{Efficiency} = \frac{H(x)}{L}$$

$$H(x) = \sum_{x=0}^{n} p_x \, log_2 \frac{1}{P_x}$$

$$L = \sum_{x=0}^{n} P_x \times (\text{length of codeword})$$

$$\begin{aligned} H(x) = {} & 0.26 \, log_2 \left(\frac{1}{0.26} \right) + 0.25 \, log_2 \left(\frac{1}{0.25} \right) \\ & + 0.14 \, log_2 \left(\frac{1}{0.14} \right) + 0.09 \, log_2 \left(\frac{1}{0.09} \right) \\ & + 0.08 \, log_2 \left(\frac{1}{0.08} \right) + 0.07 \, log_2 \left(\frac{1}{0.07} \right) \\ & + 0.07 \, log_2 \left(\frac{1}{0.07} \right) + 0.04 \, log_2 \left(\frac{1}{0.04} \right) \end{aligned}$$

$$L = (0.26 \times 2) + (0.25 \times 2) + (0.14 \times 3) + (0.09 \times 3) + (0.08 \times 4) + (0.07 \times 4) + (0.07 \times 4) + (0.04 \times 4)$$
$$= 0.52 + 0.50 + 0.42 + 0.27 + 0.32 + 0.28 + 0.28 + 0.16$$
$$= 2.75 \text{ bits/symbol}$$

$$\text{Efficiency} = \frac{H(x)}{L}$$

2.729/ 2.75

=0.9923

99.23%

Obtain Huffman coding word COMMITEE.

Total number of symbols in the word COMMITTEE is 9.

$$Probability = \frac{Total\ number\ of\ occurrence\ of\ symbol\ in\ message}{Total\ number\ of\ symbols\ in\ the\ message}$$

Probability of a symbol C = p(C)= 1/9

Probability of a symbol O= p(O)=1/9

Probability of a symbol M=p(M)=2/9

Probability of a symbol I=p(I)= 1/9

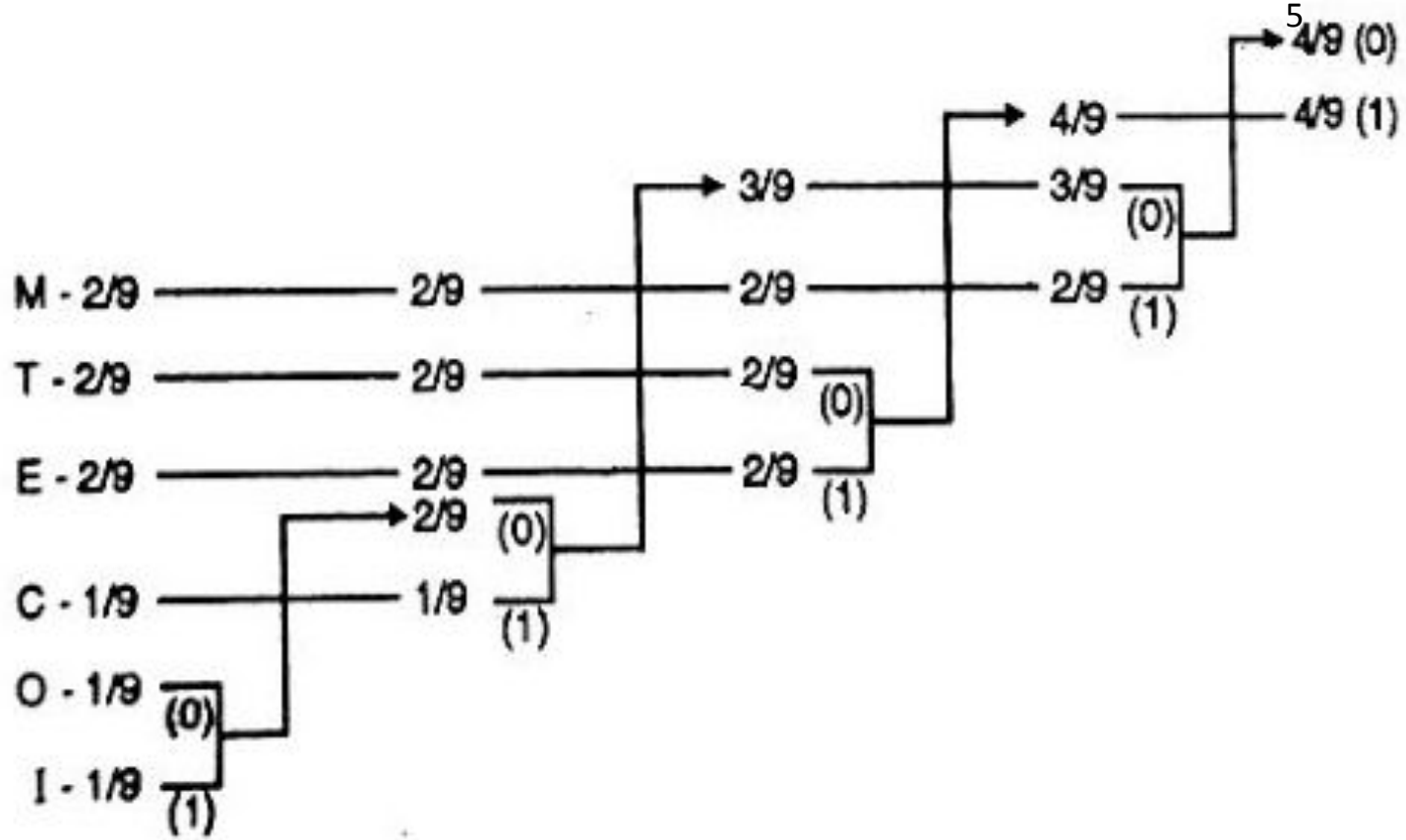
Probability of a symbol T=p(T)= 2/9

Probability of a symbol E=p(E) =2/9

Step1: Arrange the symbols in descending order according to the probability.

Symbol	Probability
M	2/9
T	2/9
E	2/9
C	1/9
O	1/9
I	1/9

Step 2: Construction of Huffman tree



Step 3: Codeword for the Huffman code Tree

Symbol	Probability	Binary Huffman	Method
-	-	Codeword	Word length
M	2/9	01	2
T	2/9	10	2
E	2/9	11	2
C	1/9	001	3
O	1/9	0000	4
I	1/9	0001	4

$\bar{L}$ = average length of the symbol.

Pk= Probability of occurrence of the symbol.

lk=Length of each symbol.

$$\bar{L} = (2/9)x2 + (2/9)x2 + (2/9)x2 + (1/9)x3 + (1/9)x4 + (1/9)x4$$

Solving this, we get the average length as

$\bar{L}$ = 2.5535 bits/symbol

Step 5: Determination of the entropy (H(s))

$$H(s) = -(1/0.3010)(\tfrac{2}{9})log10(\tfrac{2}{9}) + (\tfrac{2}{9})log10(\tfrac{2}{9}) + (\tfrac{2}{9})log10(\tfrac{2}{9}) + (\tfrac{1}{9})log10(\tfrac{1}{9}) + (\tfrac{1}{9})log10(\tfrac{1}{9}) + (\tfrac{1}{9})log10(\tfrac{1}{9})$$

Simplifying the value, the entropy is obtained as H(S)= 2.5034 bits/symbol.

Step 6: Determination of efficiency

$$\eta = (\tfrac{2.5034}{2.5553}) = 0.9797 = 97.97$$

Example 4

A source is transmitting SIX messages with probabilities 0.30, 0.25, 0.15, 0.12, 0.10 and 0.08 respectively

i) Find the binary Huffman code?

ii) Determine its average word length, efficiency

Huffman coding:

X_i	$P(X_i)$	CODE									
X_1	0.30	(00)	0.30	(00)	0.30	(00)	0.43	(1)	0.57	(0)	
X_2	0.25	(10)	0.25	(10)	0.27	(01)	0.30	(00)	0.43	(1)	
X_3	0.15	(010)	0.18	(11)	0.25	(10)	0.27	(01)			
X_4	0.12	(011)	0.15	010	0.18	(11)					
X_5	0.10	(110)	0.12	011							
X_6	0.08	(111)									

$$\text{Entropy } H(X) = \sum_{i=1}^{6} P_i \log_2 \frac{1}{P_i}$$

$$= -[0.30\log_2(0.30) + 0.25\log_2(0.25) + 0.15\log_2(0.15) + 0.12\log_2(0.12) + 0.10\log_2(0.10) + 0.08\log_2(0.08)]$$

$$= 2.42 \text{ bits/messages}$$

$$\text{Average length } (\bar{L}) = \sum_{i=1}^{6} P_i x_i$$

$$= 0.3 \times 2 + 0.25 \times 2 + 0.15 \times 3 + 0.12 \times 3 + 0.10 \times 3 + 0.08 \times 3. = 2.45$$

$$\therefore \text{Efficiency } (\eta) = \frac{H(X)}{\bar{L}} \times 100 = \frac{2.42}{2.45} \times 100 = 98.7\%$$

Try it yourself

A message source generates eight symbols m_1, m_2, ...m_8 with probabilities 0.25, 0.03, 0.19, 0.16, 0.11, 0.14, 0.08 and 0.04 respectively. Give the Huffman codes for these Symbols, Calculate the entropy of the source and the average number of bits per symbol.

- Huffman coding is a technique used to compress files for transmission
- Uses statistical coding
  - more frequently used symbols have shorter code words
- Works well for text and fax transmissions
- An application that uses several data structures
- Drawback is both encoder and decoder need to use the same encoding table, cannot be dynamically created as in Dictionary (if tree changes)

Adaptive Huffman coding

- Huffman coding suffers from the fact that the uncompressor need have some knowledge of the probabilities of the symbols in the compressed files
- this can need more bit to encode the file
- if this information is unavailable compressing the file requires two passes
    - **first pass:** find the frequency of each symbol and construct the huffman tree
    - **second pass:** compress the file
- Adaptive Huffman coding (also called Dynamic Huffman coding) is an adaptive coding technique based on Huffman coding.
- It permits building the code as the symbols are being transmitted, **having no initial knowledge of source distribution**, that allows one-pass encoding and adaptation to changing conditions in data.

Lossy Compression

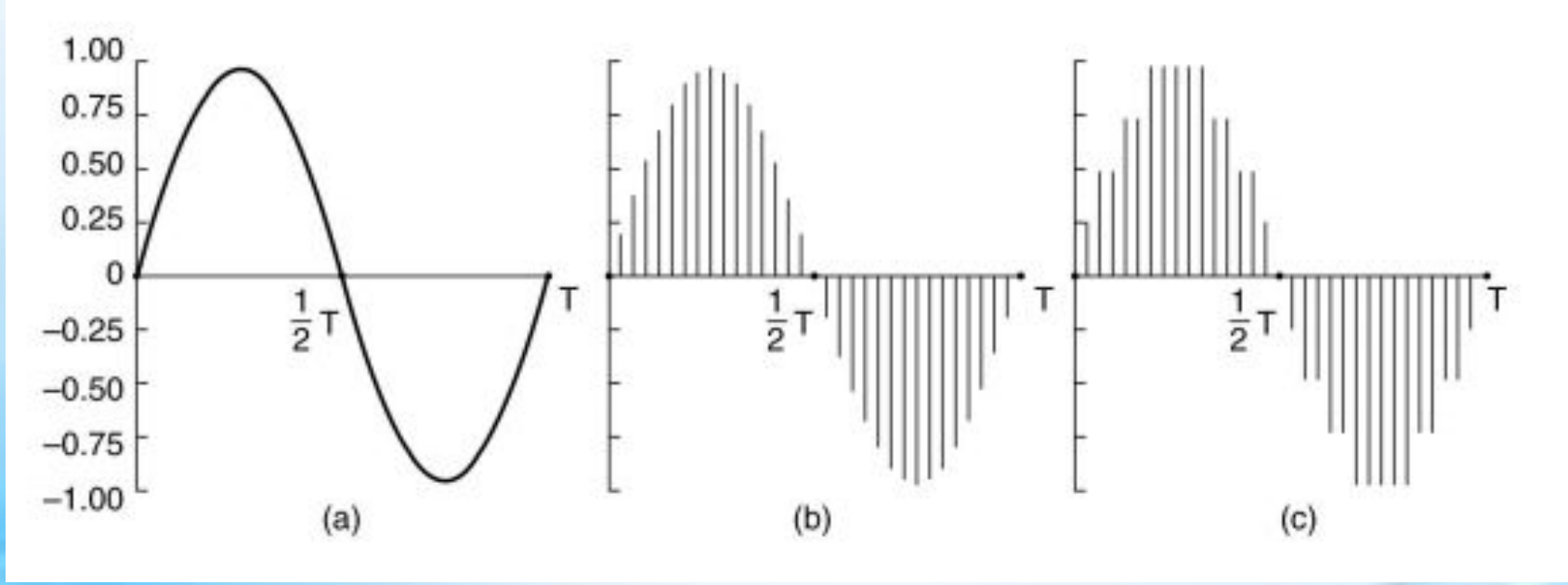
- Lossless has limits on compression
- Sacrifice for accuracy for compression
- Not with text, but audio, video, images
- Approaches
  - ✔ Predictive Coding
    - An analog signal is digitized
    - eg: Pulse code modulation, Delta Modulation, Adaptive DM (ADM)
  - ✔ Transform Coding
    - Mathematical transformation is applied to input signal to produce transformed output, that is recoverable

Speech Compression

Digital Audio

- An audio (sound) wave is a one-dimensional acoustic (pressure) wave.
- When an acoustic wave enters the ear, the eardrum vibrates, causing the tiny bones of the inner ear to vibrate along with it, sending nerve pulses to the brain — These pulses are perceived as sound by the listener
- In a similar way, when an acoustic wave strikes a **microphone**, the microphone generates an electrical signal, representing the sound **amplitude as a function of time**
  - This signal is called an analog audio signal.
  - This signal can be digitized by an ADC (Analog to Digital Converter) to produce a digital signal
  - Digitized sound can be easily processed by PCs in software

Audio Sampling and Quantization

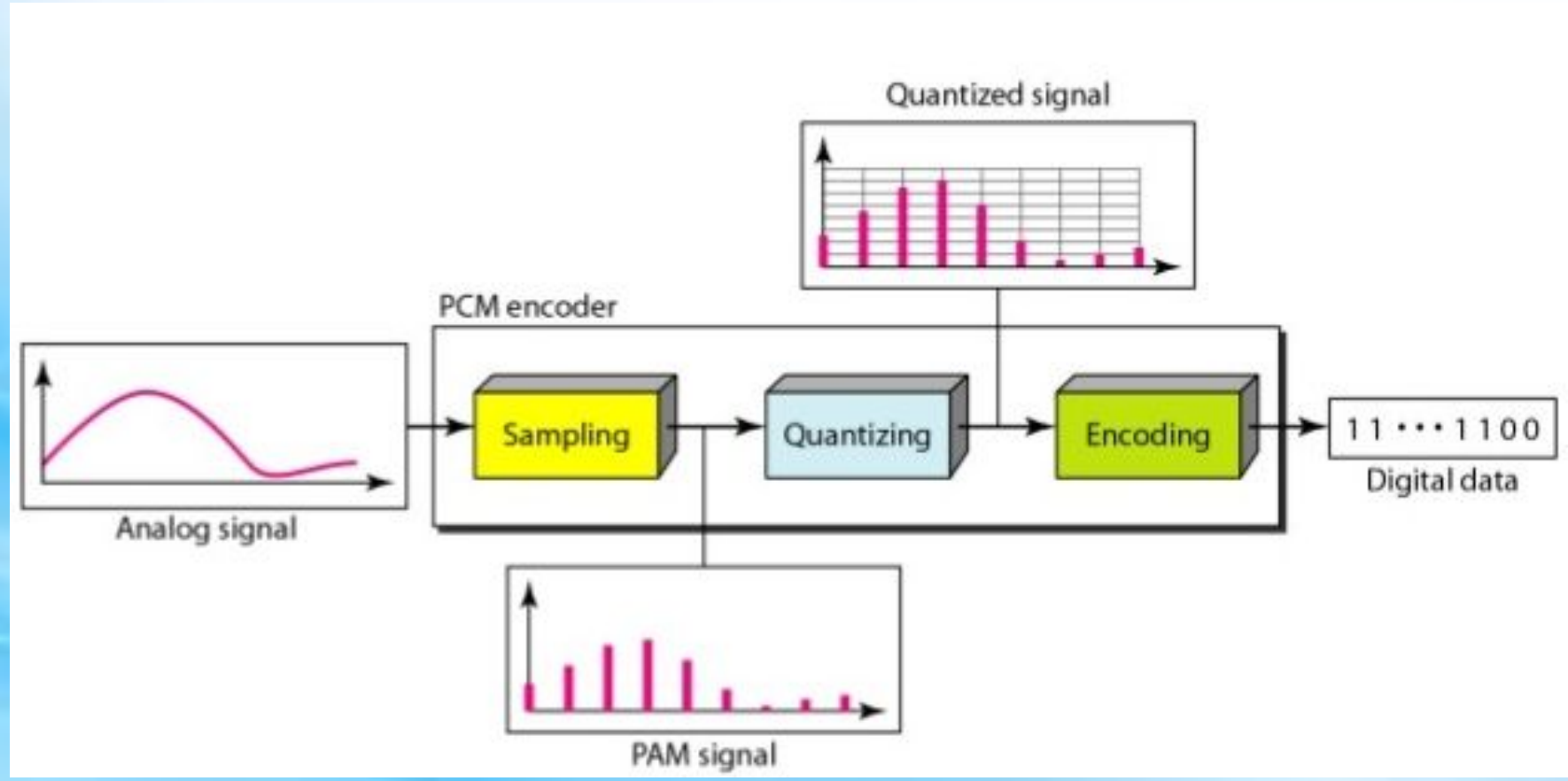


a) An original sine wave to be sampled.

b) The sine wave is sampled

c) The samples are never exact. They are quantized or rounded to one of a finite number of values, determined by the number of bits desired to represent each sample.

Pulse Code Modulation - Lossy Technique



- What do compression mean ?
  - Compression is the **reduction** in size of data in order to save space or **transmission time.**
  - Compression can be used to: Reduce File Size **Save disk space** Reduce transmission time
  - Compression is performed by a program that uses an algorithm to determine how to compress or decompress data.
- Humans can generally sense sounds at frequencies between **20 and 20,000 Hz**
- Speech Compression is the process of lessening the dynamic range between the loudest and quietest parts of an audio signal.

- Digital audio in recording applications is stored on audio-specific technologies including

  eg: Compact Disc, Digital Audio tape (DAT),Speech compression for Telephony (CCITT), Wide band Speech Signal (CCITT G. 722) Digital Compact Cassette, MiniDisc
- Digital audio may be stored in a standard audio file formats and stored on

  eg: Hard disk recorder, Blu-ray or DVD-Audio.
- Depending on the quality of the audio/ speech need to be retained different techniques are used

Image Compression

B)Image

- Photograph, fax page, frame in moving picture
- Digital Image : digitization
  - Representing image as 2D array of pixels
    - bit depth – BW – 1/0
      - grayscale – 8 bits (256)
      - colour – 24bit (24bits – RGB – 8 bit each)
        - alpha channel for background

a. Using a black and white image with a bit depth of 1, we need.

$$\text{Transmission time} = (1280 \times 720 \times 1) / 100{,}000 \approx 9 \text{ seconds}$$

b. Using a gray image with a bit depth of 8, we need.

$$\text{Transmission time} = (1280 \times 720 \times 8) / 100{,}000 \approx 74 \text{ seconds}$$

c. Using a color image with a bit depth of 24, we need.

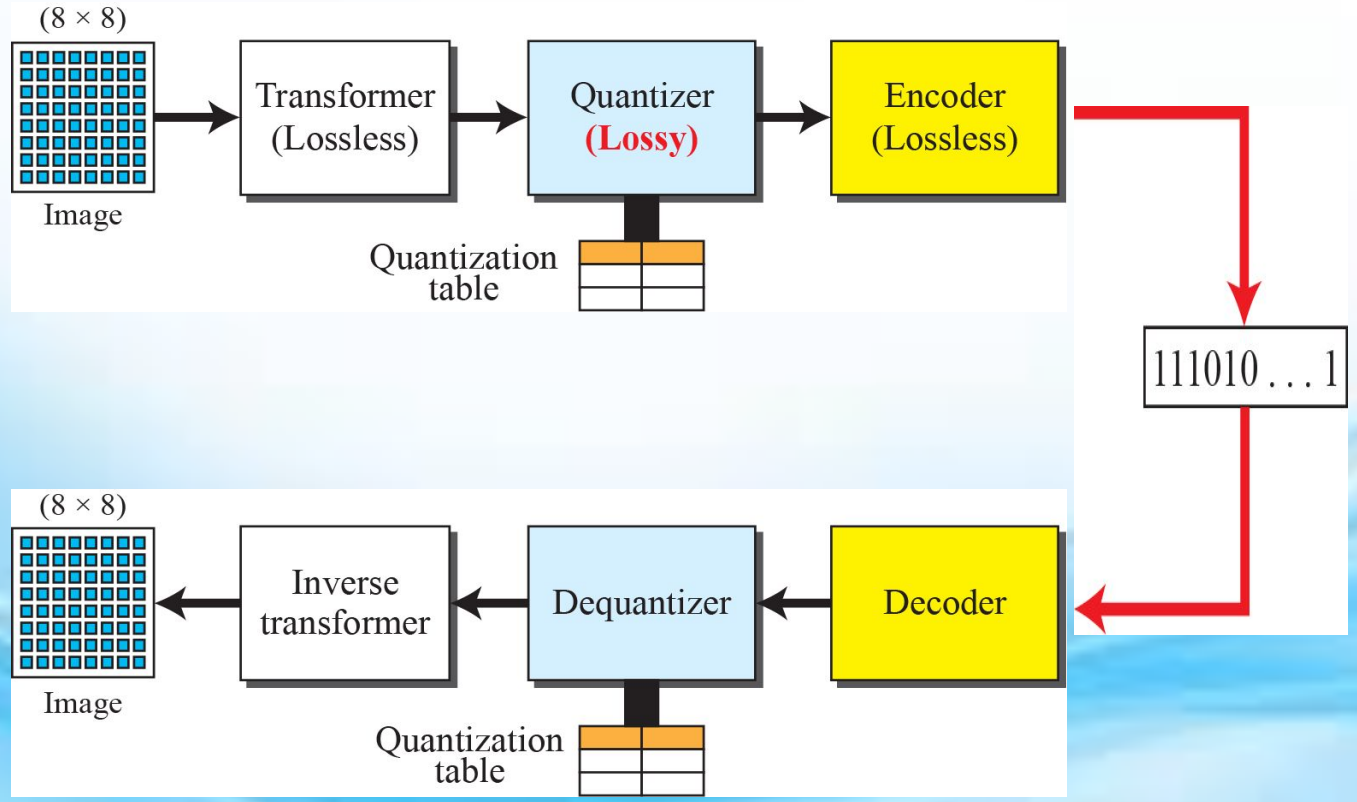
$$\text{Transmission time} = (1280 \times 700 \times 24) / 100{,}000 \approx 215 \text{ seconds}$$

The following shows the time required to transmit an image of 1280 × 720 pixels using the transmission rate of 100 kbps.

- Compression required

Image Compression: JPEG

- Joint Photographic Experts Group
- Lossy compression
- Used for both grayscale and colour images
- Grayscale Compression



- Same method for 3 channel in coloured image

- JPEG is designed to discard information the human eye cannot easily see. The eye barely notices slight changes in colour but will pick out slight changes in brightness or contrast.

 Transformation

  - 8X8, macroblocks - > DCT (convert Spatial Domain to Frequency Domain)

 Quantization

  - Matrix of real no - > rounding of , as real no require lots of bits for encoding
  - Lossy Compression
  - discard high-frequency (noise-like) detail as humans are less critical to the loss of information

$$\begin{bmatrix} 313 & 56 & -27 & 18 & 78 & -60 & 27 & -27 \\ -38 & -27 & 13 & 44 & 32 & -1 & -24 & -10 \\ -20 & -17 & 10 & 33 & 21 & -6 & -16 & -9 \\ -10 & -8 & 9 & 17 & 9 & -10 & -13 & 1 \\ -6 & 1 & 6 & 4 & -3 & -7 & -5 & 5 \\ 2 & 3 & 0 & -3 & -7 & -4 & 0 & 3 \\ 4 & 4 & -1 & -2 & -9 & 0 & 2 & 4 \\ 3 & 1 & 0 & -4 & -2 & -1 & 3 & 1 \end{bmatrix} \div \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} = \begin{bmatrix} 20 & 5 & -3 & 1 & 3 & -2 & 1 & 0 \\ -3 & -2 & 1 & 2 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

8 x 8 DCT Terms

Quantization table (Matrix)

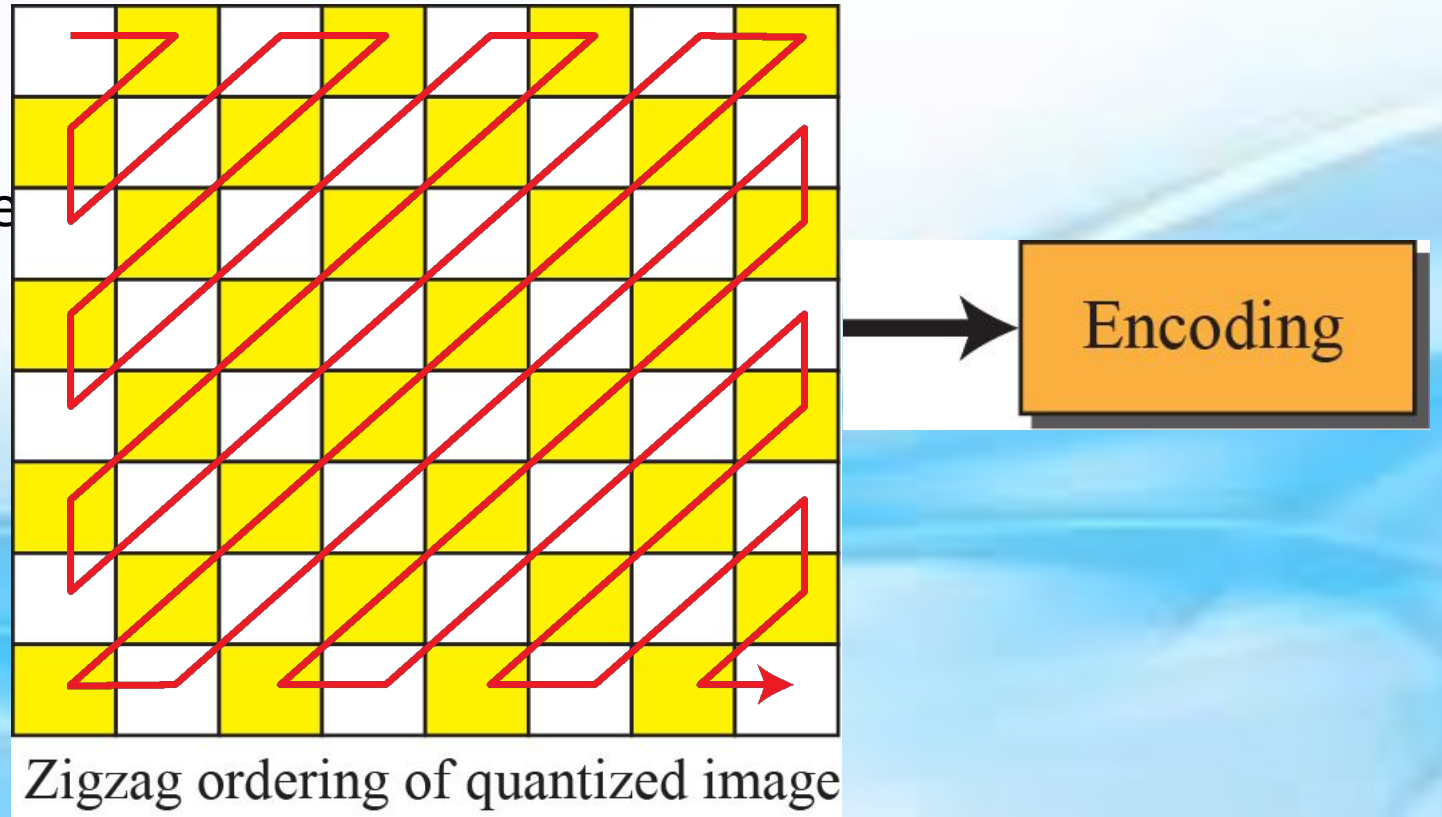
Result

- Each DCT term is divided by the corresponding position in the Quantisation table and then ***rounded to the nearest integer***
- In each table the ***low frequency terms*** are in the top left hand corner and the ***high frequency terms*** are in the bottom right hand corner
- Higher numbers mean lower quality image !
- Lower numbers mean higher quality image !

- Quantization phase introduces max compression - lossy
- Not completely reversible

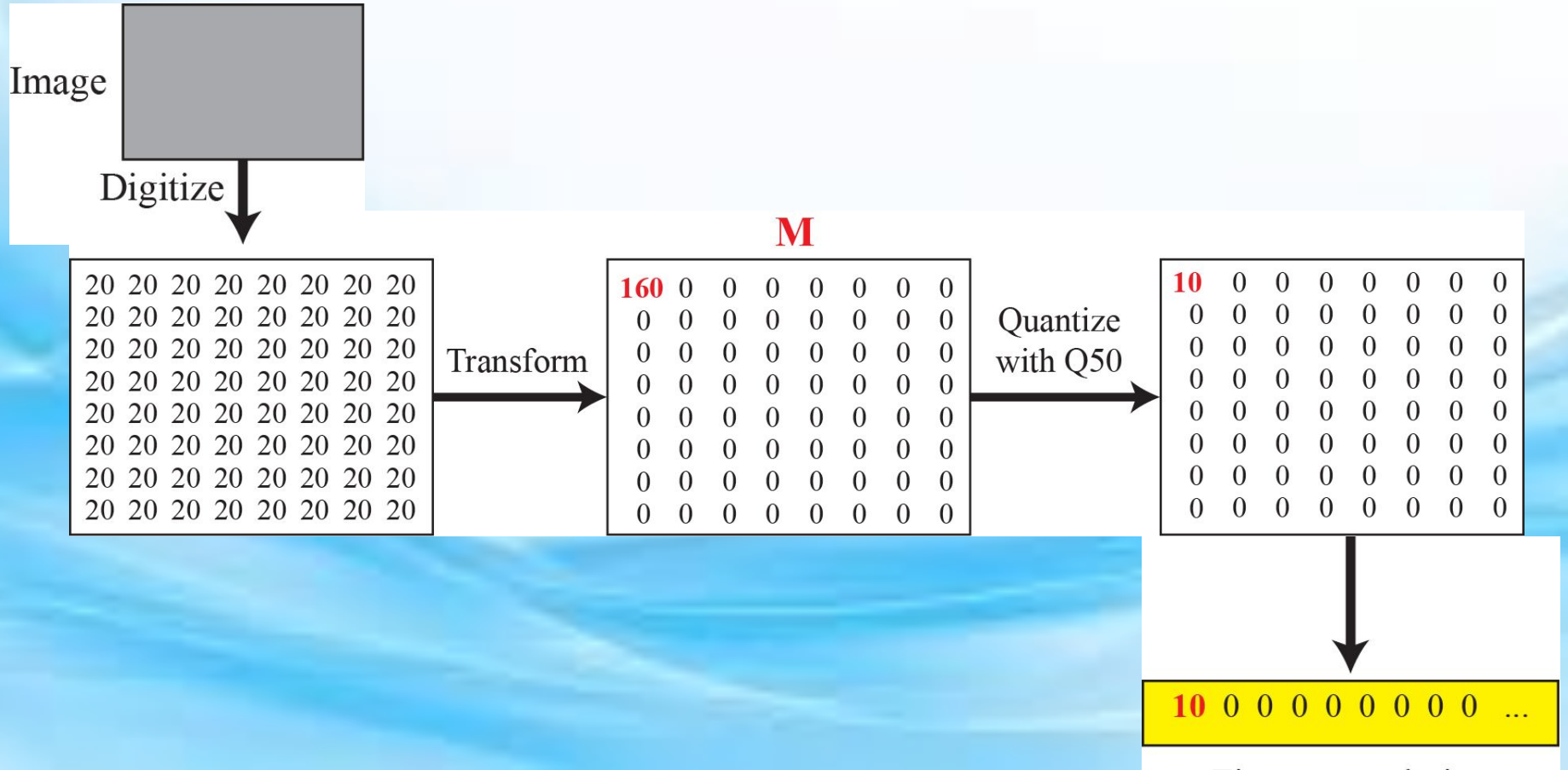
 Encoding

- Low frequency feed in before high frequency
- Encoding is lossless with RLE or arithematic coding or Huffman



Example: gray scale

To show the idea of JPEG compression, we use a block of gray image in which the bit depth for each pixel is 20



Example : *gradient gray scale*

A block that changes gradually; there is no sharp change between the values of neighboring pixels. We still get a lot of zero values

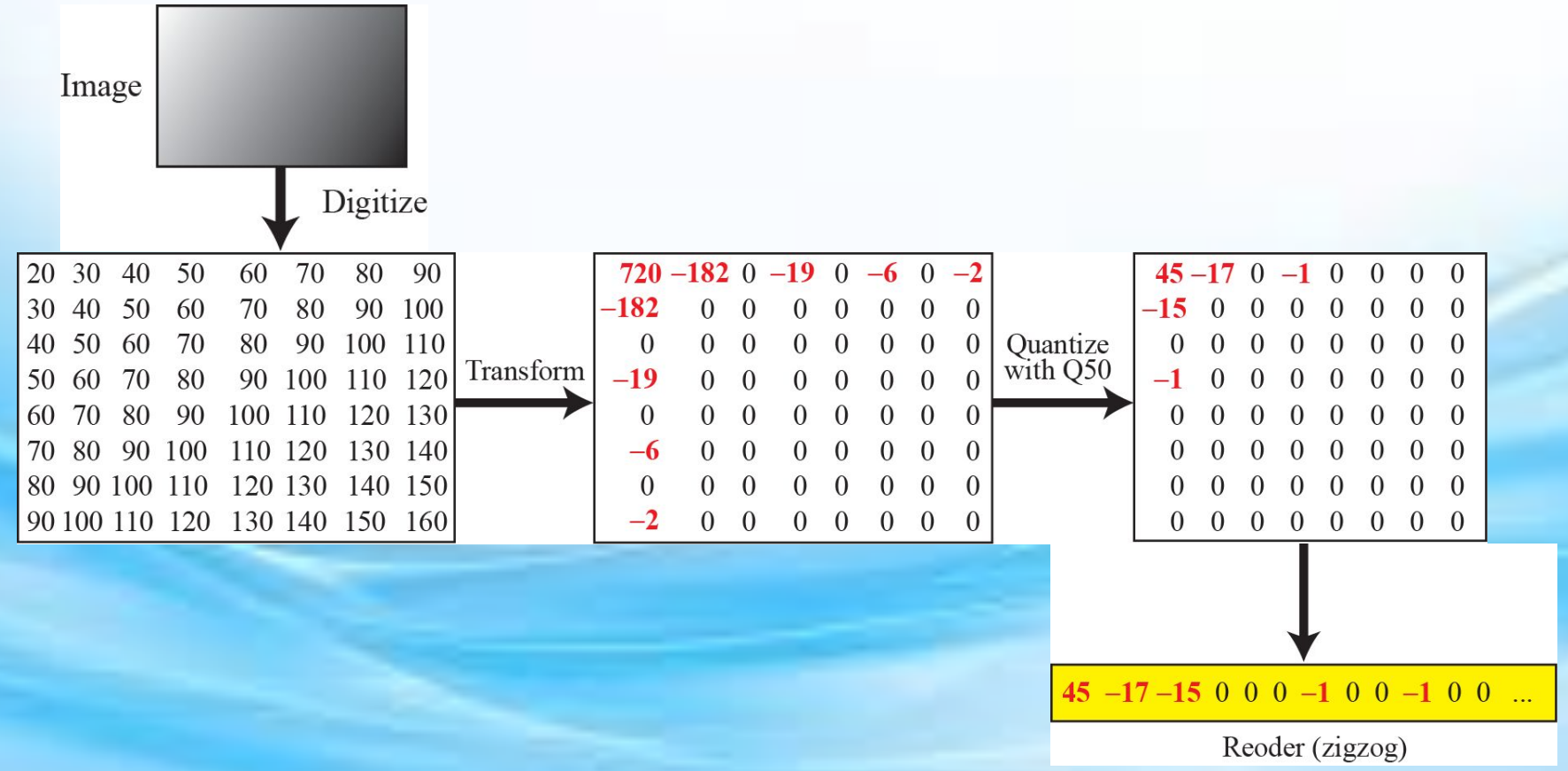


Image Compression :GIF

- Uses a palette of up to 256 colours
  - JPEG :- (2^{24})16 million colours, no transparency, single image
  - GIF :- 256 colours, transparency, multiple images
  - GIF colour palette is smaller
- not good for photographs, but great for text/diagrams
  - Uses **LZW** compression of the bitmap data
- Supports multiple images (animation!!)
- 1989 version supports transparency and interlacing
- At most one colour in the palette may be declared transparent

- GIF’s sharper that JPEG
- Lossless compression is achieved using fewer colours