Written by
Sadman Shaharier Mahim
30 September, 2023

CSE 4105
Computing for Engineers

A brief introduction to the wide theoretical aspects of Computer and IT.

Dedicated to

My mother,
Who continues to care for me each day,
Just as she did during my school years.

Written by

Sadman Shaharier Mahim

IUT CSE 22

On

30 September, 2023

Redistributed on

Saturday, 19 October, 2024

This edition has not yet been verified or proofread.

Preface

CSE 4105 can initially seem overwhelming and difficult to navigate. However, the essence of this course goes far beyond just learning how to code or program. It's designed to give freshmen a broader perspective on what the field of Computer Science truly encompasses. While programming is a core component, CS offers much more. Fields such as cybersecurity, networking, deep learning, system design, and hardware-software integration all fall within its scope. This course serves as an introduction to the vast opportunities within Computer Science. It is intended to be a way to have a look at the big picture, a way to dive in and see what CSE hold in store for you and what you want to be in this wide and diverse field.

DISCLAIMER

To my eager and passionate juniors, I'm simply sharing what I created for myself and my friends while studying this course. I must clarify that this work is still incomplete and hasn't been thoroughly verified by Prof. Kamrul Hasan. While it's a work in progress, this version may contain some inaccuracies or mistakes. I've done my best to ensure its precision, but there may still be inconsistencies. Then again the things taught by sir to you and to us might differ. Therefore, I must stress that I cannot be held liable for any academic harm resulting from any errors. If you encounter any issues or have questions, feel free to reach out to me through This Link.

Contents

- This is the course outline for CSE 4105(2023)
- Here in the following pages the above topics are discussed with each chapters or "Course Outline" being written as CO#: Lorem Ipsum.
- Besides the topics of the course outline, extra topics discussed in class have also been noted. Missing topic from your syllabus as far as I know:

    QR code, Printers.

- Asterisks ** and *** mean emphasis on topics during our time (2023)
- References are given in footnotes.
- The page numbers in the index are interactive and can be used to jump.

CO1 & 2: Algorithms and concepts

Topic 1: Algorithms**

To solve a specific problem in a logical, step-by-step way is called an algorithm The word algorithm came from the Muslim mathematician Musa Al Khwarizmi.

- According to its formal definition, an algorithm is a finite set of instructions carried out in a specific order to perform a particular task.

Based on the type of programming problem a programmer must write algorithms that divide the problem into smaller sections. So the program execution steps are easy to understand and code. An algorithm is a set of commands that must be followed for a computer to perform calculations or other problem-solving operations. Conventionally an algorithm can have one or more inputs but only one output. In other words algorithms are predefined steps or process to write chunks of code that have a specific function like a sorting algorithm that sorts the data in an array. Knowing such algorithms are necessary to be able to write down the codes to common functions in large projects. Again knowing how to write algorithms is necessary so that one may be able to efficiently write down the do-hows of a new kind of program or function. The necessities of writing algorithms are:

1. Computer Science: Algorithms form the basis of computer programming and are used to solve problems ranging from simple sorting and searching to complex tasks such as artificial intelligence and machine learning.

2. Mathematics: Algorithms are used to solve mathematical problems, such as finding the optimal solution to a system of linear equations or finding the shortest path in a graph.

3. Artificial Intelligence: Algorithms are the foundation of artificial intelligence and machine learning, and are used to develop intelligent systems that can perform tasks such as image recognition, natural language processing, and decision-making.

4. Data Science: Algorithms are used to analyze, process, and extract insights from large amounts of data in fields such as marketing, finance, and healthcare

5. Algorithms are necessary for solving complex problems efficiently and effectively.

6. They help to automate processes and make them more reliable, faster, and easier to perform.

7. Algorithms also enable computers to perform tasks that would be difficult or impossible for humans to do manually.

8. They are used in various fields such as mathematics, computer science, engineering, finance, and many others to optimize processes, analyze data, make predictions, and provide solutions to problems.

.

As one would not follow any written instructions to cook the recipe, but only the standard one. Similarly, not all written instructions for programming are an algorithm. For some instructions to be an algorithm, it must have the following characteristics:

1. and Unambiguous: The algorithm should be unambiguous. Each of its steps should be clear in all aspects and must lead to only one meaning.
2. Well-Defined Inputs: If an algorithm says to take inputs, it should be well-defined inputs. It may or may not take input.
3. Well-Defined Outputs: The algorithm must clearly define what output will be yielded and it should be well-defined as well. It should produce at least 1 output.
4. Feasible: The algorithm must be simple, generic, and practical, such that it can be executed with the available resources. It must not contain some future technology or anything.
5. Language Independent: The Algorithm designed must be language-independent, i.e. it must be just plain instructions that can be implemented in any language, and yet the output will be the same, as expected.
6. Input: An algorithm has zero or more inputs. Each that contains a fundamental operator must accept zero or more inputs.
7. Output: An algorithm produces at least one output. Every instruction that contains a fundamental operator must accept zero or more inputs.
8. Definiteness: All instructions in an algorithm must be unambiguous, precise, and easy to interpret. By referring to any of the instructions in an algorithm one can clearly understand what is to be done. Every fundamental operator in instruction must be defined without any ambiguity.

9. Finiteness: An algorithm must terminate after a finite number of steps in all test cases. Every instruction which contains a fundamental operator must be terminated within a finite amount of time. Infinite loops or recursive functions without base conditions do not possess finiteness.

10. Effectiveness: An algorithm must be developed by using very basic, simple, and feasible operations so that one can trace it out by using just paper and pencil.

Types of Algorithms

1. Brute Force Algorithm: A straightforward approach that exhaustively tries all possible solutions, suitable for small problem instances but may become impractical for larger ones due to its high time complexity.

2. Recursive Algorithm: A method that breaks a problem into smaller, similar subproblems and repeatedly applies itself to solve them until reaching a base case, making it effective for tasks with recursive structures.

3. Encryption Algorithm: Utilized to transform data into a secure, unreadable form using cryptographic techniques, ensuring confidentiality and privacy in digital communications and transactions.

4. Backtracking Algorithm: A trial-and-error technique used to explore potential solutions by undoing choices when they lead to an incorrect outcome, commonly employed in puzzles and optimization problems.

5. Searching Algorithm: Designed to find a specific target within a dataset, enabling efficient retrieval of information from sorted or unsorted collections.

6. Sorting Algorithm: Aimed at arranging elements in a specific order, like numerical or alphabetical, to enhance data organization and retrieval.

7. Hashing Algorithm: Converts data into a fixed-size hash value, enabling rapid data access and retrieval in hash tables, commonly used in databases and password storage.

8. Divide and Conquer Algorithm: Breaks a complex problem into smaller subproblems, solves them independently, and then combines their solutions to address the original problem effectively.

9. Greedy Algorithm: Makes locally optimal choices at each step in the hope of finding a global optimum, useful for optimization problems but may not always lead to the best solution.

10. Dynamic Programming Algorithm: Stores and reuses intermediate results to avoid redundant computations, enhancing the efficiency of solving complex problems.

11. Randomized Algorithm: Utilizes randomness in its steps to achieve a solution, often used in situations where an approximate or probabilistic answer suffices.

Advantages of Algorithms:

- It is easy to understand.
- An algorithm is a step-wise representation of a solution to a given problem.
- In an Algorithm the problem is broken down into smaller pieces or steps hence, it is easier for the programmer to convert it into an actual program.

Disadvantages of Algorithms:

- Writing an algorithm takes a long time so it is time-consuming.
- Understanding complex logic through algorithms can be very difficult.
- Branching and Looping statements are difficult to show in Algorithms(imp).

How to express an Algorithm?

Natural Language:- Here we express the Algorithm in the natural English language. It is too hard to understand the algorithm from it.

Flow Chart:- Here we express the Algorithm by making a graphical/pictorial representation of it. It is easier to understand than Natural Language.

Pseudo Code:- Here we express the Algorithm in the form of annotations and informative text written in plain English which is very much similar to the real code but as it has no syntax like any of the programming languages, it can't be compiled or interpreted by the computer. It is the best way to express an algorithm because it can be understood by even a layman with some school-level knowledge.

How to express an Algorithm?

1. Natural Language:- Here we express the Algorithm in the natural English language. It is too hard to understand the algorithm from it.
2. Flow Chart:- Here we express the Algorithm by making a graphical/pictorial representation of it. It is easier to understand than Natural Language.
3. Pseudo Code:- Here we express the Algorithm in the form of annotations and informative text written in plain English which is very much similar to the

real code but as it has no syntax like any of the programming languages, it can't be compiled or interpreted by the computer. It is the best way to express an algorithm because it can be understood by even a layman with some school-level knowledge.

Example of algorithm in natural language:

Algorithm to add 3 numbers and print their sum:

1. START
2. Declare 3 integer variables num1, num2, and num3.
3. Take the three numbers, to be added, as inputs in variables num1, num2, and num3 respectively.
4. Declare an integer variable sum to store the resultant sum of the 3 numbers.
5. Add the 3 numbers and store the result in the variable sum.
6. Print the value of the variable sum
7. END

THE SECTION BELOW IS FOR FURTHER DISCUSSION ON ALGORITHMS:

(This much may not be necessary to write in the exams - section copied from GeeksforGeeks)

How to analyze an Algorithm?

For a standard algorithm to be good, it must be efficient. Hence the efficiency of an algorithm must be checked and maintained. It can be in two stages:

1. Priori Analysis:

"Priori" means "before". Hence Priori analysis means checking the algorithm before its implementation. In this, the algorithm is checked when it is written in the form of theoretical steps. This Efficiency of an algorithm is measured by assuming that all other factors, for example, processor speed, are constant and have no effect on the implementation. This is done usually by the algorithm designer. This analysis is independent of the type of hardware and language of the compiler. It gives the approximate answers for the complexity of the program.

2. Posterior Analysis:

"Posterior" means "after". Hence Posterior analysis means checking the algorithm after its implementation. In this, the algorithm is checked by implementing it in any programming language and executing it. This analysis helps to get the actual and real analysis report about correctness(for every possible input/s if it shows/ returns correct output or not), space required, time consumed, etc. That is, it is dependent on the language of the compiler and the type of hardware used.

What is Algorithm complexity and how to find it?

An algorithm is defined as complex based on the amount of Space and Time it consumes. Hence the Complexity of an algorithm refers to the measure of the time that it will need to execute and get the expected output, and the Space it will need to store all the data (input, temporary data, and output). Hence these two factors define the efficiency of an algorithm.

The two factors of Algorithm Complexity are:

Time Factor: Time is measured by counting the number of key operations such as comparisons in the sorting algorithm.

Space Factor: Space is measured by counting the maximum memory space required by the algorithm to run/execute.

Therefore the complexity of an algorithm can be divided into two types:

1. Space Complexity: The space complexity of an algorithm refers to the amount of memory required by the algorithm to store the variables and get the result. This can be for inputs, temporary operations, or outputs.

How to calculate Space Complexity?

The space complexity of an algorithm is calculated by determining the following 2 components:

Fixed Part: This refers to the space that is required by the algorithm. For example, input variables, output variables, program size, etc.

Variable Part: This refers to the space that can be different based on the implementation of the algorithm. For example, temporary variables, dynamic memory allocation, recursion stack space, etc.

Therefore Space complexity $S(P)$ of any algorithm P is $S(P) = C + SP(I)$, where C is the fixed part and $S(I)$ is the variable part of the algorithm, which depends on instance characteristic I.

Example: Consider the below algorithm for Linear Search

Step 1: START

Step 2: Get n elements of the array in arr and the number to be searched in x

Step 3: Start from the leftmost element of arr[] and one by one compare x with each element of arr[]

Step 4: If x matches with an element, Print True.

Step 5: If x doesn’t match with any of the elements, Print False.

Step 6: END

Here, There are 2 variables arr[], and x, where the arr[] is the variable part of n elements and x is the fixed part. Hence S(P) = 1+n. So, the space complexity depends on n(number of elements). Now, space depends on data types of given variables and constant types and it will be multiplied accordingly.

2. Time Complexity: The time complexity of an algorithm refers to the amount of time required by the algorithm to execute and get the result. This can be for normal operations, conditional if-else statements, loop statements, etc.

How to Calculate, Time Complexity?

The time complexity of an algorithm is also calculated by determining the following 2 components:

Constant time part: Any instruction that is executed just once comes in this part. For example, input, output, if-else, switch, arithmetic operations, etc.

Variable Time Part: Any instruction that is executed more than once, say n times, comes in this part. For example, loops, recursion, etc.

Therefore Time complexity of any algorithm P is T(P) = C + TP(I), where C is the constant time part and TP(I) is the variable part of the algorithm, which depends on the instance characteristic I.

Example: In the algorithm of Linear Search above, the time complexity is calculated as follows:

Step 1: –Constant Time

Step 2: — Variable Time (Taking n inputs)

Step 3: –Variable Time (Till the length of the Array (n) or the index of the found element)

Step 4: –Constant Time

Step 5: –Constant Time

Step 6: –Constant Time

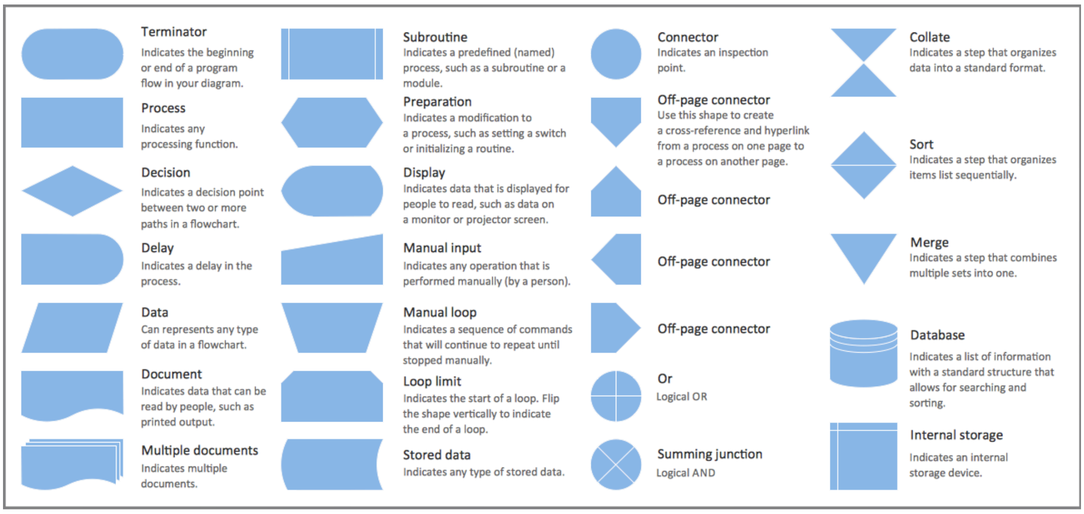
Hence, T(P) = 1 + n + n(1 + 1) + 1 = 2 + 3n, which can be said as T(n).

Topic 2: Flowcharts**

Flowcharts are a variant of algorithms. They are nothing but the graphical representation of the data or the algorithm for a better understanding of the code visually. It displays step-by-step solutions to a problem, algorithm, or process. It is a pictorial way of representing steps that are preferred by most beginner-level programmers to understand algorithms of computer science, thus it contributes to troubleshooting the issues in the algorithm. A flowchart is a picture of boxes that indicates the process flow sequentially. Since a flowchart is a pictorial representation of a process or algorithm, it's easy to interpret and understand the process.

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task.

How ever to draw a flow chart specific shapes are followed. This sets a standardized way of representing flowcharts because if everyone uses different shapes for their flowcharts then it they wouldn't be really intelligible to others working on the same project. The shapes also ensure that the parts of the programme are further easily understood. The most used shape are given below:



The following are the uses of a flowchart:

1. It is a pictorial representation of an algorithm that increases the readability of the program.
2. Complex programs can be drawn in a simple way using a flowchart.
3. It helps team members get an insight into the process and use this knowledge to collect data, detect problems, develop software, etc.
4. A flowchart is a basic step for designing a new process or adding extra features.
5. Communication with other people becomes easy by drawing flowcharts and sharing them.

Flowcharts are mainly used in the below scenarios:

1. It is most importantly used when programmers make projects. As a flowchart is a basic step to make the design of projects pictorially, it is preferred by many.
2. When the flowcharts of a process are drawn, the programmer understands the non-useful parts of the process. So flowcharts are used to separate sound logic from the unwanted parts.
3. Since the rules and procedures of drawing a flowchart are universal, a flowchart serves as a communication channel to the people who are working on the same project for better understanding.
4. Optimizing a process becomes easier with flowcharts. The efficiency of the code is improved with the flowchart drawing.

Three types of flowcharts are listed below:

1. Process flowchart: This type of flowchart shows all the activities that are involved in making a product. It provides a pathway to analyze the product to be built. A process flowchart is most commonly used in process engineering to illustrate the relation between the major as well as minor components present in the product. It is used in business product modeling to help understand employees about the project requirements and gain some insight into the project.
2. Data flowchart: As the name suggests, the data flowchart is used to analyze the data, specifically it helps in analyzing the structural details related to the project. Using this flowchart, one can easily understand the data inflow and outflow from the system. It is most commonly used to manage data or to analyze information to and fro from the system.
3. Business Process Modeling Diagram: Using this flowchart or diagram, one can analytically represent the business process and help simplify the concepts needed to understand business activities and the flow of information. This flowchart illustrates the business process and models graphically which paves the way for process improvement.

Advantages of Flowchart

1. It is the most efficient way of communicating the logic of the system.
2. It acts as a guide for a blueprint during the program design.
3. It also helps in the debugging process.
4. Using flowcharts we can easily analyze the programs.

5. flowcharts are good for documentation.

Disadvantages of Flowchart

1. Flowcharts are challenging to draw for large and complex programs.
2. It does not contain the proper amount of details.
3. Flowcharts are very difficult to reproduce.
4. Flowcharts are very difficult to modify.

For more details on individual segments go to
https://www.geeksforgeeks.org/what-is-a-flowchart-and-its-types/

Examples:

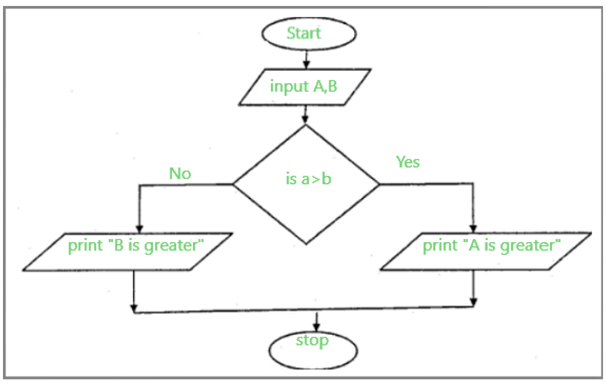
Question 1. Draw a flowchart to find the greatest number among the 2 numbers.

Solution:

Algorithm:

1. Start
2. Input 2 variables from user
3. Now check the condition If a > b, goto step 4, else goto step 5.
4. Print a is greater, goto step 6
5. Print b is greater
6. Stop

FlowChart:



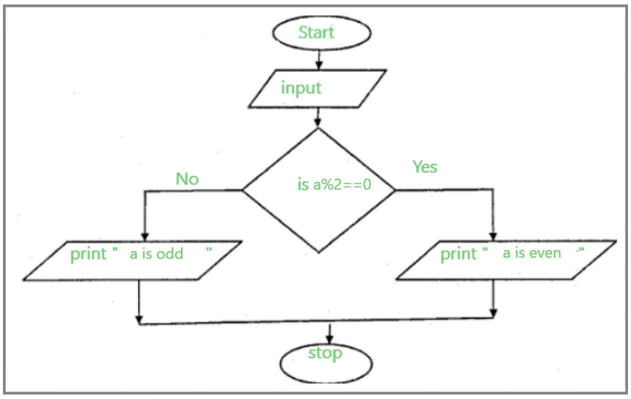
Question 2. Draw a flowchart to check whether the input number is odd or even

Solution:

Algorithm:

1. Start
2. Put input a
3. Now check the condition if a % 2 == 0, goto step 5. Else goto step 4
4. Now print("number is odd") and goto step 6
5. Print("number is even")
6. Stop

FlowChart:



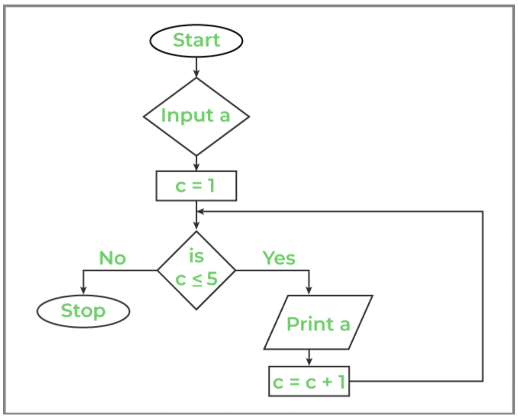
Question 3. Draw a flowchart to print the input number 5 times.

Solution:

Algorithm:

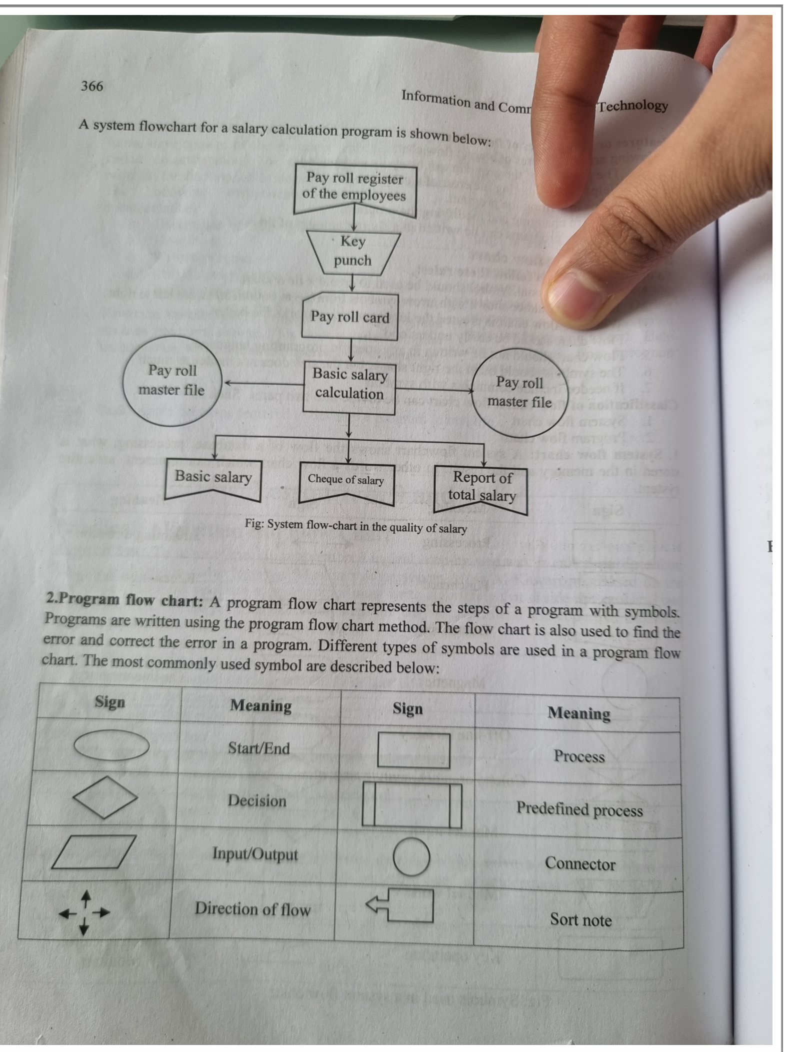
1. Start
2. Input number a
3. Now initialise c = 1
4. Now we check the condition if c <= 5, goto step 5 else, goto step 7.
5. Print a
6. c = c + 1 and goto step 4
7. Stop

FlowChart:



More example are available on the internet.

Snippets from the book:

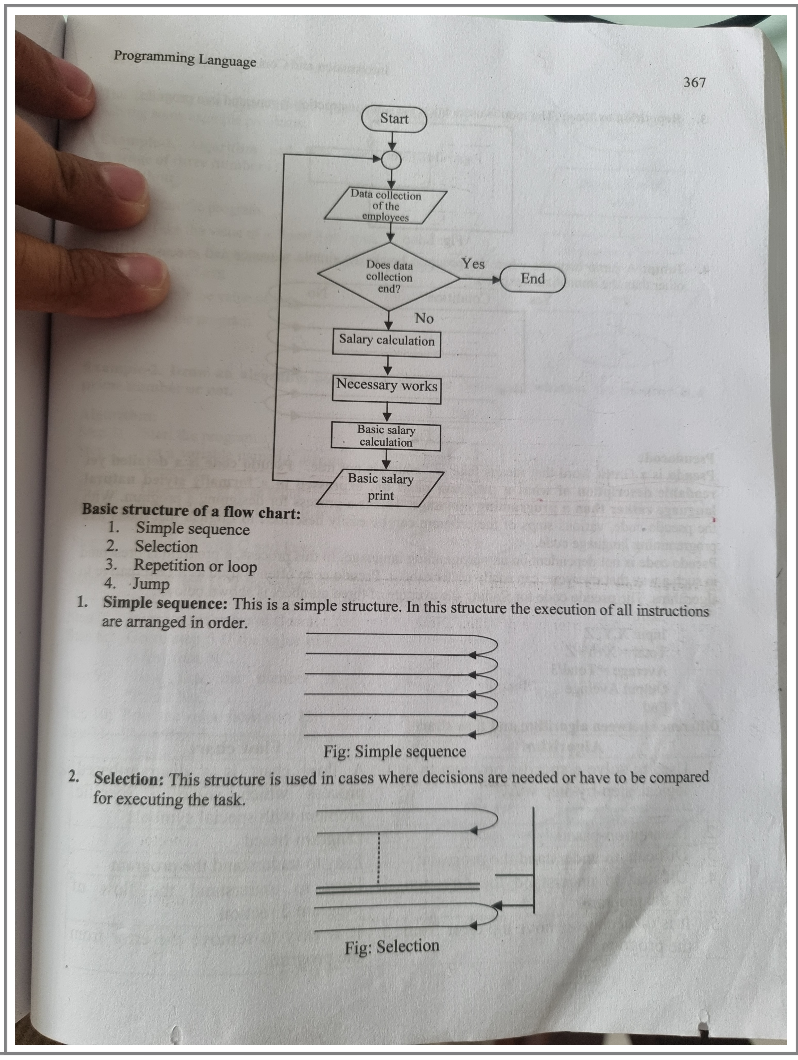


A system flowchart for a salary calculation program is shown below:

Fig: System flow-chart in the quality of salary

2.Program flow chart: A program flow chart represents the steps of a program with symbols. Programs are written using the program flow chart method. The flow chart is also used to find the error and correct the error in a program. Different types of symbols are used in a program flow chart. The most commonly used symbol are described below:

Sign	Meaning	Sign	Meaning
	Start/End		Process
	Decision		Predefined process
	Input/Output		Connector
	Direction of flow		Sort note



Basic structure of a flow chart:

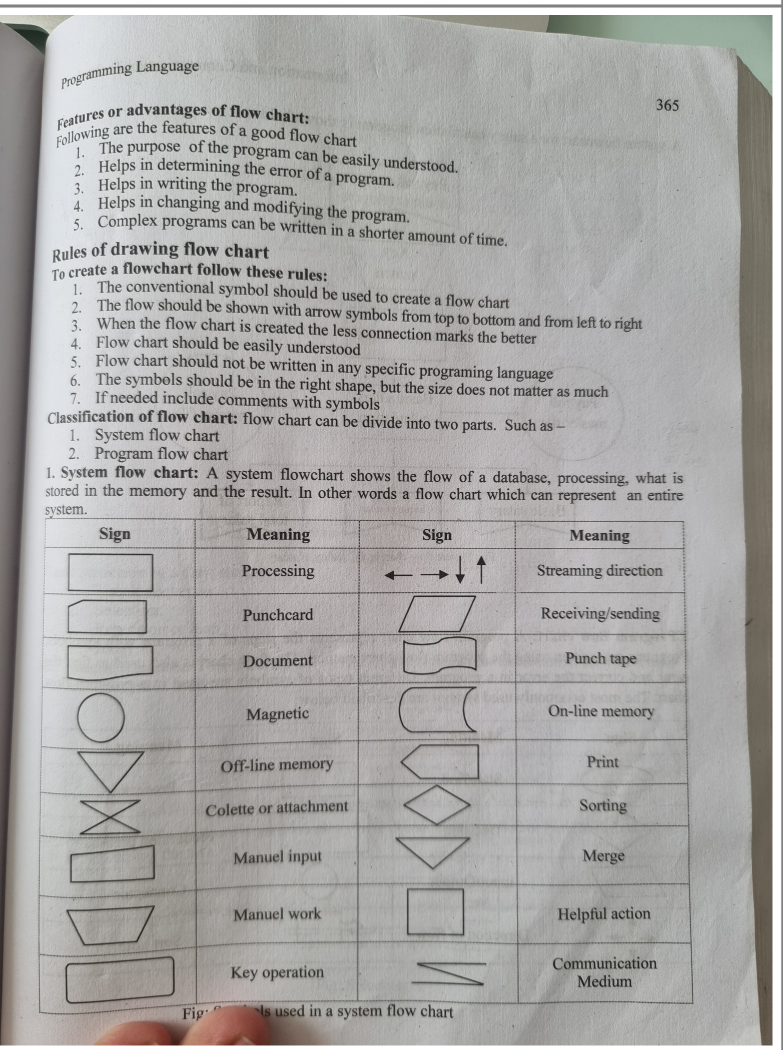
1. Simple sequence
2. Selection
3. Repetition or loop
4. Jump

1. **Simple sequence:** This is a simple structure. In this structure the execution of all instructions are arranged in order.

Fig: Simple sequence

2. **Selection:** This structure is used in cases where decisions are needed or have to be compared for executing the task.

Fig: Selection



Features or advantages of flow chart:

Following are the features of a good flow chart

1. The purpose of the program can be easily understood.
2. Helps in determining the error of a program.
3. Helps in writing the program.
4. Helps in changing and modifying the program.
5. Complex programs can be written in a shorter amount of time.

Rules of drawing flow chart

To create a flowchart follow these rules:

1. The conventional symbol should be used to create a flow chart
2. The flow should be shown with arrow symbols from top to bottom and from left to right
3. When the flow chart is created the less connection marks the better
4. Flow chart should be easily understood
5. Flow chart should not be written in any specific programing language
6. The symbols should be in the right shape, but the size does not matter as much
7. If needed include comments with symbols

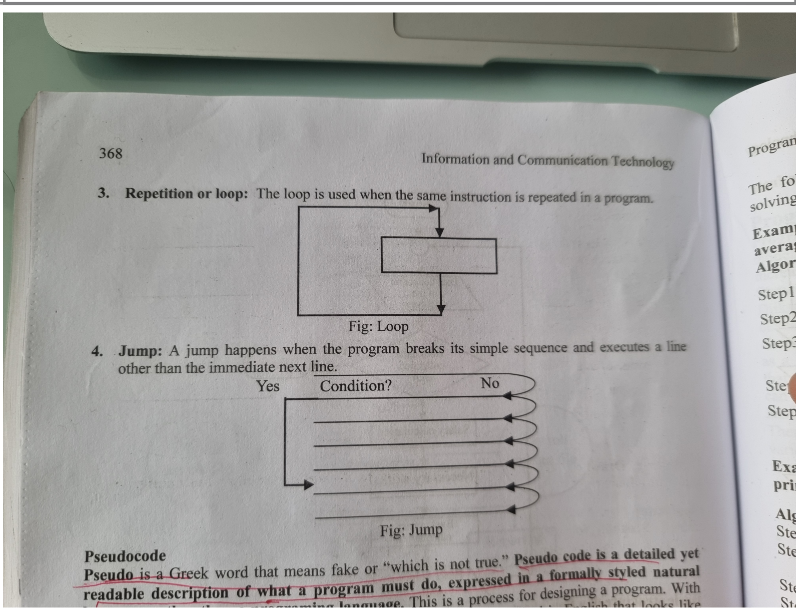
Classification of flow chart: flow chart can be divide into two parts. Such as –

1. System flow chart
2. Program flow chart

1. **System flow chart:** A system flowchart shows the flow of a database, processing, what is stored in the memory and the result. In other words a flow chart which can represent an entire system.

Sign	Meaning	Sign	Meaning
	Processing		Streaming direction
	Punchcard		Receiving/sending
	Document		Punch tape
	Magnetic		On-line memory
	Off-line memory		Print
	Colette or attachment		Sorting
	Manuel input		Merge
	Manuel work		Helpful action
	Key operation		Communication Medium

Fig: ...ls used in a system flow chart



3. **Repetition or loop:** The loop is used when the same instruction is repeated in a program.

Fig: Loop

4. **Jump:** A jump happens when the program breaks its simple sequence and executes a line other than the immediate next line.

Fig: Jump

Pseudocode

Pseudo is a Greek word that means fake or "which is not true." **Pseudo code is a detailed yet readable description of what a program must do, expressed in a formally styled natural** ... **language.** This is a process for designing a program. With

Topic 3: Pseudocode**

Another form of algorithm is Pseudocode. It is actually a method through which one can represent the implementation of an algorithm but doesn't really write the code. That is it is the false representation of a code written using normal language before the actual code is written. Often at times, algorithms are represented with the help of pseudo codes as they can be interpreted by programmers no matter what their programming background or knowledge is. Pseudo code, as the name suggests, is a false code or a representation of code which can be understood by even a layman with some school level programming knowledge. While an algorithm is an organized logical sequence of the actions or the approach towards a particular problem are expressed using natural verbal but somewhat technical annotations, pseudocode are an implementation of an algorithm in the form of annotations and informative text written in plain English. It has no syntax like any of the programming language and thus can't be compiled or interpreted by the computer.

Advantages of Pseudocode

1. Improves the readability of any approach. It's one of the best approaches to start implementation of an algorithm.
2. Acts as a bridge between the program and the algorithm or flowchart. Also works as a rough documentation, so the program of one developer can be understood easily when a pseudo code is written out. In industries, the approach of documentation is essential.And that's where a pseudo-code proves vital.
3. The main goal of a pseudo code is to explain what exactly each line of a program should do, hence making the code construction phase easier for the programmer.

How to write a Pseudo-code?

1. Arrange the sequence of tasks and write the pseudocode accordingly.
2. Start with the statement of a pseudo code which establishes the main goal or the aim. Example:

```
This program will allow the user to check
the number whether it's even or odd.
```

References: "ICT class xi-xii A.Z.M. kamruzzaman, Md Masud Rana, Akkharpatra Publications" simplilearn.com, geeksforgeek.org (/introduction-to-algorithms/)

3. The way the if-else, for, while loops are indented in a program, indent the statements likewise, as it helps to comprehend the decision control and execution mechanism. They also improve the readability to a great extent.

Example:

```
if "1"
    print response
        "I am case 1"

if "2"
    print response
        "I am case 2"
```

4. Use appropriate naming conventions. The human tendency follows the approach to follow what we see. If a programmer goes through a pseudo code, his approach will be the same as per it, so the naming must be simple and distinct.

5. Use appropriate sentence casings, such as CamelCase for methods, upper case for constants and lower case for variables.

6. Elaborate everything which is going to happen in the actual code. Don't make the pseudo code abstract.

7. Use standard programming structures such as 'if-then', 'for', 'while', 'cases' the way we use it in programming.

8. Check whether all the sections of a pseudo code is complete, finite and clear to understand and comprehend.

9. Don't write the pseudo code in a complete programmatic manner. It is necessary to be simple to understand even for a layman or client, hence don't incorporate too many technical terms.

Do's :

- Use control structures
- Use proper naming convention
- Indentation and white spaces are the key
- Keep it simple.
- Keep it concise.

Don'ts :

- Don't make the pseudo code abstract.
- Don't be too generalized.

Some examples of Pseudocode vs Natural algorithm:

<u>Add two numbers:</u>

Pseudocode:

1. BEGIN
2. NUMBER len, area,perimeter
3. INPUT len
4. area = len*len
5. perimeter = len*4
6. OUTPUT area
7. OUTPUT perimeter
8. END

Natural/normal algorithm:

1. The variables “len”, “area”, and “perimeter” are declared.
2. The program prompts the user to input the length of one side of the square and stores the value in “len”.
3. The area of the square is calculated by squaring the length of one side and storing the result in the “area” variable.
4. The perimeter of the square is calculated by multiplying the length of one side by 4 and stored in the “perimeter” variable.
5. The program outputs the value stored in the “area” variable.
6. The program outputs the value stored in the “perimeter” variable.

<u>Calculate the area of a rectangle:</u>

Pseudocode:

1. BEGIN
2. NUMBER b1,b2,area,perimeter
3. INPUT b1
4. UNPUT b2
5. area=b1*b2

6. perimeter=2*(b1+b2)
7. OUTPUT area
8. OUTPUT perimeter
9. END

Natural/Normal algorithm:

1. The variables “b1”, “b2”, “area”, and “perimeter” are declared.
2. The program prompts the user to input the length of the rectangle and stores the value in “b1”.
3. The program prompts the user to input the width of the rectangle and stores the value in “b2”.
4. The area of the rectangle is calculated by multiplying “b1” and “b2” and stored in the “area” variable.
5. The perimeter of the rectangle is calculated by multiplying 2 by the sum of “b1” and “b2” and stored in the “perimeter” variable.
6. The program outputs the value stored in the “area” variable.
7. The program outputs the value stored in the “perimeter” variable.

<u>Solving Quadratic equation</u>

Pseudocode:

1. BEGIN
2. NUMBER a, b, c, d, x1, x2
3. INPUT a,b,c
4. d = b^2-4ac
5. IF (d >= 0) THEN
6.   x1 = (-b+√d)/2a yada x1 = (-b+d^(1/2)/2a
7.   x2 = (-b-√d)/2a yada x2 = (-b-d^(1/2)/2a
8.   OUTPUT "ROOT 1:”+x1
9.   OUTPUT "ROOT 2:”+x2
10. ELSE IF (d == 0) THEN
11.   x1=x2= -b/2a

12. OUTPUT "ROOT 1:"+x1
13. OUTPUT "ROOT 2:"+x2
14. ELSE
15. OUTPUT "There is no real root"
16. ENDIF
17. END

Normal/natural algorithm:

1. The variables “a”, “b”, “c”, “d”, “x1”, and “x2” are declared.
2. The program prompts the user to input the values of the coefficients “a”, “b”, and “c”.
3. The value of “d”, the discriminant, is calculated as the square of “b” minus 4 times “a” times “c”.
4. If “d” is greater than or equal to 0, then the program calculates the two real roots of the equation using the formula (-b±√d) / 2a and stores the values in “x1” and “x2”.
5. If “d” is equal to 0, then the program calculates the one real root of the equation using the formula -b / 2a and stores the value in both “x1” and “x2”.
6. If “d” is less than 0, then the program outputs the message “There is no real root”.
7. The program outputs the message “ROOT 1:” followed by the value stored in “x1” and the message “ROOT 2:” followed by the value stored in “x2”.

The following are algorithm and flow chart versions for solving some example problems:

Example-1. Algorithm and flow chart to find the average of three numbers

Algorithm:

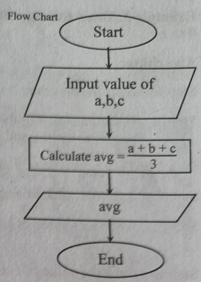
Step1: Start the program.

Step2: Take the value of a, b and c as input.

Step3: avg = (a+b+c)/3 using the formula to calculate the value of avg

Step4: Output the value of avg

Step5: End the program.



Example-2. Draw an algorithm and a flow chart to determine whether the number is a prime number or not.

Algorithm:

Step 1: Start the program.

Step 2: Let a variable named C whose value is 2.

Step 3: Take value of N as input.

Step 4: If the value of N is less than 3 then go to step 9.

Step 5: Divide N with C and keep the remainder in variable X.

Step 6: Go to step 11 if the value of T is 0.

Step7: Increase the value of C to 1.

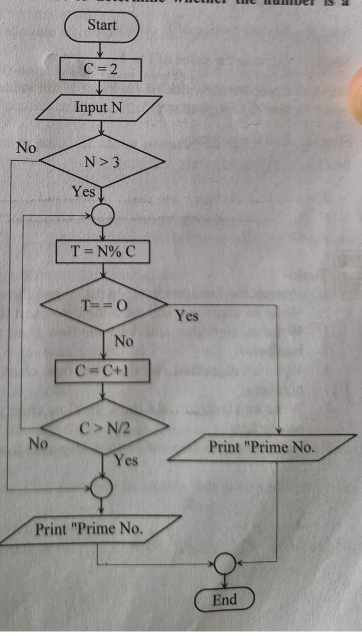
Step 8: Go to step 5 If the value of C is less than N/2.

Step 9: Show that the number is "Prime No."

Step 10: Print the value from step 12.

Step 11: Show that the number is not a "Prime No."

Step 12: End the program.



Example-3: Draw the algorithm and flow chart to determine all the Fibonacci numbers from 0 to n.

Algorithm:

Step 1: Start the program.

Step 2: Create three variables a, b, I whose values are sequentially 0, 1 and 1.

Step 3: Take the value of n as input.

Step 4: Take the value of a and b as results.

Step 5: Add the values of a and b and keep the result in variable F.

Step 6: show the value of F as result.

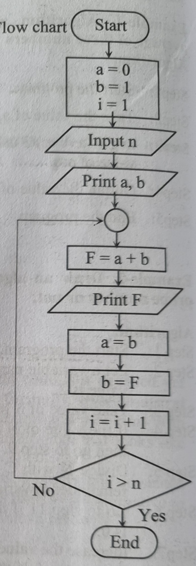
Step 7: Keep the value of b in variable a.

Step 8: Keep the value of F variable b.

Step9: Increase the value of I to 1

Step 10: Compare the value of I with n. If the value is less than n then go to step 5.

Step 11: End the program.



Task:

1. Present the importance of algorithm and flow-chart before program creation.
2. Write an algorithm and draw the flow chart to determine the area of a rectangle.
3. Write an algorithm and draw the flow chart to determine the smallest number from three numbers.
4. Write an algorithm and draw the flow chart to determine least common multiple of two numbers.
5. Write an algorithm and draw the flow chart to convert the temperature fro[illegible]eit to centigrade.
6. Determine which is and is not a leap year from among the yea[illegible]91[illegible].

Fig: Jump

Pseudocode

Pseudo is a Greek word that means fake or "which is not true." **Pseudo code is a detailed yet readable description of what a program must do, expressed in a formally styled natural language rather than a programing language.** This is a process for designing a program. With the pseudo code, various steps of the program can be easily described in English that looks like programming language code.

Pseudo code is not dependent on any programing language. In this process, a program is presented in such a way that everyone can easily understand it. Pseudo code often is used as an alternative to algorithms. The pseudo code for finding the average of three numbers is shown below:

```
Start
Input X,Y, Z
Total = X+Y+Z
Average = Total/3
Output Average
End
```

Difference between algorithm and flow chart

Algorithm	Flow chart
1. Used to solve a specific problem in a logical, step-by-step way.	1. A flow chart is a diagram-based process which solve a specific problem with special symbols
2. Description-based.	2. Diagram-based
3. Difficult to understand the program.	3. Easy to understand the program
4. Difficult to understand the direction of the program.	4. Easy to understand the flow of program direction
5. It is difficult to remove the error from the program.	5. It is easy to remove the error from the program

Snippets from the book:

CO 3: How computer works

Topic 1: Booting***

Starting a computer or a computer-embedded device is called boot sequence or booting. It's the process that starts when we turn on the computer (using the power button or by a software command) and ends when the operating system is loaded into the memory. Booting takes place in two steps –

- Switching on power supply
- Loading operating system into computer's main memory
- Keeping all applications in a state of readiness in case needed by the user

The first program or set of instructions that run when the computer is switched on is called BIOS or Basic Input Output System. BIOS is a firmware, i.e. a piece of software permanently programmed into the hardware.

If a system is already running but needs to be restarted, it is called rebooting. Rebooting may be required if a software or hardware has been installed or system is unusually slow.

There are two types of booting –

- Cold Booting – When the system is started by switching on the power supply it is called cold booting. The next step in cold booting is loading of BIOS.
- Warm Booting – When the system is already running and needs to be restarted or rebooted, it is called warm booting. Warm booting is faster than cold booting because BIOS is not reloaded.

Here is the boot sequence:

References: https://www.tutorialspoint.com/basics_of_computers/basics_of_computers_quick_guide , https://www.baeldung.com/cs/computer-boot-process , https://www.knowledgehut.com/blog/web-development/booting-in-operating-system

1. **BIOS:** When we turn on the computer, there is no program inside the computer's main memory (RAM), so the CPU looks for another program, called the BIOS (Basic Input/Output System), and runs it. The BIOS is a firmware that is located on the motherboard and is run by the CPU to start the booting sequence:
2. **Running POST:** After the BIOS starts running, it starts a process called POST (Power-On Self-Test) which tests all the hardware devices and makes sure there are no issues. Moreover, if POST finds some issues in the hardware, the booting process stops and the computer fails to boot. The 4 main hardware components that are checked in post are: (i)CPU, (ii) DRAM, (iii)GPU, (iv)Boot drive.
3. **Loading MBR to RAM:** After running POST, the BIOS proceeds to load the MBR (Master Boot Record) from the bootable device into RAM. The MBR consists of 512 or more bytes located at the very beginning sector of the bootable device (which can be an HDD, an SSD, or a flash drive). Below is a simplified structure of the MBR:

Bootstrap code	Partition table	Boot signature
446 bytes	64 bytes	2 bytes

4. **Running the Bootloader:** After loading the MBR into RAM, the BIOS runs the first instruction loaded from the MBR. The first instruction is typically the bootstrap code, aka the bootloader, which is a program written in machine code that loads the operating system into RAM. Each operating system has its own bootloaders. For example, GNU GRUB, LILO (Linux Loader), and rEFInd are a few popular Linux bootloaders.
5. **Running the OS:** Once the OS is loaded into the memory, the OS starts running. Further, the OS starts its own initialization(mentioned in step 6, 7 & 8 which includes loading device drivers, setting up libraries, etc).
6. **System Configuration:** Device drivers are put into the memory after the OS is loaded to ensure the proper operation of our gadgets.
7. **Loading System Utilities:** In this step, system utilities like antivirus and volume control are loaded into the memory.
8. **User Authentication:** Finally, when the OS initialization is finished, the OS starts a shell that displays a login prompt to the user. The system will prompt the user to input their credentials if any user authentication is configured. Once the system has received valid credentials, it will typically launch the GUI shell or the CLI shell.

Dual Booting of Operating System

Dual booting of operating system is installation of two operating systems on a single computer, known as dual-booting. Numerous operating systems can be installed on such a machine. To choose which operating system to boot, a boot loader familiar with various file systems and operating systems can occupy the boot space.

Once it has been loaded, one of the operating systems on the disc can be booted. On the disc, there may be several partitions, each of which houses a different operating system; when a computer system boots in the operating system, a boot manager program displays a menu so the user can choose the operating system to use.

Topic 2: Computing Paradigms

(**This was not shown in the class, so the importance is very low**)

Computing paradigms refer to fundamental approaches or models for organizing and performing computations and solving problems in the field of computer science. It is a way of thinking about how to use computers to model and simulate the real world, and to design and implement software systems. These paradigms represent different ways of thinking about and structuring the processes involved in computing. Some common computing paradigms include:

1. Imperative Programming Paradigm: In imperative programming, the focus is on describing a sequence of statements that change a program's state. This paradigm includes languages like C, C++, and Java, where you explicitly define how the program should execute step by step.

2. Functional Programming Paradigm: Functional programming is centered around the concept of functions as first-class citizens. It emphasizes immutability and the avoidance of mutable state. Languages like Haskell, Lisp, and JavaScript (to some extent) follow this paradigm.

3. Object-Oriented Programming Paradigm: Object-oriented programming (OOP) organizes code around objects, which represent real-world entities and encapsulate

both data and methods (functions). Languages like Python, Java, and C# are examples of OOP languages.

4. Logic Programming Paradigm: Logic programming is based on formal logic and focuses on describing relationships and constraints. Prolog is a well-known language that follows this paradigm, where you declare facts and rules to infer new information.

5. Parallel and Distributed Computing Paradigm: This paradigm deals with solving problems by breaking them down into smaller tasks that can be executed concurrently or distributed across multiple processors or computers. Technologies like MPI (Message Passing Interface) and frameworks like Hadoop and Spark fall into this category.

6. Event-Driven Programming Paradigm: Event-driven programming is common in GUI applications and systems that respond to external events. It involves defining event handlers that react to specific events or triggers. JavaScript and many graphical user interface (GUI) frameworks use this paradigm.

7. Quantum Computing Paradigm: Quantum computing is an emerging paradigm that leverages the principles of quantum mechanics to perform computations. It has the potential to solve certain problems exponentially faster than classical computers. Quantum computing languages like Q# and Cirq are being developed.

8. Cognitive Computing Paradigm: Cognitive computing involves creating computer systems that can simulate human thought processes, such as learning, reasoning, and problem-solving. It often incorporates artificial intelligence and machine learning techniques.

9. Biological Computing Paradigm: This paradigm draws inspiration from biology and seeks to build computational systems that mimic biological processes. It includes topics like DNA computing and neural networks inspired by the human brain.

References: https://www.researchgate.net/publication/300887204_Computing_Paradigms by Xiaohong Qiu, Jianjun Zhao, Guoliang Wang, and Xiaoxin Chen, Computational Paradigm: https://mathworld.wolfram.com/ComputationalParadigm.html by Stephen Wolfram, https://www.geeksforgeeks.org/different-computing-paradigms/

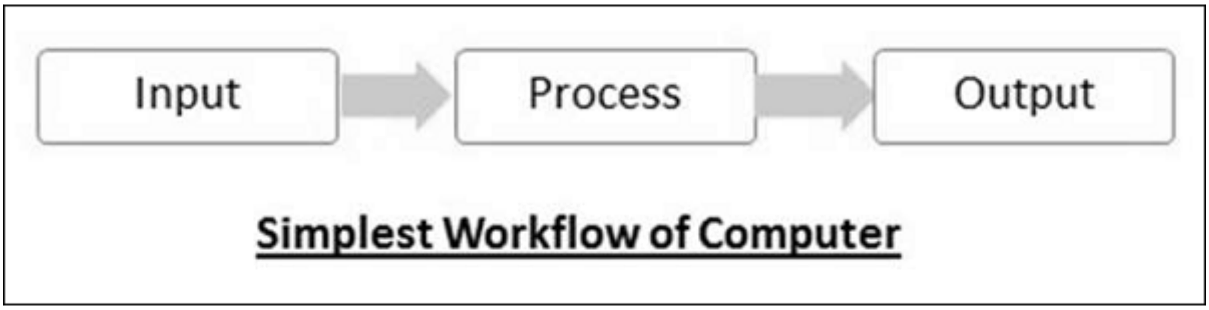
10. Natural Language Processing (NLP) Paradigm: NLP is a specialized paradigm focused on processing and understanding human language. It involves techniques like text analysis, sentiment analysis, and language generation.

These paradigms are not mutually exclusive, and many modern programming languages and systems incorporate elements from multiple paradigms. The choice of paradigm depends on the problem being solved and the programming language or technology best suited for that task. Additionally, new paradigms may continue to emerge as technology evolves and new computational models are developed.

Topic 3: Input-Process-Output Model*

(**This was minutely discussed in the class- note copied from sir's notes**)

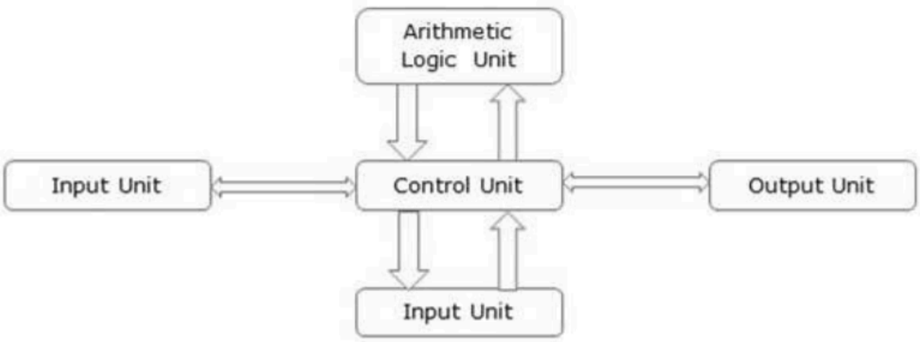
Computer input is called data and the output obtained after processing it, based on user's instructions is called information. Raw facts and figures which can be processed using arithmetic and logical operations to obtain information are called data.



The processes that can be applied to data are of two types −

1. Arithmetic operations − Examples include calculations like addition, subtraction, differentials, square root, etc.
2. Logical operations − Examples include comparison operations like greater than, less than, equal to, opposite, etc.

The corresponding figure for an actual computer looks something like this –



The basic parts of a computer are as follows –

1. Input Unit – Devices like keyboard and mouse that are used to input data and instructions to the computer are called input unit.
2. Output Unit – Devices like printer and visual display unit that are used to provide information to the user in desired format are called output unit.

3. Control Unit – As the name suggests, this unit controls all the functions of the computer. All devices or parts of computer interact through the control unit.

4. Arithmetic Logic Unit – This is the brain of the computer where all arithmetic operations and logical operations take place.

5. Memory – All input data, instructions and data interim to the processes are stored in the memory. Memory is of two types – primary memory and secondary memory. Primary memory resides within the CPU whereas secondary memory is external to it.

Control unit, arithmetic logic unit and memory are together called the central processing unit or CPU. Computer devices like keyboard, mouse, printer, etc. that we can see and touch are the hardware components of a computer. The set of instructions or programs that make the computer function using these hardware parts are called software. We cannot see or touch software. Both hardware and software are necessary for working of a computer.

Topic 4: Basics of Computers - Classification

(**Very slight discussions in the class like server, mainframe etc - copied form sir's note, very low importance**)

Historically computers were classified according to processor types because development in processor and processing speeds were the developmental benchmarks. Earliest computers used vacuum tubes for processing, were huge and broke down frequently. However, as vacuum tubes were replaced by transistors and then chips, their size decreased and processing speeds increased manifold.

All modern computers and computing devices use microprocessors whose speeds and storage capacities are skyrocketing day by day. The developmental benchmark for computers is now their size. Computers are now classified on the basis of their use or size –

- Desktop
- Laptop
- Tablet
- Server
- Mainframe
- Supercomputer

Let us look at all these types of computers in detail.

Desktop

Desktop computers are personal computers (PCs) designed for use by an individual at a fixed location. IBM was the first computer to introduce and popularize use of desktops. A desktop unit typically has a CPU (Central Processing Unit), monitor, keyboard and mouse. Introduction of desktops popularized use of computers among common people as it was compact and affordable.

Riding on the wave of desktop's popularity many software and hardware devices were developed specially for the home or office user. The foremost design consideration here was user friendliness.

Laptop

Despite its huge popularity, desktops gave way to a more compact and portable personal computer called laptop in 2000s. Laptops are also called notebook computers or simply notebooks. Laptops run using batteries and connect to networks using Wi-Fi (Wireless Fidelity) chips. They also have chips for energy efficiency so that they can conserve power whenever possible and have a longer life.

Modern laptops have enough processing power and storage capacity to be used for all office work, website designing, software development and even audio/video editing.

Tablet

After laptops computers were further miniaturized to develop machines that have processing power of a desktop but are small enough to be held in one's palm. Tablets have touch sensitive screen of typically 5 to 10 inches where one finger is used to touch icons and invoke applications.

Keyboard is also displayed virtually whenever required and used with touch strokes. Applications that run on tablets are called apps. They use operating systems by Microsoft (Windows 8 and later versions) or Google (Android). Apple computers have developed their own tablet called iPad which uses a proprietary OS called iOS.

Server

Servers are computers with high processing speeds that provide one or more services to other systems on the network. They may or may not have screens attached to them. A group of computers or digital devices connected together to share resources is called a network.

Servers have high processing powers and can handle multiple requests simultaneously. Most commonly found servers on networks include –

- File or storage server
- Game server
- Application server
- Database server
- Mail server
- Print server

Mainframe

Mainframes are computers used by organizations like banks, airlines and railways to handle millions and trillions of online transactions per second. Important features of mainframes are –

- Big in size
- Hundreds times Faster than servers, typically hundred megabytes per second
- Very expensive
- Use proprietary OS provided by the manufacturers
- In-built hardware, software and firmware security features

Supercomputer

Supercomputers are the fastest computers on Earth. They are used for carrying out complex, fast and time intensive calculations for scientific and engineering applications. Supercomputer speed or performance is measured in teraflops, i.e. 1012 floating point operations per second. Summit (as of Oct 1st 23') is currently the most powerful supercomputer in the world. It is located at the Oak Ridge National Laboratory in Tennessee, USA. The machine was developed by IBM and NVIDIA and has a processing power of 200 petaflops. It is used for research in various fields, including physics, energy, and healthcare.

Topic 5: CPU & Microprocessor**

A central processing unit (CPU) is often regarded as the "brain" of a computer, responsible for executing instructions that govern the operation of a computer system. At the heart of the CPU lies a microprocessor, a complex integrated circuit that interprets and performs these instructions. This essay explores the essential components of a CPU, the microprocessor's role, and the instruction set architecture that governs their operation.

The CPU:

The CPU is a fundamental component of a computer, responsible for executing instructions, performing calculations, and managing data. It consists of several key elements, including:

1. Control Unit (CU): The control unit orchestrates the operation of the CPU by decoding and executing instructions. It manages the flow of data and instructions within the CPU and coordinates interactions with memory and other peripherals.

2. Arithmetic Logic Unit (ALU): The ALU is responsible for performing arithmetic and logical operations, such as addition, subtraction, multiplication, and comparison. It manipulates data based on the instructions received.

3. Registers: Registers are small, high-speed storage locations within the CPU used for temporary data storage and manipulation. Common registers include the program counter (PC), instruction register (IR), and accumulator.

4. Memory Interface: The memory interface allows the CPU to read and write data to and from main memory (RAM) and other storage devices.

The Microprocessor: “The Heart of the CPU”

At the core of the CPU is the microprocessor, a silicon chip that interprets and executes instructions. Microprocessors come in various architectures and designs, but they all follow the same fundamental principles:

1. Instruction Fetch: The microprocessor fetches instructions from memory using the program counter (PC) to determine the memory location of the next instruction.

2. Instruction Decode: Once fetched, the microprocessor decodes the instruction to understand its purpose and identify the operation code (opcode) and operands.

3. Execute: The microprocessor performs the specified operation based on the opcode and operands. This could involve arithmetic, logic, data transfer, or control instructions.

4. Write Back: If necessary, the microprocessor writes the results of the operation back to registers or memory.

Instruction Set Architecture (ISA): The Language of the CPU

The instruction set architecture (ISA) defines the set of instructions that a microprocessor can understand and execute. It serves as the interface between software and hardware. ISAs can vary widely between different microprocessors, but they typically include the following components:

1. Opcode: The opcode is a binary code that represents a specific operation or instruction. It tells the microprocessor what action to perform, such as addition, subtraction, or loading data into a register.

2. Operands: Operands are data values or memory addresses used by instructions. Some instructions have explicit operands, while others use implicit operands from registers or memory.

3. Addressing Modes: Addressing modes define how operands are specified within an instruction. Common addressing modes include immediate (the operand value is part of the instruction), direct (the operand is a memory address), and register (the operand is in a CPU register).

In summary, the CPU and its microprocessor are at the core of every computer system, responsible for executing instructions that drive the entire computing process. The microprocessor fetches, decodes, and executes instructions based on the instruction set architecture (ISA), which defines the CPU's language. Understanding these essential components helps demystify the inner workings of computers and underscores their critical role in modern technology.

Certainly, let's delve a bit deeper into how a processor performs a basic operation like addition, including the binary representation and instruction execution.

Binary Representation of Addition in microprocessor

In a computer's memory, data is represented in binary format. Each binary digit is called a "bit," and combinations of bits represent different numbers. For example, in the binary system:

- 0 represents the binary digit "0."
- 1 represents the binary digit "1."

To perform addition, the CPU needs to understand how to add binary numbers. Consider adding two binary numbers, A and B:

```
 A: 1101
+ B: 1010
____
```

Here's how the processor goes about this:

1. Binary Addition: The processor starts by adding the rightmost bits of A and B. In this case, 1 + 0 = 1.

```
  A: 1101
+ B: 1010
---------
 1
```

2. Carry: When performing binary addition, if the sum of two bits exceeds 1, there is a carry value of 1 to be added to the next column on the left. In this case, 1 + 0 + 1 (carry) = 0 with a carry of 1

```
  A: 1101
+ B: 1010
---------
   10
```

3. Continue Addition: The processor moves to the next column, adding the bits and any carry value from the previous step. Here, 1 + 1 (carry) + 0 = 10.

```
  A: 1101
+ B: 1010
---------
  101
```

4. Carry Propagation: The process continues, propagating any carry values leftward. In the next step, 1 + 1 + 1 (carry) = 11.

```
  A: 1101
+ B: 1010
---------
 1101
```

5. Final Result: The processor completes the addition, and the result is 1101, which is the binary representation of 13 in decimal.

Machine Code Instruction for Addition

Now, let's discuss how the processor receives the instruction to perform addition:

- In the machine code (binary) instruction, the opcode for addition is specified. This opcode tells the CPU that it needs to perform an addition operation.

- The instruction also includes information about where the operands (A and B in this case) are located. These operands can be registers, memory addresses, or immediate values, depending on the architecture and addressing modes of the CPU.

For example, a simplified machine code instruction for addition might look like this:

Opcode	Operand1	Operand2	Destination
1101	0010	1011	0101

In this instruction:

- Opcode 1101 signifies addition.
- Operand1 (0010) and Operand2 (1011) represent binary values to be added.
- Destination (0101) specifies where the result should be stored, such as in a specific register.

When the processor fetches and decodes this instruction, it understands that it needs to perform an addition operation with Operand1 and Operand2 and store the result in the specified destination.

This is a simplified representation, and real machine code instructions are more complex, including details like addressing modes, register identifiers, and more.

In summary, a processor performs addition by executing machine code instructions that specify the operation to be performed (in this case, addition) and the location of the operands. The processor then follows a series of binary addition steps, as shown earlier, to compute the result.

Topic 6: Programming Lang, Compiler and Compilation Chain***

Language Processor

As discussed earlier, an important function of system software is to convert all user instructions into machine understandable language. When we talk of human machine interactions, languages are of three types −

1. Machine-level language − This language is nothing but a string of 0s and 1s that the machines can understand. It is completely machine dependent.
2. Assembly-level language − This language introduces a layer of abstraction by denying mnemonics. Mnemonics are English like words or symbols used to denote a long string of 0s and 1s. For example, the word "READ" can be defined to mean that computer has to retrieve data from the memory. The complete instruction will also tell the memory address. Assembly level language is machine dependent.
3. High level language − This language uses English like statements and is completely independent of machines. Programs written using high level languages are easy to create, read and understand.

Program written in high level programming languages like Java, C++, etc. is called source code. Set of instructions in machine readable form is called object code or machine code. System software that converts source code to object code is called language processor. There are three types of language interpreters−

1. Assembler − Converts assembly level program into machine level program.
2. Interpreter − Converts high level programs into machine level program line by line.
3. Compiler − Converts high level programs into machine level programs at one go rather than line by line.

Machine Language

This language only Os and Is are used. That's why computers can directly understand any instruction given in this language. This allows us to communicate with the computer directly. In machine language bits, bytes and memory addresses are used, such as:

1. Arithmetic namely addition, subtraction, multiplication, division
2. Control namely load, store, jump
3. Input-output namely read and write
4. Direct use namely start, halt and end

Advantage

1. The main advantage of machine language is that it can directly communicate with computers

2. There is no need to convert machine language. That's why it works fast.

3. Machine language does not need large memory. 4. To understand the inner structure of a computer this language is needed

Disadvantage

1. No program is generally understood in machine language.

2. It is hard to write in machine language because of Os and 1s.

3. Writing programs in this language take a lot of time and more likely to be wrong. It is difficultto find and correct errors. 4. The biggest disadvantage of this language is that written programs for a computer cannot be used on other computers. Machine language is also called lower level language.

Assembly language

Assembly language is also called symbolic language. It started usage in 1950. This language was widely used in second generation computers. In the assembly language, the instruction and data address is given with a symbol and not a binary or hexadecimal number. This symbol is called symbolic code or nemonic code. It is easily understandable.

For example:

Addition is written as ADD

Subtraction is written as SUB

Multiplication is written as MUL

Division is written as DIV

Every instruction has four parts: label, op-code, operand and comment

Label Op-code Operand Comment

Label: The label contains the instruction for the symbolic address. During jump, the address of the next instruction is given, but the label may not always be there. There are one or two alphanumeric letters in a label, but there is no gap between the letter. Nemonic and resistor name cannot be used as labels.

Op-code: The instruction is nemonic. These nemonics may be different on different computers as mentioned below.

Nemonic instruction	Pronunciation and full name	Explanation
LDA	Load Accumulator	The main memory is instructed to keep the number of specific positions in the accumulator
ADD	ADD	Instruction for addition of two operands
CLR	CLEAR	Instruction for emptying accumulator
STA	Store Accumulator	Instruction to store data in accumulator
SUB	Subtract	Instruction to subtract of two operands
MUL	Multiply	Instruction to multiply of two operands
DIV	Divide	Instruction to divide two operands
OR	OR	Instruction for logical OR operation of two operands
JMU	JUMP	Instruction to go to a specific position
INP	INPUT	Receive data or instruction and keep it in a specific location of memory
OUT	OUTPUT	Instruction for sending output to a specific issue of memory
STP	STOP	Instruction to stop the program

Operand: The location on which the op-code works is called an operand. Generally alphanumeric letters are used to indicate the address of the operand's position. Examples include A, B, X, Y, AM, XY, etc.

Comments: The comment is not part of the instruction, but explains the instruction so that in the future the programmer or anyone can understand the meaning of the program easily. The program is used for its own convenience. After the operand field, the comment can be written with a colon (:) or semicolon (;).

Example: Add A and B and put the result into C. The address of A or B's position is also called A or B. An assembly language program is given below by showing instructions to add A and B and placing the result in C position:

CLR	Empty the accumulator.
INP: A	Put A in main memory from input A
INP: B	Put B in main memory from input B
LDA: A	Put A in accumulator
ADD: B	Add B with accumulator number and put result in the accumulator
STA : C	Put accumulator number in C position
OUT : C	Show Result in C variable
STP	Stop

The program written in assembly language cannot directly understand the computer. For this reason, this kind of program has to convert into machine language. A special program is used for this conversion An assembler is a program that helps to convert assembly language into machine language.

Advantages of assembly language

1. Program written is assembly language is easier than machine language
2. The program takes less time to write
3. It's easy to change the program Disadvantages of assembly language

Disadvantages of assembly language

1. It’s harder to find error
2. Need converter program
3. Programmer needs to know about machine to write the program
4. Different machines use different assembly languages
5. It is a machine-dependent language

High level language

High level language is similar to human language. The program written in high level language can be used in different types of machines. This language is beyond the control of computers, hence this language is called is high level language. It is easy to understand for people but computers cannot understand directly so need to convert it into machine language with the help of a translator program. These include: Qbasic, Pascal, C/C++ and JAVA.

Classifications of High Level Language

Depending on the application high level language is divided into two parts:

• General purpose language

• Special purpose language

The language that is designed to be suitable for all kinds of work is called general purpose language. These are BASIC, PASCAL, C, etc. Those languages that are designed for special work is called special purpose language, such as COBOL, ALGOL, FORTAN, etc.

Uses of high level language

- For making big programs
- Making large programs used for big data processing
- In order to create software for those areas where a lot of memory is needed
- For creating complex mathematical calculation software
- For creating application package software
- Different types of automatic process control work

Advantages

1. Easy to write programs in high level language and takes less time to write
2. Less possibility for error and finding error and correction is easy
3. In order to create programs in this language there is no need to know about the internal structure of a computer
4. Written programs for one type of computer can run on other models of computers

Disadvantages

1. The main problem of high level language is that it cannot communicate with computers directly

2. Programs need to be translated for the computer 3. Needs a huge memory

These three are known as 1st, 2nd and 3rd gen languages. There are also 4th gen and 5th gen.

—Translator programs:

Assembler:

Translates programs written in assembly language into machine language. Assembly language programs are converted from nemonic codes to machine language. If there is an error in the program an error massage is given.

Interpreter :

The major advantage of assembly level language was its ability to optimize memory usage and hardware utilization. However, with technological advancements computers had more memory and better hardware components. So ease of writing programs became more important than optimizing memory and other hardware resources.

In addition, a need was felt to take programming out of a handful of trained scientists and computer programmers, so that computers could be used in more areas. This led to development of high level languages that were easy to understand due to resemblance of commands to English language.

The system software used to translate high level language source code into machine level language object code line by line is called an interpreter. An interpreter takes each line of code and converts it into machine code and stores it into the object le.

The advantage of using an interpreter is that they are very easy to write and they do not require a large memory space. However, there is a major disadvantage in using interpreters, i.e., interpreted programs take a long time in executing. To overcome this disadvantage, especially for large programs, compilers were developed.

Linker:

A linker is an important utility program that takes the object files, produced by the assembler and compiler, and other code to join them into a single executable file. There are two types of linkers, dynamic and linkage. It is a program in a system which helps to link object modules of a program into a single object file. It performs the process of linking. Linkers are also called as link editors. Linking is a process of collecting and maintaining piece of code and data into a single file. Linker also links a particular module into system library. It takes object modules from assembler as input and forms an executable file as output for the loader. Linking is performed at both compile time, when the source code is translated into machine code and load time, when the program is loaded into memory by the loader. Linking is performed at the last step in compiling a program.

Compiler:

System software that store the complete program, scan it, translate the complete program into object code and then creates an executable code is called a compiler. On the face of it compilers compare unfavorably with interpreters because they −

These are the steps in compiling source code into executable code −

• are more complex than interpreters

• need more memory space

• take more time in compiling source code

Compilation Chain:

However, compiled programs execute very fast on computers. The following image shows the step-by-step process of how a source code is transformed into an executable code –

Source code

→ preprocessing

→Lexical analysis

→parsing

→compiler

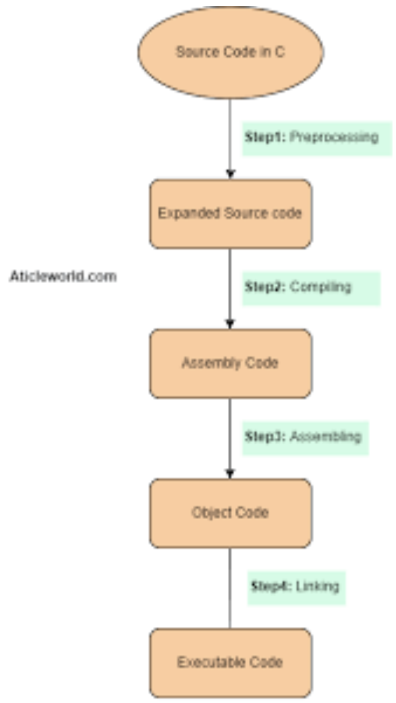
→Assembler

→Object code

→Linker

→Executable file

→Loader



Note: The pre-processing, lexical analysis, and parsing steps are typically done together by the compiler.

Pre-processing – In this stage pre-processor instructions, typically used by languages like C and C++ are interpreted, i.e. converted to assembly level language.

Lexical analysis – Here all instructions are converted to lexical units like constants, variables, arithmetic symbols, etc.

Parsing – Here all instructions are checked to see if they conform to grammar rules of the language. If there are errors, compiler will ask you to x them before you can proceed.

Compiling – At this stage the source code is converted into object code.

Linking (the thing done by the linker)– If there are any links to external files or libraries, addresses of their executable will be added to the program. Also, if the code needs to be rearranged for actual execution, they will be rearranged. The final output is the executable code that is ready to be executed.

CO 4: Internet architecture:***

Topic 1: LAN, WAN, CABLES, etc

Local Area Network (LAN):

A Local Area Network (LAN) is a network of interconnected computers and devices within a limited geographical area, typically confined to a single building, a home, or a campus. LANs are designed to facilitate data sharing, communication, and resource sharing among devices within the same physical location. Here is a comprehensive overview of LANs:

1. Key Characteristics of LANs:

Limited Geographic Scope: LANs cover a relatively small area, typically within a single building or a group of closely located buildings.

High Data Transfer Rates: LANs are capable of high data transfer speeds, often measured in megabits or gigabits per second, making them suitable for tasks that require quick data exchange.

Common Medium: LAN devices typically use a common communication medium, such as Ethernet cables, Wi-Fi, or optical fibers, for transmitting data.

Private Ownership: LANs are usually privately owned and managed by organizations, institutions, or individuals for their specific needs.

2. Components of a LAN:

Computers and Devices: LANs consist of computers, servers, laptops, printers, smartphones, and other devices capable of network communication.

Network Switches and Hubs: Network switches and hubs are used to connect devices within a LAN, enabling data transmission between them.

Router: A router may be used in a LAN to connect it to external networks, such as the Internet. It manages data traffic between the LAN and external networks.

Access Points (APs): In wireless LANs (Wi-Fi), access points serve as the interface between wireless devices and the wired LAN.

3. LAN Topologies:

Star Topology: In a star topology, all devices are connected to a central hub or switch. This design simplifies network management and troubleshooting but can be less fault-tolerant if the central hub fails.

Bus Topology: In a bus topology, all devices share a single communication line. While it's simple and inexpensive, it can be prone to congestion and difficult to expand.

Ring Topology: In a ring topology, devices are connected in a closed loop. Data circulates around the ring until it reaches its destination. Ring topologies are resilient but can be complex to set up.

4. LAN Protocols:

Ethernet: Ethernet is the most common LAN protocol and defines the rules for data transmission over wired connections.

Wi-Fi (Wireless LAN): Wi-Fi protocols (e.g., 802.11ac, 802.11ax) govern wireless LANs, allowing devices to connect without physical cables.

Token Ring: A less common LAN protocol, Token Ring uses a token-passing mechanism to manage access to the network medium.

5. LAN Applications:

File Sharing: LANs enable users to share files and resources (e.g., printers, scanners) among connected devices.

Intranet: LANs are the foundation for creating company intranets, internal websites for sharing information among employees.

Online Gaming: LANs are often used for local multiplayer gaming, where players connect their devices to compete or collaborate in real-time games.

VoIP and Video Conferencing: LANs support Voice over Internet Protocol (VoIP) and video conferencing applications, allowing for cost-effective communication.

6. Security in LANs:

Firewalls: Firewalls are often used to protect LANs from external threats by controlling incoming and outgoing network traffic.

Access Control: Network administrators implement access control policies to restrict unauthorized access to LAN resources.

Encryption: Encryption protocols are used to secure data transmission within a LAN, ensuring that data remains confidential.

Wide Area Network (WAN):

A Wide Area Network (WAN) is a network that spans a large geographic area, often connecting multiple Local Area Networks (LANs) or individual devices across cities, countries, or even continents. WANs are designed to facilitate long-distance communication and data exchange between geographically separated locations. Here is a comprehensive overview of WANs:

1. Key Characteristics of WANs:

Large Geographic Coverage: WANs cover a wide geographical area, which can range from a few kilometers to intercontinental distances.

Public and Private Networks: WANs can be public networks (like the Internet) or private networks established and managed by organizations for their specific needs.

Low Data Transfer Rates: WANs often have lower data transfer rates compared to LANs, and the actual speed can vary depending on factors such as the technology used and network congestion.

Diverse Communication Media: WANs use various communication media, including leased lines, optical fibers, satellite links, microwave links, and the Internet.

2. Components of a WAN:

Routers: Routers play a vital role in WANs by forwarding data packets between networks, determining the best path for data, and ensuring data reaches its destination.

Modems and Media Converters: These devices help connect different types of communication media and enable data transmission over diverse networks.

Communication Links: WANs use various communication links, including leased lines (dedicated connections), public Internet connections, and virtual private networks (VPNs).

WAN Gateways: WAN gateways provide access to remote networks or the Internet, acting as an entry and exit point for data.

3. WAN Technologies:

Point-to-Point Protocol (PPP): PPP is a common protocol used to establish direct connections between two devices over serial links.

Frame Relay: Frame Relay is a WAN technology that provides data transmission with minimal error checking, making it suitable for high-speed data transfer.

ATM (Asynchronous Transfer Mode): ATM is a high-speed WAN technology that uses fixed-size cells to transmit data efficiently.

MPLS (Multiprotocol Label Switching): MPLS is used by service providers to efficiently route data through WANs, ensuring data quality and speed.

4. WAN Applications:

Internet Access: WANs connect users and organizations to the global Internet, providing access to websites, email, and online services.

Branch Office Connectivity: WANs are used to connect branch offices of organizations, enabling centralized data sharing and resource access.

Wide Area File Sharing: Organizations use WANs to share large files and data across multiple locations.

Voice and Video Conferencing: WANs support voice over IP (VoIP) and video conferencing, enabling real-time communication across distances.

5. Security in WANs:

Virtual Private Networks (VPNs): VPNs are commonly used to secure data transmission over public WANs by encrypting data packets.

Firewalls and Intrusion Detection Systems (IDS): These security measures help protect WANs from unauthorized access and cyber threats.

Authentication and Access Control: WANs implement strong authentication mechanisms and access control policies to ensure only authorized users can access network resources.

Wide Area Networks are essential for connecting geographically dispersed locations, facilitating global communication, and enabling remote access to data and services. WAN technologies continue to evolve, providing faster and more reliable connectivity to meet the demands of businesses, institutions, and individuals across the world.

Topic 2: Connection medias:

Cable Media: Twisted Pair, Co-axial and Optical Fiber

(Copied from book)

Normally cable means such a thing that can connect two ends. It is a solid thing through which data signals can go to its destination from the source. Here we can also say that in this media, signal will be only in the cables or cable guide the signal to go to its destination, that's why this media is called guided media. In this case data signals can only be received through cables. So, receiver must be connected with cable. Basically cable is used for short distance communication or networking. But it is also used for high speedy data transfer or high range communication. There are different types of cable media. Among them, most important medias are:

a. Twisted pair cable
b. Co-axial cable
c. Fibre optic cable etc.

Twisted Pair Cable

In twisted pair cable, some pair cables are rolled with each other through which data signal is transmitted. These types of cables are only used for telecommunication. If copper cables are nearby each other, each signal will influence the other signal which called crosstalk Cables are rolled for minimizing crosstalk and other interfaces. If cables are rolled, each signal will make the other signal neutral. Colour coding is used in twisted pair and there is an insulator in every cable. These insulated cables are rolled. Rolled cable pairs are wrapped in a plastic jacket for protection. 4 paid cables are used on this type of cables. Among every pair cables, a cable is covered by white colour plastic code. Each while colour cable has blue, pink, green and brown colour coded cables with it, Cable is connected with connectors depending on these codes. An insulator substance is used for separating two rolled cables. Twisted pair cables are of two types, such as:

1. Unshielded twisted pair cable(UTP)
2. Shielded twisted pair cable(STP)

1. **UTP**: There is no extra shield or cover outside of unshielded twisted pair. We can found different standard uses of UTP cable. Among all these standards, CAT-5 and CAT-6 are much popular. Bandwidth of UTP cable is basically 10 Mbps. But it may reach from 1 to 155 Mbps. It has attenuation. So, its distance is limited to 100 mitre. It has much EMI influence.

2. **STP**: In shielded twisted pair cable, every rolled pair cables are kept in a strong shield or cover. That's why these cables are much protective. Shield is mainly made of aluminium or polyester. Again, every shielded pair cables are kept in a plastic jacket. Special connector is used in STP cable. That's why we will face more problems at the time of using STP cable comparing to UTP cable. This cable is thick and strong, so it is

difficult to shift. Its bandwidth is normally 16 Mbps. But it can reach up to 500 Mbps. Its attenuation is more likely to UTP cable, so its range is limited to 100 mitre. Big advantage of using STP cable is that it reduce EMI influence but it can't fully recover from it.

Advantages of Twisted Pair Cable:

1. Twisted pair cable is mostly used for less distance communication.
2. Can be repaired easily.
3. It is less costly comparing to other cables.
4. It can be installed easily.
5. It is an old data transfer technique.
6. Twisted pair cable is used for sending both analog and digital data.

Disadvantages of Twisted Pair Cable:

1. For sending data more than 2 km, a repeater has to set for twisted pair cable.
2. Transmission loss is respectively high.
3. It can be easily broken because of having thin structure.
4. It is easily influenced by noise signal.

Use of Twisted Pair Cable:

1. This cable is used in telephone line.
2. This cable is also used for digital signalling and LAN.

Co-axial Cable

(Copied form book)

Co-axial cable is a type of electrical cable that has an inner conductor surrounded by a tubular insulating layer, surrounded by a tubular conducting shield. Many co-axial cables also have an insulating outer sheath or jacket. The term co- axial comes from the inner conductor and the outer shield sharing a geometric axis Co-axial cable was invented by English engineer and mathematician Oliver Heaviside, who patented the design in 1880. Co-axial cable differs from other shielded cables because the dimensions of the cable are controlled to give a precise, constant conductor spacing, which is needed for it to function efficiently as a transmission line. Data transfer rate is comparatively high in this type of cables. But data transfer rate depends on the length of the cable. Normally we can send digital data up to 1 km by using co-axial cable. On this case data transfer rate may reach up to 200 Mbps (Megabits per second) and moreover transmission loss is comparatively low.

Co-axial cables used in communication networks are divided into two types. These are-

1. Thinnet: Networks made using thin co-axial cables are called thinnet cables. The width of these co-axial cables is 0.25 inches, and such cables of length 185mb may be used without any repeater. These networks are also called 10 base-2 (10BASE2) networks. Here, 10 is the network bandwidth (10 Mbps) and 2 is the length of the cable in metres (200 metres). (BNC connectors are used in thinnet cables.

2. Thicknet: The width of thicknet co-axial cables is almost 0.5 inches, and such cables of length 500m may be used without any repeater. These networks are also called [10 base 5] networks (10BASES). Vampire taps and drop cables are used within thicknet cables to make connections between devices. Toldin

Characteristics of Co-axial Cables:

1. Can be easily installed.
2. More secure.
3. Cheaper.
4. Allows data transmission over longer distances.
5. Allows for faster data transmission.

Advantages of Co-axial Cables:

1. Less expensive compared to fibre optic cables.
2. This cables is used in both analogue and digital data transmissions.
3. Data can be transmitted over longer distances compared to twisted pair cables.
4. Loss of transmission is comparatively low.
5. This cables is more widely used in television networks

Disadvantages of Co-axial cables:

1. More expensive compared to twisted pair cables.
2. Establishing a connection between networks using co-axial cables is more complicated.
3. Its data transmission rate depends on the length of the cables.
4. Data cannot be sent over a distance of more than 1km without a repeater.

Uses of Co-axial Cables:

• Co-axial cables are used for Local Area networks,

• Its use is also seen in cables television systems

Fiber Optic Cable

(Copied form book)

Fibre optic cables are the most powerful cables among cables media. The main cables at the centre of a fibre optic cables is made from silica, glass or transparent plastic. The main advantage of using glass as media is that, it does not have EML That is why possibility of changing data signal is almost impossible. Since data is transmitted through glasses, its speed is very high. Fibre optic cables transmit light signals instead of electrical signals. They carry data from the source to the destination by the total internal reflection of light. Light signals cross through fibre optic cables in two ways: laser systems or LED systems. Although it is expected that lasers can be used in fibre optic cables, instead of using lasers, LEDs are more widely used today. This is because, LED devices are less expensive compared to lasers. Moreover LEDs are more durable. Fibre optic cables used in today's Earth have the bandwidth range of 100Mbps to 2Gbps.

Swiss physicist Daniel Colladon and French physicist Jacques Babinet discovered the technique of the reflection of light inside a fibre. Using this concept, Heinrich Lamm, and one of his students became the first persons in history to be able to send an image through a transparent glass fibre in the year 1920. However the image was blurred because the transmission technique they used had no cladding. That is why most of the light spread out from the cable, and the received signal was poor. Later, American physicist Brian O'Brien became the first individual to successfully send and receive digital signals using a cladded optical fibre.

Structure:

For making the optical fibre, borosilicate glass, soda-lime silica glass and soda-alumina silica glass (alumina-silicate) are used. The mentionable characteristics of these glasses are transparency, chemical stability, and uncomplicated processing capability. Plastics are often used in the cladding of fibres..

Fibre optic cables have 3 parts. These are-

(1) Core: a dry-electric core whose width is mainly 8 to 100 micron.

(ii) Cladding: a cladding of Kevlar covers the optical fibre at its centre. It reflects light. For this reason, light signals can be reflected through fibre optic cables.

(iii) Jacket: Which works as a covering.

Characteristics of Fibre Optic Cables:

Since light signals do not spread out as electrical signals, the attenuation is negligible.

Light signals are transmitted from it, instead of electrical signals. It transfers data in the range of Gbps or more. Fibre optic cables are often used as the network backbone of a much larger network.

Fibre optic cables do not have EMI, so they can be used in almost all circumstances.

Advantages of Fibre Optic Cables:

1. High band speed Free from EMI
2. Accurate data transfers
3. Not influenced by temperature or pressure of the environment
4. Lightweight and portable Minimum energy loss
5. A repeater does not need to be setup at any particular place
6. Saves data and provides greater security

Disadvantages of Fibre Optic Cables:

1. Installation is very complicated
2. Expensive compared with other cables
3. Highly skilled workers are needed for installation and maintenance It cannot be bent easily, so installation is more difficult
4. Fibre optic cables cannot be split easily. Electric fusion or epoxy resin is required for repairing damaged cables

Classification of Fibre Optic Cables:

Depending on the refraction of structural elements fibre optic cables are divided into two types. Step-index fibre:

1. Refraction of the core of step-index fibres is the same everywhere.
2. Graded-index fibre: Refraction of the core is much higher at the core of a graded-index fibre and diminishes gradually towards its radius. Variation in the refraction of the core causes variation in the direction of the light rays within the two types of fibres.

Depending on the diameter of the core, fibre optic cables are divided into two types:

1. Single mode fiber: A single mode cable is a single strand of Single-mode fiber glass fibre with a diameter of 8.3 to 10 microns that has one mode of transmission. Single mode fibre with a relatively narrow diameter, through which only one mode will propagate typically 1310nm or 1550nm. Single mode fibre optic cables are used over long distances.
2. 2. Multi mode fiber: The core size of multimode fibres is 62.5/125 micron. It is mostly used and suitable in network applications. Light waves are dispersed into numerous paths in multi and signal can reach its destination at the same time using these paths. That is why receiver thinks that he/she receives data through only one light wave. It can utilise LEDs, so the cost of multi mo fibres is low.

Submarine Cables:

Submarine cables are the unheralded heroes of global communication networks, serving as the critical infrastructure facilitating the transmission of data, voice, and internet traffic across immense oceanic expanses. They are made of a number of five optical cables and have a high about of shielding applied with them. They also have occasional repeaters to enhance the signal strength.

1. SEA-ME-WE 4 (South East Asia-Middle East-Western Europe 4):

SEA-ME-WE 4 is a transcontinental submarine cable system covering an astonishing distance of approximately 20,000 kilometers. This remarkable undersea network interconnects Southeast Asia, the Middle East, and Western Europe via numerous landing points along its route.

Commissioned in 2005, SEA-ME-WE 4 has undergone a series of upgrades and expansions to bolster its data transmission capabilities continually. Presently, it supports data speeds of up to 1.28 terabits per second (Tbps) [Design capacity: 1.28 Tbit/s (2005); 2.8 Tbit/s (2010); 4.6 Tbit/s (2015) Lit capacity: 2.3 Tbits/s/pair (two fibre pairs)], enabling rapid international data exchange.

SEA-ME-WE 4 plays a pivotal role in facilitating international data traffic between Europe, Asia, and the Middle East, serving as a linchpin in connecting nations and regions for both commercial ventures and personal communications.

2. SEA-ME-WE 5 (South East Asia-Middle East-Western Europe 5):

SEA-ME-WE 5, a more recent and advanced addition to the undersea cable landscape, mirrors its predecessor's route, connecting Southeast Asia, the Middle East, and Western Europe. Launched in 2016, it represents a substantial upgrade over SEA-ME-WE 4.

Extending across approximately 20,000 kilometers, SEA-ME-WE 5 leverages state-of-the-art fiber-optic technology to offer significantly higher data transmission capacities. It currently supports data speeds of up to 24 Tbps [Design capacity: 38 Tbit/s (3 fibre pairs) Lit capacity: 17 Tbit/s], signifying a substantial leap in connectivity prowess.

SEA-ME-WE 5 is engineered to meet the burgeoning demands of international data traffic, including the surging reliance on cloud computing, high-definition video streaming, and intricate global business operations.

Topic 3: Mobile Internet

Mobile phone internet connectivity in a cellular network, such as BTS (Base Transceiver Station) and MC (Mobile Core), operates through a complex system of hardware and software components. Here's a simplified explanation of how mobile internet works in a chain involving these key elements:

1. User Device (Mobile Phone):

Your mobile phone is the endpoint of the chain, and it contains the necessary hardware and software to connect to cellular networks.

2. Base Transceiver Station (BTS):

The BTS is the first point of contact in the chain. It's a fixed station that transmits and receives radio signals with mobile devices within its coverage area. Each BTS serves a specific geographical area known as a cell.

When you use your mobile phone, it communicates wirelessly with the nearest BTS. Your phone sends data, such as requests for web pages or apps, to the BTS through radio waves.

3. Base Station Controller (BSC):

The BSC is responsible for managing one or more BTSs. It coordinates the resources allocated to each BTS, handles call setup and termination, and manages handovers (when you move from one cell to another without dropping the call).

The BSC plays a crucial role in routing data and voice traffic between BTSs and the Mobile Core (MC).

4. Mobile Core (MC):

The Mobile Core is the central part of the cellular network. It consists of various components, including Mobile Switching Centers (MSCs), Home Location Registers (HLRs), and Serving GPRS Support Nodes (SGSNs), among others.

When you access the internet or make a call, the MC processes your request. It routes your data or voice to the appropriate destination, whether it's within the same cellular network or to an external network (e.g., the internet).

5. Gateway to the Internet:

To access the internet, the MC acts as a gateway. It forwards your data requests to external networks through various network elements, including routers and switches.

Once your request leaves the MC, it traverses through the internet, where it may encounter multiple servers and routers that direct the data to its final destination.

6. **Data Exchange and Retrieval**:

Your data request, such as a web page or app request, goes through multiple servers and routers on the internet until it reaches the destination server.

The destination server processes your request, retrieves the requested data (e.g., a web page), and sends it back through the same path.

7. Return Path:

The data you requested makes its way back through the internet and the MC, following a similar path in reverse.

The MC ensures that the data is correctly routed back to your mobile phone through the BSC and BTS.

8. Display on Your Mobile Device:

Once your mobile device receives the data, it processes and displays it on your screen. You can now interact with the internet or the requested service, such as browsing a webpage or using an app.

This entire process occurs in a matter of seconds, allowing you to access the internet and communicate with others seamlessly through your mobile phone. The cellular network infrastructure, from BTS to MC, plays a vital role in ensuring reliable and efficient mobile internet connectivity.

Topic 5: Connection devices:

Routers:

Routers are fundamental networking devices that play a pivotal role in directing data packets between different networks, allowing devices to communicate with each other. They serve as traffic controllers on the internet and within local area networks (LANs). Here's a comprehensive overview of routers and their functions:

1. What is a Router:

A router is a network device that operates at the network layer (Layer 3) of the OSI (Open Systems Interconnection) model. It connects multiple networks together and determines the best path for data packets to travel between them.

2. Key Functions of Routers:

Packet Forwarding: Routers inspect the destination IP address of incoming data packets and decide where to send them based on routing tables. They are responsible for forwarding packets between different networks.

Network Address Translation (NAT): Routers use NAT to allow multiple devices within a private network to share a single public IP address for internet access, conserving public IP addresses.

Firewall Functionality: Many modern routers include firewall capabilities, filtering incoming and outgoing traffic to enhance network security.

Dynamic Routing: Routers can dynamically adjust their routing tables based on network conditions, using protocols like RIP, OSPF, and BGP.

3. Components of a Router:

CPU (Central Processing Unit): The router's CPU performs routing decisions, executes routing protocols, and handles other network-related tasks.

Memory: Routers have both volatile (RAM) and non-volatile (flash) memory. RAM stores routing tables and running configuration, while flash memory holds the router's operating system and configuration files.

Interfaces: Routers have multiple network interfaces, including Ethernet ports, WAN (Wide Area Network) ports, and wireless interfaces for connecting to various network types.

Operating System: Routers run specialized operating systems designed for networking tasks. Examples include Cisco's IOS, Juniper's Junos, and open-source options like DD-WRT.

4. Home Routers vs. Enterprise Routers:

Home Routers: These are consumer-grade routers commonly found in households. They provide basic routing, NAT, firewall, and wireless functionality. They are typically used for connecting devices to the internet and creating local wireless networks.

Enterprise Routers: Enterprise-grade routers are more powerful and feature-rich. They are used in business environments and data centers to handle larger networks, more complex routing, and advanced security features.

5. WAN and LAN Connectivity:

Routers have both WAN and LAN interfaces. WAN interfaces connect to external networks, such as the internet or wide-area networks. LAN interfaces connect to local devices within a network.

6. DHCP Server:

Many routers include a DHCP server that dynamically assigns IP addresses to devices on the local network, simplifying network configuration.

7. Quality of Service (QoS):

Some routers support QoS features, allowing users to prioritize certain types of network traffic to ensure a better user experience for critical applications.

8. Routing Tables:

Routers maintain routing tables that contain information about the best paths for data packets to take when forwarding them to their destinations. These tables are essential for efficient packet routing.

9. Redundancy:

In enterprise networks, routers are often used in pairs or more to provide redundancy and failover capabilities, ensuring network availability.

10. Security:

Routers can enhance network security by implementing access control lists (ACLs), firewall rules, and intrusion detection/prevention systems.

In summary, routers are essential ne

Switch:

A switch is a critical networking device that operates at the data link layer (Layer 2) of the OSI model. Its primary function is to efficiently forward data packets within a local area network (LAN) by making intelligent decisions based on MAC addresses. Switches are essential for modern network infrastructure and play a pivotal role in ensuring fast and reliable data transmission. Here's a detailed overview of switches and their key features:

1. Basic Function:

A switch receives data packets from devices within the same LAN and forwards those packets only to the specific destination device for which they are intended.

2. MAC Address Learning:

Switches maintain a MAC address table, also known as a CAM (Content Addressable Memory) table. This table associates MAC addresses with specific switch ports.

When a device sends a data packet to the switch, the switch learns the source MAC address by examining the source port.

It then updates its MAC address table with this information, associating the MAC address with the source port.

3. Intelligent Packet Forwarding:

When a data packet arrives at the switch with a destination MAC address, the switch looks up the MAC address in its table.

If the MAC address is found, the switch forwards the packet only to the port associated with the destination device, reducing network congestion and optimizing bandwidth usage.

If the MAC address is not found in the table, the switch broadcasts the packet to all connected devices within the LAN until it learns the location of the destination device.

4. Unicast, Broadcast, and Multicast Traffic:

Switches efficiently handle unicast traffic (one-to-one communication) by forwarding data packets directly to the target device.

Broadcast traffic (one-to-all communication) is sent to all devices within the LAN to ensure all receive the broadcast message.

Multicast traffic (one-to-many communication) is directed only to devices that have subscribed to the multicast group, conserving network bandwidth.

5. VLAN Support:

Many switches support Virtual LANs (VLANs), which allow network administrators to segment a single physical LAN into multiple logical LANs. This enhances network organization, isolation, and security.

6. High-Speed Ports:

Switches often come with high-speed Ethernet ports, such as Gigabit Ethernet (1 Gbps) or 10 Gigabit Ethernet (10 Gbps), to accommodate high-bandwidth applications and services.

7. Quality of Service (QoS):

Some switches offer QoS features, allowing network administrators to prioritize specific types of traffic for optimized performance.

8. Managed vs. Unmanaged Switches:

Managed switches provide greater control and configurability, allowing network administrators to set up VLANs, monitor traffic, and implement security policies.

Unmanaged switches are simpler and require minimal configuration, making them suitable for smaller networks where advanced features are not needed.

9. Redundancy and Spanning Tree Protocol:

In larger networks, switches can be configured with redundancy using protocols like Spanning Tree Protocol (STP) to ensure network availability in case of link failures.

Switches are the backbone of local networks, providing efficient and intelligent data packet forwarding. They are essential for ensuring high-speed data transmission, reducing network congestion, and organizing LANs effectively. Whether in homes, offices, data centers, or enterprise environments, switches are crucial for modern network infrastructure.

Routers, hubs, and switches

are all essential networking devices, but they serve distinct purposes and have different functions in a network. Let's explore each of these devices and their roles:

Router:

Function: Routers are primarily responsible for connecting multiple networks together and directing data packets between them. They operate at the network layer (Layer 3) of the OSI model and make decisions based on IP addresses.

Key Features:

1. Packet Forwarding: Routers determine the best path for data packets to travel between different networks based on routing tables.
2. Network Address Translation (NAT): Routers use NAT to allow multiple devices within a private network to share a single public IP address for internet access.
3. Firewall: Many routers include firewall functionality to filter incoming and outgoing traffic, enhancing network security.
4. Dynamic Routing: Routers can dynamically adjust their routing tables based on network conditions using routing protocols.

Typical Use Cases: Routers are used to connect home or office networks to the internet, manage traffic between different network segments, and enforce security policies.

Hub:

Function: Hubs are basic networking devices that operate at the physical layer (Layer 1) of the OSI model. They simply receive incoming data packets and broadcast them to all connected devices within the same network segment.

Key Features:

1. Broadcasting: Hubs lack intelligence and do not make any decisions about where to send data packets. They broadcast data to all connected devices, leading to inefficient use of network bandwidth.
2. No Filtering: Hubs do not filter or manage network traffic, making them unsuitable for complex network setups

Typical Use Cases: Hubs are rarely used in modern networks due to their inefficiency. They have been largely replaced by switches, which are more intelligent and efficient.

Switch:

Function: Switches are networking devices that operate at the data link layer (Layer 2) of the OSI model. They are designed to efficiently forward data packets within a local network.

Key Features:

1. MAC Address Learning: Switches maintain a MAC address table to associate MAC addresses with specific switch ports. This allows them to intelligently forward data only to the appropriate destination device, reducing network congestion.

3. Unicast Traffic: Switches forward data only to the specific device for which it is intended, rather than broadcasting it to all devices like hubs.
4. VLAN Support: Many switches support Virtual LANs (VLANs) to segment a network into logical subnetworks, improving network organization and security.
5. High-Speed Ports: Switches often have high-speed ports, such as Gigabit Ethernet or 10 Gigabit Ethernet, for fast data transmission.

Typical Use Cases: Switches are used within local networks to efficiently manage and direct traffic between devices. They are commonly found in homes, offices, data centers, and enterprise networks.

In summary, routers connect multiple networks and make routing decisions based on IP addresses, hubs are basic devic

Topic 6: Internet Access Control

Firewall:

A firewall is a network security device or software application that acts as a barrier between a trusted internal network and untrusted external networks, such as the internet. Its primary purpose is to monitor and control incoming and outgoing network traffic, filtering data packets based on a set of predefined rules or security policies. Firewalls play a critical role in protecting networks and devices from unauthorized access, cyberattacks, and other security threats. Here's an in-depth look at firewalls and their key functions:

1. Packet Filtering:

Firewalls examine data packets as they traverse the network and determine whether to allow or block them based on predefined rules. These rules can be configured to filter traffic by source IP address, destination IP address, port number, and protocol.

2. Stateful Inspection:

Many modern firewalls use stateful inspection, also known as dynamic packet filtering. This technique tracks the state of active connections and allows firewalls to make context-aware decisions. For example, it can distinguish between legitimate responses to outgoing requests and unsolicited incoming traffic.

3. Access Control:

Firewalls enforce access control policies to restrict or permit traffic to and from specific IP addresses, applications, or services. This helps prevent unauthorized access to sensitive resources.

4. NAT (Network Address Translation):

Firewalls often include NAT functionality, allowing them to map multiple internal private IP addresses to a single public IP address. This conceals internal network structure and conserves public IP addresses.

5. Application Layer Filtering:

Some advanced firewalls perform deep packet inspection (DPI) to examine the application-layer data in network packets. This enables them to detect and block specific applications or protocols, such as instant messaging or peer-to-peer file sharing.

6. Proxy Services:

Firewalls can act as proxy servers, forwarding network requests on behalf of clients. This helps hide the client's identity and adds an additional layer of security by inspecting and filtering traffic at the proxy.

7. Intrusion Detection and Prevention (IDP):

Many modern firewalls include intrusion detection and prevention features to identify and block suspicious or malicious traffic patterns and behaviors.

8. VPN (Virtual Private Network) Support:

Firewalls can facilitate secure remote access by supporting VPN technologies, allowing remote users to connect to the internal network securely over the internet.

9. Logging and Reporting:

Firewalls maintain logs of network activity, which can be invaluable for monitoring and auditing network security. They often provide reporting capabilities to help administrators analyze and respond to security incidents.

10. Hardware and Software Firewalls:

Hardware firewalls are dedicated devices that offer robust protection for an entire network. They are often used in enterprise environments.

Software firewalls are installed on individual devices, such as computers or servers, and provide protection at the device level. They are common in home networks and can be customized for specific needs.

11. Zones and Segmentation: Firewalls are used to create security zones within a network, segmenting it into different trust levels. For example, a firewall can separate an organization's internal network from its guest Wi-Fi network.

CO 5: Web architecture:***

Topic 1: Web connection systems

Dynamic Host Configuration Protocol (DHCP): Automating IP Address Assignment**

Dynamic Host Configuration Protocol (DHCP) is a network protocol used to automate and simplify the process of assigning IP addresses and other network configuration parameters to devices on a local network. DHCP plays a critical role in managing IP addresses within a network, making it easier to set up and maintain large-scale networks. Here's how DHCP works:

1. DHCP Server:

A DHCP server is a device or software application responsible for managing and distributing IP addresses within a network. Most commonly, routers and dedicated DHCP servers handle this role.

2. DHCP Client:

DHCP clients are devices, such as computers, smartphones, or any networked device, that request an IP address and other network configuration settings from the DHCP server.

3. DHCP Lease Process:

DHCP Discover: When a DHCP client connects to a network, it sends a DHCP Discover broadcast message to discover DHCP servers available on the network. This message is sent as a request for an IP address lease.

DHCP Offer: DHCP servers on the network receive the Discover message and respond with a DHCP Offer. Each server may offer an IP address lease, along with other network configuration information, such as subnet mask, default gateway, and DNS server addresses.

DHCP Request: The client reviews the DHCP Offers and selects one. It sends a DHCP Request message to the chosen DHCP server, indicating its acceptance of the offered IP address lease.

DHCP Acknowledge: The DHCP server that received the Request message sends a DHCP Acknowledge message to the client, confirming the assignment of the chosen IP address lease. This message also includes the lease duration.

4. Lease Duration:

The DHCP server assigns an IP address lease to the client for a specific duration, which is configured by the network administrator. This lease duration can vary from minutes to days or even longer.

As the lease expiration approaches, the client can renew the lease by sending a DHCP Request to the server. If the server accepts the renewal, it extends the lease duration. If not, the client must go through the DHCP Discover process again to obtain a new lease.

5. IP Address Management:

DHCP servers maintain a pool of available IP addresses and keep track of which addresses are currently leased to clients. This prevents IP address conflicts and ensures efficient use of available addresses.

Benefits of DHCP:

Simplified Network Configuration: DHCP automates the assignment of IP addresses and other network settings, reducing the need for manual configuration, which is particularly valuable in large networks.

Reduced IP Address Conflicts: DHCP helps prevent IP address conflicts by carefully managing address allocation.

Efficient Resource Utilization: DHCP servers can reclaim and reuse IP addresses when clients disconnect or release their leases, optimizing IP address usage.

Centralized Management: Network administrators can centrally manage IP address assignments and network settings from the DHCP server, making it easier to implement changes or updates.

Scalability: DHCP scales well for networks of various sizes, from small home networks to large enterprise networks.

https://tools.ietf.org/html/rfc2131

Internet Protocol address (IP address):

An IP address (Internet Protocol address) is a numerical label assigned to each device participating in a computer network that uses the Internet Protocol for communication. IP addresses serve two primary functions: host or network interface identification and location addressing.

Here are some key aspects of IP addresses:

1. Host Identification:

IP addresses are used to uniquely identify devices on a network. Each device connected to a network, such as a computer, smartphone, router, or server, is assigned an IP address.

The format of an IP address depends on the version of the Internet Protocol being used. The two most common versions are IPv4 (Internet Protocol version 4) and IPv6 (Internet Protocol version 6).

2. IPv4 Address Format:

Pv4 (Internet Protocol version 4) is the older and most widely used IP address format. It uses 32 bits and can represent approximately 4.3 billion unique addresses. IPv4 addresses consist of four sets of numbers separated by periods (e.g., 192.168.1.1).

Each set, called an octet, can range from 0 to 255, resulting in a total of approximately 4.3 billion unique IPv4 addresses.

3. IPv6 Address Format:

IPv6 (Internet Protocol version 6) is the latest IP address format, designed to overcome the IPv4 address exhaustion issue. It uses 128 bits, resulting in an astronomically large number of unique addresses, ensuring the internet's growth for the foreseeable future. IPv6 addresses are longer and use hexadecimal notation (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334).IPv6 was introduced to address the exhaustion of IPv4 addresses and provides a virtually limitless number of unique addresses.

4. Public IP Addresses:

Public IP addresses are globally unique and are assigned to devices that directly connect to the internet. These addresses are used for communication between devices across the internet.

Internet Service Providers (ISPs) or organizations that operate their own networks typically allocate public IP addresses.

5. Private IP Addresses:

Private IP addresses are used within local, private networks like home or office networks. They are not routable on the public internet, allowing multiple devices to share a single public IP address.

Common private IP address ranges include 192.168.x.x, 10.x.x.x, and 172.16.x.x to 172.31.x.x.

6. Subnet Masks:

Subnet masks are used to divide IP address spaces into subnets within a network. They determine the network portion and host portion of an IP address.

For example, in the IP address 192.168.1.1 with a subnet mask of 255.255.255.0, the first three octets (192.168.1) represent the network, and the last octet (1) represents the host.

7. Static vs. Dynamic IP Addresses:

Static IP addresses are manually configured and remain constant over time. They are often used for servers and network devices that need consistent, unchanging addresses.

Dynamic IP addresses are assigned automatically by a DHCP (Dynamic Host Configuration Protocol) server. Most consumer devices use dynamic IP addresses, which can change periodically.

8. IP Address Classes (IPv4):

IPv4 addresses were historically divided into classes based on their usage and size:

Class A: Large networks with fewer hosts.

Class B: Medium-sized networks with moderate hosts.

Class C: Smaller networks with many hosts.

However, modern networking practices have moved away from class-based IP address allocation.

9. **Reserved IP Addresses:**

Certain IP address ranges are reserved for specific purposes, like loopback addresses (127.0.0.1) used for testing, and multicast addresses for efficient group communication.

10. IP Address Assignment:

IP addresses can be assigned statically by a network administrator, or dynamically through DHCP servers that automatically allocate IP addresses to devices when they join a network.

11. Network Address Translation (NAT):

NAT is a technique used in routers to allow multiple devices within a private network to share a single public IP address for accessing the internet. It maps internal private IP addresses to a single external public IP address.

10. IP Address Classes (IPv6):

Network Address Translation (NAT):

Network Address Translation (NAT) is a technology used in routers and firewall devices to enable multiple devices on a local network to share a single public IP address for internet access. NAT plays a crucial role in conserving IPv4 addresses and improving network security. Here's a comprehensive overview of NAT:

1. Why NAT is Needed:

The pool of available IPv4 addresses is limited, and as more devices connect to the internet, there's a shortage of unique public IP addresses.

NAT allows multiple devices within a private network to access the internet using a single public IP address, overcoming this shortage.

2. How NAT Works:

NAT operates at the router or firewall level. It keeps a translation table that maps private IP addresses to a single public IP address.

When devices on the local network request data from the internet, the router rewrites the source IP address in outgoing packets to its own public IP address.

When responses come back from the internet, the router uses the translation table to route the data to the correct device on the local network based on the port number.

3. Types of NAT:

Static NAT: Also known as one-to-one NAT, it maps a single private IP address to a single public IP address, typically used for hosting services.

Dynamic NAT: Maps multiple private IP addresses to a pool of public IP addresses, allocating them dynamically as needed.

PAT (Port Address Translation): Also called NAT overload, PAT maps multiple private IP addresses to a single public IP address, differentiating them based on port numbers. This is the most common form of NAT.

4. Benefits of NAT:

IP Address Conservation: NAT allows many devices on a local network to share a single public IP address, helping conserve IPv4 addresses.

Enhanced Security: By hiding internal private IP addresses, NAT provides a level of security against direct attacks from the internet.

Simplified Network Design: NAT simplifies the configuration of devices on a local network, as they don't require unique public IP addresses.

5. Limitations of NAT:

NAT can sometimes cause issues with certain online applications or services that rely on direct peer-to-peer connections. Port forwarding or UPnP (Universal Plug and Play) can be used to address these issues.

NAT introduces a level of complexity in network management, especially in large-scale enterprise networks.

6. IPv6 and NAT:

IPv6 was developed to address the limitations of IPv4 and eliminate the need for NAT. IPv6 provides a vastly larger pool of unique addresses, making NAT less necessary in IPv6 networks.

7. Impact on Home Networks:

Most home routers use NAT to allow multiple devices within a household to share a single public IP address provided by the internet service provider (ISP). This is often referred to as a "private network."

In summary, NAT is a fundamental technology that helps extend the life of IPv4 addresses by allowing multiple devices in a local network to share a single public IP address. It provides enhanced security and network address conservation, making it a critical component of modern networking, especially in the transition to IPv6.

Domain Name System (DNS): The Internet's Address Book

The Domain Name System (DNS) is a hierarchical decentralized system that plays a critical role in translating human-friendly domain names (like www.example.com) into machine-readable IP addresses (such as 192.0.2.1). It is often described as the "phone book" of the internet, as it helps users locate websites and other online resources. Here's an overview of how DNS works:

1. Domain Names:

Domain names are user-friendly alphanumeric labels used to identify websites and resources on the internet. Examples include google.com, amazon.com, and openai.org.

2. IP Addresses:

Every device connected to the internet, including web servers, routers, and user devices, is assigned a unique numerical identifier called an IP address (e.g., 203.0.113.1).

3. The Need for DNS:

While computers and servers communicate using IP addresses, humans find it easier to remember domain names. DNS bridges this gap by mapping domain names to their corresponding IP addresses.

4. DNS Hierarchy:

The DNS system is hierarchical, organized into levels or domains. At the top of the hierarchy is the root domain, followed by top-level domains (TLDs) like .com, .org, and .net, and then subdomains (e.g., www.example.com).

5. DNS Resolver:

When you enter a domain name (e.g., www.example.com) in your web browser, your computer contacts a DNS resolver. This resolver could be your ISP's DNS server or a public DNS resolver like Google's (8.8.8.8).

6. DNS Query:

The DNS resolver initiates a DNS query, asking other DNS servers for the IP address associated with the requested domain name.

7. Recursive vs. Iterative Queries:

Recursive Query: If the resolver doesn't already have the IP address, it performs a recursive query. It starts at the root DNS server, then goes to the appropriate TLD server, and finally reaches the authoritative DNS server responsible for the domain.

Iterative Query: In an iterative query, the resolver asks each DNS server for the next DNS server to contact until it reaches the authoritative DNS server.

8. Authoritative DNS Server:

The authoritative DNS server is responsible for providing the IP address associated with the requested domain. It has the authoritative information for the domain's DNS records.

9. Caching:

To reduce the load on DNS servers and speed up future queries, DNS resolvers often cache IP addresses and their corresponding domain names. Cached records have a time-to-live (TTL) value, indicating how long they are considered valid.

10. Response:

The authoritative DNS server responds to the DNS resolver with the IP address. The resolver then forwards this information to your device.

11. Accessing the Website:

With the IP address obtained from the DNS resolver, your device can now connect to the web server associated with the requested domain name. This allows you to access the website.

Benefits of DNS:

DNS simplifies internet navigation by using human-readable domain names.

It allows websites to change their IP addresses without affecting users, as long as the DNS records are updated.

DNS provides a distributed and fault-tolerant system, ensuring that internet resources remain accessible even if some DNS servers are offline.

In essence, DNS is a fundamental part of the internet infrastructure that enables users to access websites, send emails, and use various online services using easy-to-remember domain names instead of complex IP addresses. It plays a vital role in making the internet user-friendly and accessible.

ROOT DNS SERVER

The Root DNS Server is a crucial component of the Domain Name System (DNS) hierarchy and serves as the starting point for DNS resolution. It's responsible for handling requests to resolve top-level domains (TLDs) and directs those requests to the appropriate TLD DNS servers. Here's a more detailed explanation of the Root DNS Server:

1. DNS Hierarchy:

The DNS system is organized hierarchically, with the Root DNS Server sitting at the top of the hierarchy. Below the Root DNS Server are the TLD DNS servers, followed by authoritative DNS servers for specific domains and subdomains.

2. Top-Level Domains (TLDs):

TLDs are the highest level of domains in the DNS hierarchy. Examples include .com, .org, .net, .gov, and .edu. There are also country-code TLDs (ccTLDs) like .uk, .ca, and .jp, which represent specific countries or regions.

3. How It Works:

When a user enters a domain name (e.g., www.example.com) in their web browser or any other application, the DNS resolver starts the process of translating that domain name into an IP address.

If the DNS resolver doesn't have the IP address for the domain in its cache, it contacts a DNS Root Server to start the resolution process.

4. Root DNS Server Response:

The Root DNS Server doesn't have information about specific domain names like www.example.com. Instead, it provides information about the authoritative DNS servers for TLDs.

The Root DNS Server responds to the resolver with the IP addresses of the authoritative DNS servers for the relevant TLDs. For example, if the requested domain is www.example.com, the Root DNS Server would provide the IP address of the DNS server responsible for the .com TLD.

5. Recursive DNS Queries:

The DNS resolver, having received the IP address of the TLD DNS server from the Root DNS Server, then sends another DNS query to the TLD DNS server. This query seeks information about the authoritative DNS server for the next level of the domain hierarchy.

6. Further Resolution:

The TLD DNS server responds to the resolver with the IP address of the authoritative DNS server for the second-level domain. In the case of www.example.com, this would be the DNS server responsible for example.com.

7. Completing the Query:

The DNS resolver continues this process, sending queries to authoritative DNS servers down the hierarchy until it receives the final answer—an IP address for the requested domain, such as www.example.com.

8. Caching:

DNS resolvers often cache DNS responses to improve efficiency and reduce the load on DNS servers. Cached information has a Time-to-Live (TTL) value that determines how long it is considered valid.

9. Redundancy:

There are multiple Root DNS Servers distributed worldwide to ensure redundancy and fault tolerance. These servers are managed by various organizations and are identified by letters from A to M.

The Root DNS Server system is a critical part of the global DNS infrastructure, helping users resolve domain names into IP addresses and navigate the internet. It provides the initial guidance needed to direct DNS queries to the appropriate authoritative servers for further resolution.

Hole Punching: Bridging Network Boundaries for Communication

Hole punching is a networking technique used to establish a direct communication channel between two devices or peers located behind separate network address translation (NAT) routers or firewalls. NAT routers are commonly used to allow multiple devices within a private network to share a single public IP address. This can pose a challenge for devices within different private networks trying to initiate direct communication. Hole punching helps overcome this challenge by creating temporary openings, or "holes," in the NAT configurations to allow traffic to pass through. Here's how hole punching works:

1. Initial Configuration:

Two devices, let's call them Device A and Device B, are located behind different NAT routers and want to establish a direct communication channel.

2. Third-Party Server:

A third-party server, often referred to as a "rendezvous server" or "relay server," is used as an intermediary to facilitate hole punching. This server has a publicly accessible IP address and is reachable by both devices.

3. Initialization:

Device A and Device B independently connect to the third-party server. They inform the server about their intention to communicate with each other.

4. NAT Configuration:

The NAT routers of Device A and Device B have no prior knowledge of each other and initially block incoming traffic from external sources.

5. Request for Hole Punching:

Device A and Device B request the third-party server to help them establish a connection. The server relays their requests to each other.

6. NAT Traversal:

Device A sends a packet to Device B through the NAT router. However, the NAT router initially blocks this traffic, as it doesn't match any established outbound connections.

7. Symmetric NAT Mapping:

The third-party server helps Device B determine the source port and IP address from which Device A's traffic is arriving. This information is essential for hole punching.

8. Hole Punching Attempt:

Device B sends a packet to Device A's public IP address and port, simulating a response to the traffic it received earlier. This attempt often leads the NAT router to open a temporary "hole" in its firewall to allow incoming traffic from Device A.

9. Successful Hole Punching:

If the hole punching attempt is successful, Device A and Device B can now communicate directly without relying on the third-party server.

Hole punching is a clever technique that leverages the dynamics of NAT routers to establish direct communication between devices behind them. It is commonly used in peer-to-peer (P2P) applications, online gaming, and video conferencing, where users want to communicate directly without relying on centralized servers. However, it's worth noting that hole punching is not always guaranteed to work, as it depends on the specific NAT configurations and types involved.

Topic 3: Browsing

HTTP (Hypertext Transfer Protocol): The Language of the Web

HTTP, or Hypertext Transfer Protocol, is an application layer protocol used for transferring data over the World Wide Web. It forms the foundation of data communication on the internet and enables the exchange of web pages, images, videos, and other resources between web servers and web clients, such as web browsers. Here's an overview of HTTP and its key components:

1. Request-Response Model:

HTTP operates on a client-server model, where a client (typically a web browser) requests resources from a web server.

2. Key Components:

Uniform Resource Locator (URL): URLs are used to specify the location of a resource on the web. They include the protocol (e.g., http:// or https://), domain name, port, path, and query parameters.

HTTP Methods: HTTP defines various methods (verbs) that clients can use to interact with web servers. The two most common methods are:

GET: Used to request data from a web server. It typically retrieves data without causing any change or side effects on the server.

POST: Used to submit data to a web server for processing. It often involves sending data that will be stored or processed on the server.

3. Request-Response Cycle:

When a client (e.g., a web browser) wants to retrieve a web page or resource, it sends an HTTP request to a web server.

The web server processes the request, retrieves the requested resource, and sends back an HTTP response to the client.

4. HTTP Headers:

Both requests and responses include HTTP headers, which provide additional information about the data being transferred. Headers can convey details like the content type, cache settings, and more.

5. Status Codes:

HTTP responses include status codes that indicate the outcome of the request. Common status codes include:

200 OK: The request was successful, and the response contains the requested data.

404 Not Found: The requested resource was not found on the server.

500 Internal Server Error: The server encountered an error while processing the request.

6. Stateless Protocol:

HTTP is a stateless protocol, meaning each request-response interaction is independent of previous interactions. The server does not retain information about past requests from the same client.

7. HTTPS:

HTTPS (Hypertext Transfer Protocol Secure) is a secure version of HTTP. It uses encryption (SSL/TLS) to protect data transferred between the client and server, ensuring confidentiality and integrity.

8. Cookies and Sessions:

To maintain state in stateless HTTP, mechanisms like cookies and sessions are used. Cookies are small pieces of data stored on the client's side, while sessions are maintained on the server.

9. RESTful APIs:

HTTP is commonly used for building RESTful APIs (Representational State Transfer) that allow applications to interact with each other over the web. REST APIs use HTTP methods like GET, POST, PUT, and DELETE to perform actions on resources.

In summary, HTTP is the protocol that powers the World Wide Web, facilitating the transfer of web pages and resources between clients and servers. It operates on a request-response model, uses methods like GET and POST to interact with resources, and relies on headers, status codes, and URLs to manage communication. Secure versions like HTTPS provide encryption for data privacy, while HTTP forms the backbone of web communication.

Remaining topics: Number system and I/O devices.

Number system from book.

CO8: I/O devices:***

Topic 1:Keyboard and Mouse

Keyboard:

The computer keyboard is a peripheral device that was modelled after the old and traditional type-writer. As the computer gradually replaced the type writer the tele-printer style keyboard remained the main stream method of inputting text into computers to program, code and using in word processing softwares ever since the 1970s.

In the modern world, where technology permeates every aspect of our lives, the keyboard stands as a ubiquitous and indispensable tool. Whether we are typing a text message on our smartphones, composing an email on a computer, or even entering commands into a gaming console, the keyboard is our gateway to the digital realm. Its evolution over time has mirrored the growth of technology itself, making it an intriguing and vital component of our daily lives.

Keyboards work differently based on what kind of keyboard they are. The most common keyboard for a long time has been the membrane keyboard. While during the early days the keyboard was a very hard thing to type on and required fair amount of actuation force, now membrane keyboards, having conductive domes and PCB capacitive-based switches are used commonly. But in recent years the mechanical keyboards have became very popular among enthusiasts and gamers. These keyboards have individual switches and customizable actuation force for every key.

As for the software side of the keyboard, most keyboard uses a kid of key codes that are translated by the computer. This is done by recognising the layout of the keyboard and then interpreting the signals according to the position of the recognized keys. Popular keyboard layouts are ISO and ANSI. Most modern "full size" keyboard have the QWERTY layout and have 104 keys. Keyboard are practically inseparable part of computing and their only repayment maybe voice assisted input to speech-to-text.

Mouse:

The mouse is also a peripheral device of computers which are primarily used as pointer devices to interact with the UI of a computer more intuitively than a keyboard. Although they have been replaced by capacitive touch screens in the smartphone and portable devices arena, nevertheless they remain the most dominant pointing device in the computer realm.

The concept of the computer mouse was first introduced by Douglas Engelbart in the 1960s, as part of his groundbreaking work at the Stanford Research Institute. Engelbart's vision was to create a device that would simplify computer interactions, allowing users to move a cursor on the screen and perform tasks with greater ease. The first mouse was a wooden block with two perpendicular wheels, designed to detect motion in the X and Y axes.

Over the decades, the mouse has undergone numerous transformations, evolving into the sleek and ergonomic designs we use today. The traditional mechanical ball mouse, which required regular cleaning to maintain accuracy, has largely been replaced by optical and laser mice, which use sensors to track movement with precision. These modern mice offer smoother and more reliable performance, eliminating the need for mouse pads and frequent cleaning.

Now the mouse has evolve so mush so that there exists several kinds of mouse adapted to different use cases for people ranging from gamers to the regular office clerk. Some mouse have more than the standard 3 buttons ad a scroll wheel and can have unto 12 buttons with more buttons being added. This is to aid gamers have more buttons accessible to make games more immersive and interactive. The DPI of mouse can also be changed now to suit the user's preference and use cases. Wireless mouse also exist to give users the freedom to work from any distance from their workstations without being tethered by the wires.

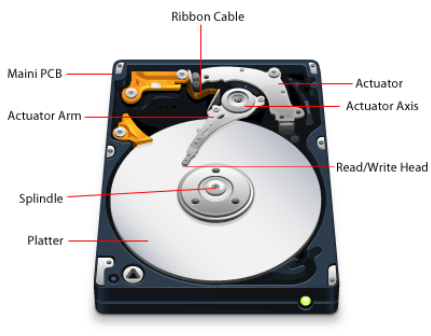
Topic 2: Storage Devices: Hard Disk and Flash Drives

Perhaps the most significant part of computers after their processing power is the capability to store and retrieve data. In fact storage is one of the 4 core parts of a computer. Without a storage device a modern computer will not even go through the POST (power on self test), the first part of booting a computer. Right now in the modern era, where digital information is imperative and essential, storage devices play a critical role in safeguarding, organizing, and providing access to our data. These devices come in various forms, from traditional hard disk drives (HDDs) and solid-state drives (SSDs) to more specialized options like USB flash drives, memory cards, and network-attached storage (NAS) solutions. Each type of storage device has its unique characteristics and applications, but collectively, they serve as the guardians of our digital world.

Hard Drive:

In the realm of digital storage, the hard disk drive (HDD) stands as one of the most venerable yet enduring technologies. Since its inception in the mid-20th century, the HDD has been a cornerstone of data storage and retrieval, becoming an essential component in personal computers, servers, and data centers worldwide. This device has gone through so many changes and improvements that it's astonishing that how much its capacity has increased. This iconic device has undergone remarkable advancements in capacity, speed, and reliability, and its role in preserving, accessing, and safeguarding our digital information is nothing short of pivotal.

Before hard drives the mode of data storage was in floppy discs. Those were very fragile and could hold very little data with the most advanced of them being able to hold only mere 1.44MB of data (IBM 1986). But the earliest hard drives could already hold megabytes of data with the ones near 2000s holding gigabytes. Not hard drives exists that can hold petabytes of data and are used in large data centres by the like of Google, Apple, Microsoft etc.



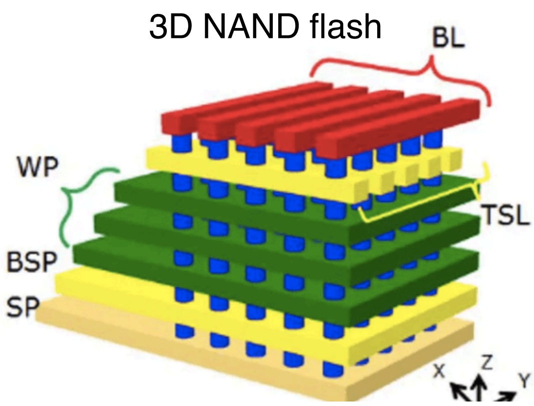
Hard drives use magnetic storage technology to hold data. The have one or more circular discs known as platter in them that spin 5,400 or 7,200 RPM based on their type and purpose. They also have an actuator head that moves very fast to and for the centre of the disc and a read/write head retrieves data or stores data using magnetic fields. The spindle motor rotates the platters and a Controller regulates the system and the flow of data. In order to compensate for the relatively slower speeds of the platter-actuator system now-a-days HDDs have a case inside them to make smaller write requests quicker by first speedy writing the data to the cashe first and then to the hard drive itself

Although the advancement of flash drives and the solid state drives have foreshadowed the practicality of the HDD, they till today remain one of the most revolutionary and cost effective methods of mass data storage to date.

Flash Memory:

Flash memory is a kind of electronically non-volatile storage medium. It is a storage technology that can be both written and erased unlike ROM(read only memory). And it is a like of solid state storage device that means there is not moving parts in it like that in hard drives. Its non volatile nature means that it can retain data when it is disconnected from power. Right now flash memory or solid state drive (SSD) is revolutionising digital data storage by providing the scope for higher reliability, redundancy and power efficiency.

Unlike traditional hard drives with spinning platters, flash memory relies on electronic cells that store data as electric charges. These cells are organized into a grid-like structure known as NAND flash, where multiple pages form blocks, enabling efficient data storage and retrieval. Data is read from and written to NAND flash memory in pages and blocks. A page is the smallest unit of data that can be read or written, typically ranging from 4 KB to 16 KB in size. Multiple pages are combined into a block, with typical block sizes ranging from 128 KB to 512 KB. One significant characteristic of NAND flash is that data can be read from a page without affecting other pages within the same block. This actually removes the scope for mechanical failure that is often possible in HDDs. Wear leveling, a feature embedded in the memory controller, ensures even wear across memory cells, extending the lifespan of flash memory devices. This innovation addresses one of the few drawbacks of flash technology: limited write/erase cycles per cell. As technology advances, newer iterations like 3D NAND have increased both capacity and longevity.



The other key strength of SSD or flash memory lies in its speed. It can read and write data at blazingly fast rates, significantly reducing data access times compared to traditional hard drives. This speed revolutionized the way we interact with technology, leading to quicker boot times, seamless multitasking, and rapid file transfers. PCIe gen 4 drives have a maximum sequential read/write speeds up to 7,450/6,900 MB/s (Samsung 990 pro) while gen 5 SSD have a speed of over 10000MB/s.

Flash memory's versatility extends to its various forms, from the ubiquitous USB flash drive to memory cards used in cameras and the robust SSDs revolutionizing storage in laptops and data centers. Flash memory's ability to store vast amounts of data in compact, lightweight packages has made it an indispensable tool for data portability and storage expansion.

Different kinds of flash memory drives

In the grand scheme of things flash memory had embedded itself into our lives. Today it spread into practically all kinds of IoT that require data storage. It has offered our computer systems to further progress with higher transfers speeds and performance. And it is going to keep improving through our efforts and the progress of technology.

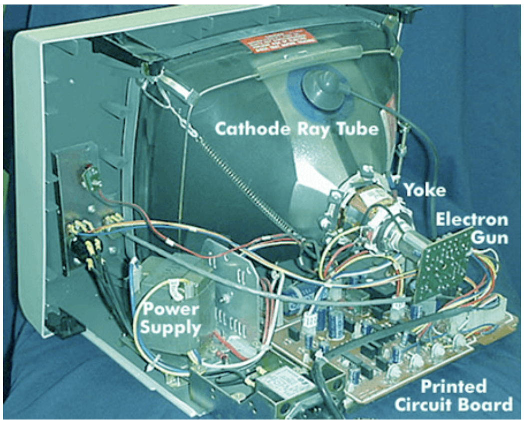
Topic 3: Display technologies:

Along with the invocations of I/O and storage systems came another technology that actually changed the way we spend out leisure time vastly besides contributing to business, education, medical care etc. This technology is the display technology. It was the first technology to allow humans to visualize something beyond the means of drawing on paper or cave walls. It allowed humans to interact with computers and exchange visual information. And the most prominent of the display technologies are CRT, LCD & LED.

CRT(Cathode Ray Tube)

The CRT is a vacuum tube-based technology that was developed in the late 19th century and reached its zenith of popularity in the mid-20th century. CRT displays functioned by using an electron gun to fire a stream of electrons at a phosphorescent screen, creating tiny points of light known as pixels. These electrons were precisely controlled to scan the screen line by line, illuminating pixels and creating images.

The story of CRT begins with the discovery of cathode rays. In the late 19th century, scientists like Sir William Crookes and Julius Plücker conducted experiments with vacuum tubes containing partially evacuated glass tubes. They observed a glow when an electric current was passed through the tube, which they called "cathode rays." In 1897, German physicist Karl Ferdinand Braun developed the first CRT, which he called the "Braun tube." His design included an electron gun that emitted electrons, a fluorescent screen that emitted light when struck by electrons, and an anode to accelerate the electrons. This early CRT produced a rudimentary form of visual display.Bell Telephone Laboratories introduced the first color CRT in 1928. This development paved the way for color television systems, although it would take several decades for color television to become widespread.



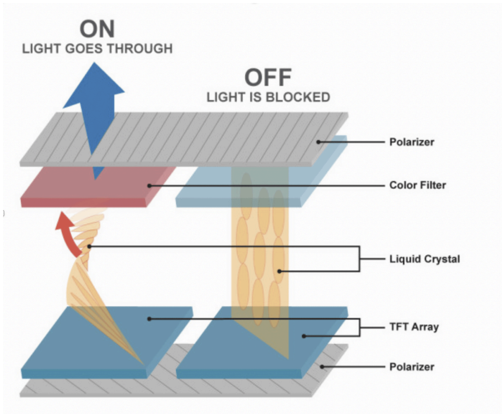
CRT monitors gave people the potential to visualise computer programs and code. It opened up doors to a whole new world and brought the world to a new new road of development. Because computers alone would to have been able to revolutionise the world this much. To this day it is a thing of awe to the civilization.

LCD(Liquid Crystal Display):

Liquid Crystal Displays, commonly known as LCDs, have emerged as one of the most ubiquitous display technologies in the modern world. These thin, energy-efficient screens are found in a wide range of devices, from smartphones and computer monitors to televisions and digital signage. The history and development of LCDs represent a remarkable journey through innovation and technological progress.

The concept of liquid crystals dates back to the late 19th century when Austrian botanist Friedrich Reinitzer first observed the unusual properties of a substance that could flow like a liquid while maintaining some crystalline characteristics. However, it wasn't until the 1960s that scientists began to explore the potential applications of liquid crystals in displays. In 1968, George H. Heilmeier and his team at RCA introduced the Twisted Nematic Field-Effect (TN) display. This marked a significant breakthrough in LCD technology. The TN-LCD utilized an electric field to control the alignment of liquid crystal molecules, allowing them to twist and block or transmit light, creating an image. The simplicity and low power consumption of TN-LCDs made them suitable for early calculators and digital watches. Throughout the 1970s and 1980s, researchers and engineers made innovations like Super Twisted Nematic (STN) and Thin-Film Transistor (TFT) technology to enhance the contrast, viewing angles, and response times of LCDs.

Color LCDs became more prevalent in the late 1990s, as improvements in color filtering and backlighting enabled vibrant, full-color displays. The 21st century witnessed the widespread adoption of LCD technology in consumer electronics, including smartphones, flat-screen televisions, computer monitors, and tablets. LCDs became thinner, lighter, and more energy-efficient, making them suitable for an array of portable devices.



LCD panel layers

LCDs marked the beginning of another great technology that provided further improvements to human-computer interaction and the progress of technology. And this was further improved by the advent of LED.

LED(Light Emitting Diodes):

Light-Emitting Diodes, or LEDs, represent a groundbreaking innovation in the field of lighting and display technology. These tiny, energy-efficient semiconductors have made their way into countless devices, illuminating our world in ways previously unimaginable. The history and development of LEDs are a testament to human ingenuity and their transformative impact on various industries.

The first practical LED was developed in 1962 by American engineer Nick Holonyak Jr. at General Electric. Holonyak's LED emitted visible red light and was made of gallium arsenide phosphide (GaAsP). This red LED marked the beginning of the LED revolution, as it demonstrated the feasibility of creating light with solid-state semiconductor materials. In the years following Holonyak's red LED, researchers worked to create LEDs that emitted different colors of light. In 1994, Isamu Akasaki, Hiroshi Amano, and Shuji Nakamura, working in Japan, succeeded in creating efficient blue LEDs using gallium nitride (GaN) materials. This breakthrough completed the RGB (red, green, blue) color spectrum, enabling the creation of white light through color mixing. The availability of blue LEDs paved the way for the development of white LEDs. White LEDs are created by combining blue LEDs with phosphor coatings that convert some of the blue light into other colors, including yellow. The mixture of blue and yellow light creates white light. White LEDs quickly found applications in lighting, from flashlights and automotive lighting to general illumination in homes and commercial spaces.

The 21st century has witnessed the LED revolution, with LEDs becoming the dominant technology for various applications. In addition to lighting, LEDs have revolutionized display technology. LED displays offer vibrant colors, high contrast ratios, and fast response times, making them ideal for applications ranging from digital signage and outdoor billboards to large-screen TVs and monitors. This color accuracy is particularly necessary in the film industry where proper colour representation is necessary for appropriate colour grading.


Apple Pro Display XDR

The story of LED technology is far from over. Ongoing research and development continue to push the boundaries of what LEDs can achieve. Innovations such as Mini-LED and Micro-LED are leading to even thinner and higher-resolution displays. This is being lead by notably Apple computers in creating their extreme dynamic range(XDR) displays. Additionally, smart lighting systems, which incorporate LEDs and connectivity, are transforming how we control and personalize lighting in our homes and workplaces.

In conclusion, the history of LED technology is a testament to human curiosity, ingenuity, and perseverance. LEDs have illuminated our world in countless ways, from lighting our homes and streets to enhancing our visual experiences through high-quality displays. As LED technology continues to evolve, it promises a brighter, more energy-efficient future for lighting and visual communication.

The End