**System Flowchart:**

A system flowchart visually represents the flow of data through a system.

It shows how data is input, processed, stored, and output, along with the devices or programs involved.

Audience: IT professionals, developers, system analysts

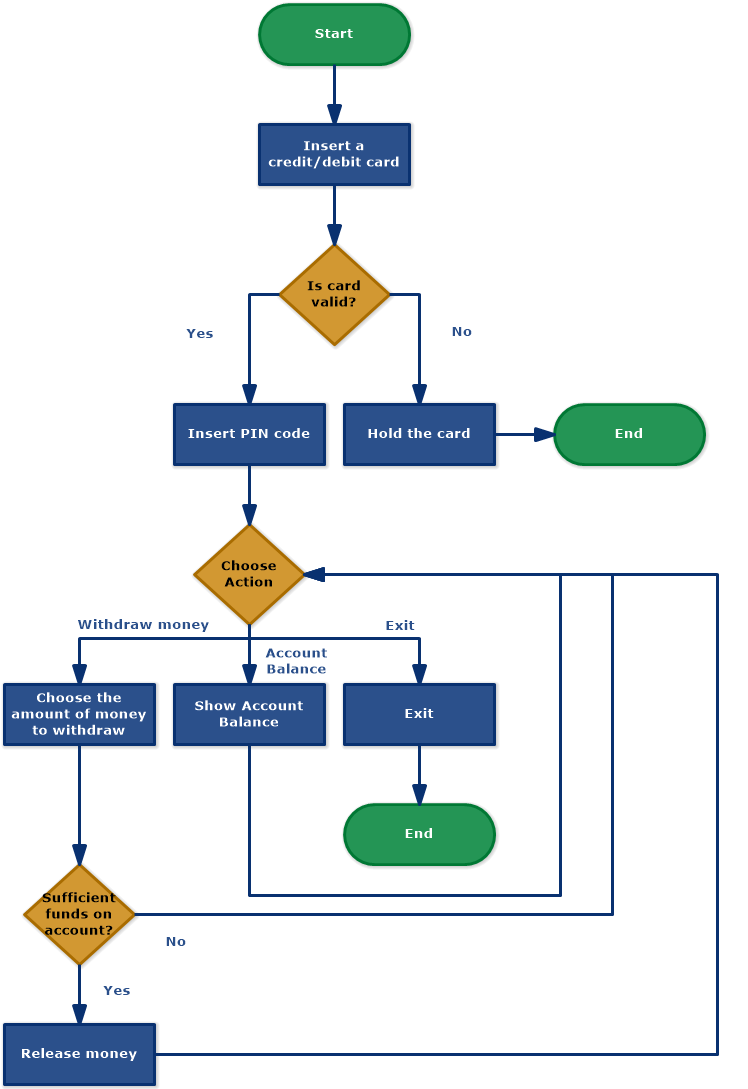
**Examples:**

**ATM System Flowchart**

Here's a simplified flow of an **ATM system**:

1. **User inserts card**
2. **System reads card info**
3. **Prompt for PIN**
4. **Verify PIN with database** (Decision: correct or not?)
5. **If correct: Show transaction menu**
6. **User selects withdrawal**
7. **System checks balance** (Decision)
8. **If sufficient: dispense cash & update database**
9. **Print receipt & eject card**

This process would be illustrated using symbols for **input (card/PIN)**, **decisions (PIN and balance checks)**, **database storage**, and **output (cash, receipt)**.



**User Login System**

**Steps:**

1. User enters username and password.
2. System sends credentials to server.
3. Server verifies credentials from database.
4. If valid, user is redirected to dashboard.

**System Flowchart Includes:**

**Input/Output Devices:** Keyboard, screen, scanner, printer

**Processing Steps:** System functions like database queries or logic operations

**Files/Databases:** Where data is stored or retrieved

**Decision Logic:** Based on data values or system rules

**Process flow charts:**

**Process flowcharts, or process diagrams, illustrate the steps involved in a particular process.**

Audience: Business analysts, managers, process owners

**Example**: Baking a cake



**Used in**: Manufacturing, service delivery, administrative processes.

**Employee Onboarding Process**

**Steps:**

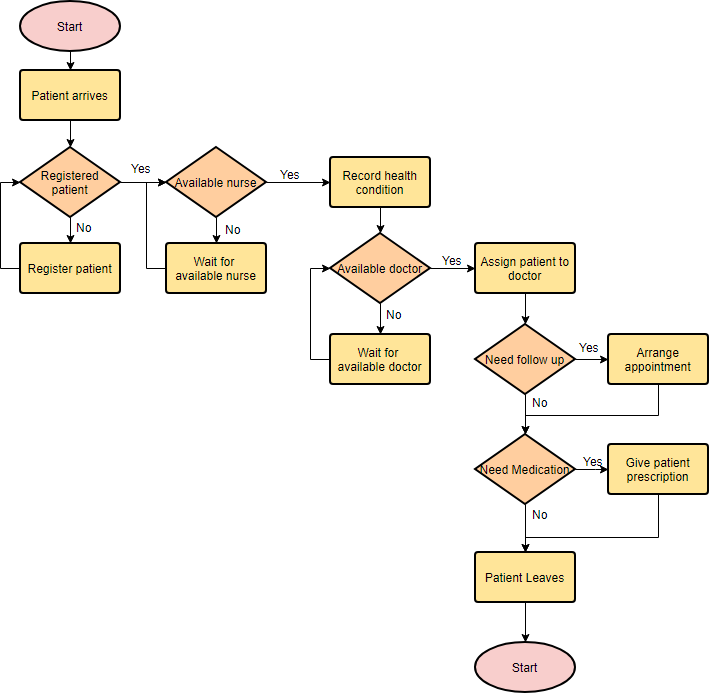
1. HR receives new hire info.
2. Create employee account.
3. Assign equipment.
4. Schedule orientation.
5. Supervisor assigns first task.

**Order Fulfillment Process**

**Steps:**

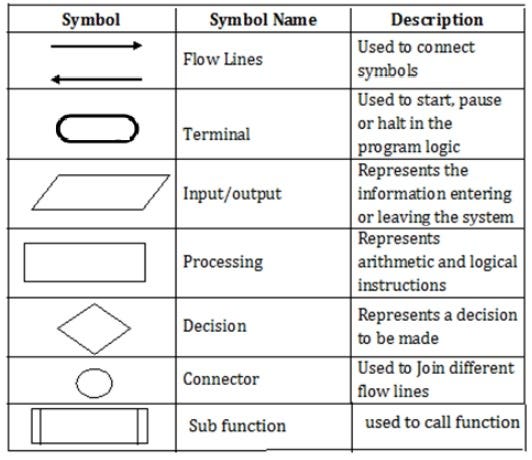
1. Customer places an order.
2. Inventory is checked.
3. Order is packed.
4. Order is shipped.
5. Tracking info is sent.

**Patient Management Process:**



A **Process Flowchart** is used to:

* Represent **steps in a business process**
* Show **decisions**, **actions**, and **flow of tasks**
* Focus on **what happens**, not how it's technically implemented



**Program Flowchart:**

A **Program Flowchart** shows the **detailed logic** of a program, typically focusing on how code executes step by step — including inputs, outputs, decisions, loops, and processes.

It’s more technical than a process or system flowchart and is usually used by **developers or technical analysts**.

Example:

1. **Start**
2. Input: Patient ID
3. Decision: Is Patient Registered?
   * No → Input Patient Details → Store in DB
4. Check Nurse Availability
   * No → Loop until available
5. Input Symptoms → Store
6. Check Doctor Availability
   * No → Loop until available
7. Assign Doctor
8. Input Diagnosis
9. Decision: Follow-Up Required?
   * Yes → Schedule Appointment
10. Decision: Medication Needed?
    * Yes → Print Prescription
11. End

**Difference:**

|  |  |
| --- | --- |
| **Process Flowchart** | Business steps & activities |

|  |  |
| --- | --- |
| **System Flowchart** | System-level data flow and components |

|  |  |
| --- | --- |
| **Program Flowchart** | Program logic and control flow |

**Where to Use?**

Flowcharts are widely used in real-world scenarios to streamline processes, improve communication, and support decision-making.

**1. Software Development**

* **Application**: Designing an algorithm for a login system.
* **Use**: Developers use flowcharts to visualize the logic behind user authentication—like checking username/password validity and handling failed login attempts.

**2. Healthcare**

* **Application**: Clinical decision support systems.
* **Use**: Doctors and nurses follow flowcharts for diagnosing illnesses or deciding treatment plans (e.g., when to administer antibiotics).

**3. Business Process Management**

* **Application**: Order fulfillment process.
* **Use**: Companies create flowcharts to document steps from order receipt to delivery, helping improve efficiency and reduce delays.

**4. Customer Support**

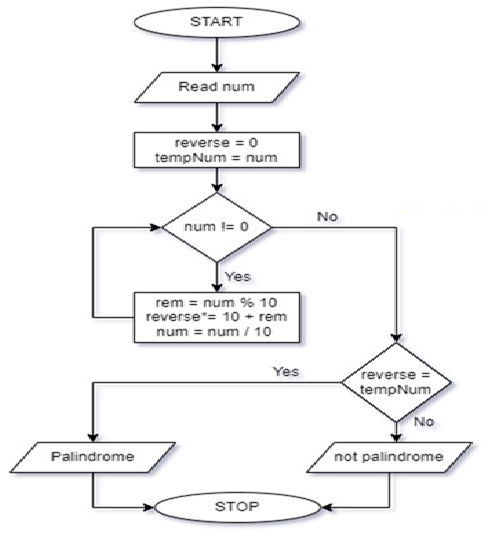
* **Application**: Call center troubleshooting scripts.
* **Use**: Agents follow decision trees (a type of flowchart) to guide conversations and solve customer issues step by step.

**6. Project Management**

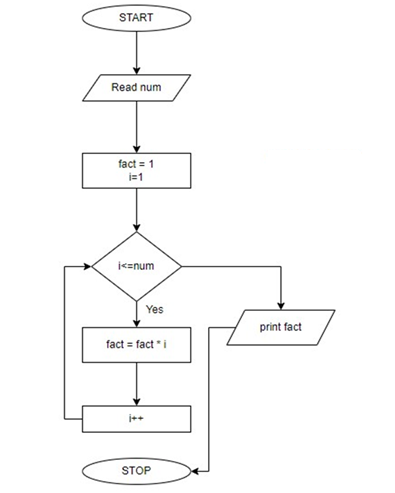
* **Application**: Task approval workflows.
* **Use**: Project managers design flowcharts to show how tasks progress through planning, review, approval, and execution.

**Program flow charts:**

**Palindrome:**



**Factorial of a number:**



# **Session 2: Algorithms**

**Algorithm to check if a given number is Palindrome or not**

Step 1: Start

Step 2: Read the input number from the user

Step 3: Declare and initialize the variable reverse and assign input to a temp variable tempNum=num

Step 4: Start the while loop until num !=0 becomes false

* rem = num % 10
* reverse\*= 10 + rem
* num = num / 10

Step 5 : Check if reverse == tempNum

Step 6: If it’s true then the number is a palindrome

Step 7: If not, the number is NOT a palindrome

Step 8: Stop

**Algorithm for finding the factorial of a number**

Step 1: Start

Step 2: Read the input number from the user

Step 2: Declare and initialize variables fact = 1 and i =  1

Step 4:  Repeat the loop until  i<=num

–  fact = fact \* i

–  i =  i++

Step 5:  Print fact to get the factorial of a given number

Step 6: Stop

**Session 3 - Pseudocode**

**Add Two Numbers**

Start

Input A

Input B

Sum ← A + B

Output Sum

End

**Find the Largest of Two Numbers**

Start

Input A

Input B

If A > B Then

Output A is larger

Else

Output B is larger

End If

End

**Check Even or Odd**

Start

Input N

If N mod 2 = 0 Then

Output "Even"

Else

Output "Odd"

End If

End

**Pseudocode for Linear Search**

Start

Input: List of elements A[1...n], Target value T

For i ← 1 to n do

If A[i] = T then

Output "Found at position", i

Stop

End If

End For

Output "Not Found"

End

**Pseudocode for Binary Search:**

Start

Input: Sorted array A[1...n], Target value T

Set low ← 1

Set high ← n

While low ≤ high do

mid ← (low + high) / 2

If A[mid] = T then

Output "Found at position", mid

Stop

Else If A[mid] < T then

low ← mid + 1

Else

high ← mid - 1

End If

End While

**Bubble Sort Pseudocode**

Start

Input: Array A of size n

For i ← 1 to n - 1 do

For j ← 1 to n - i do

If A[j] > A[j + 1] then

Swap A[j] and A[j + 1]

End If

End For

End For

Output: Sorted array A

End

**Matrix Addition:**

Start

Input: Matrix A of size m x n

Matrix B of size p x q

IF m ≠ p or n ≠ q THEN

OUTPUT "Matrices must have the same dimensions"

Exit or return

ELSE

Set <- Initialize matrix C of size m x n

For i from 0 to m - 1:

For j from 0 to n - 1:

C[i][j] = A[i][j] + B[i][j]

End For

End For

Output: Matrix C (sum of A and B) of size m x n

End

**Matrix Transpose**:

START

Input: Matrix A of size m x n

Initialize matrix T of size n x m

For i from 0 to m - 1:

For j from 0 to n - 1:

T[j][i] = A[i][j]

End For

End For

Output: Transposed Matrix T of size n x m

END

**Implementation:**

**Binary Search:**

int low = 0;

int high = arr.length - 1;

while (low <= high) {

int mid = (high+low)/2;

if (arr[mid] == element) {

return mid;

} else if (arr[mid] < element) {

low = mid + 1;

} else {

high = mid - 1;

}

}

return -1; // element not found

**Bubble Sort:**

int n = arr.length;

boolean swapped;

for (int i = 0; i < n - 1; i++) {

swapped = false;

for (int j = 0; j < n - 1 - i; j++) {

if (arr[j] > arr[j + 1]) {

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

swapped = true;

}

}

// If no two elements were swapped by inner loop, then break

if (!swapped) break;

