Introduction to Computing

WEEK 1

Course Overview

❖**Batch:** Fall 2024 (F24)

❖**Number of Credits:** 4 (3+1)

❖**Lecture:** 3 credit hours (2 hours lecture + 1 hour lecture in a week)

❖**Lab:** 1 credit hour (3 hours lab once in a week)

❖**Program Name:** Bachelors of Science in Computer Science (BSCS)

❖**Resource Person Contact Information**

- Hina Tahir
- **E-mail:** hina.tahir@ucp.edu.pk

Course Overview

- ❖The course will consist of
  - 8+8 Weeks
  - 32 Lectures (2+1 hours each)
  - 4 Assignments, 5 Quizzes
    - **(no retake of quizzes, Assignment late submission-Marks deduction)**
  - 2 Exams (Mid, Final)
- ❖Grading Criteria
  - Assignments: 15%
  - Quizzes: 20%
  - Class Participation 5%
  - Mid Term Exam: 20%
  - Final Term Exam: 40%

Course Overview

❖**Attendance:**

- Students must have minimum of **75-80%** attendance to appear in the terminal exam
- Attendance is strongly recommended since there will be **in-class quizzes.**

❖**Classroom Policy:**

- Students must be **punctual** for lectures
- Students must conduct themselves in an **orderly manner** while in class
- Before entering in the classroom make sure your **mobile phone is on Silent mode.**

❖**Cheating & Plagiarism Policy:**

- All the parties involved in **first cheating case will be awarded Zero** for that evaluation.
- Afterwards, all students involved in the **second instance will be awarded ZERO** in the **entire evaluation category.**
- Marks shall be **up-loaded on portal** and can be contested within a week or would be considered final.

Course Overview

❖**Rules and Regulation:**

- https://ucp.edu.pk/rules-regulations/

❖How to get a good grade? ***Understand, not memorize***

❖In order to do well in this course, most of you should plan on ***three-four hours a week*** reading book, preparing notes and doing homework assignments

❖**Expectations:** Attend class, do your assignments, ***avoid cheating/copying***, ask questions and participate!

Course Overview

❖**Recommended Textbooks**

- ***D.S. Malik*****.*****C++ Programming*****: From Problem Analysis to *****Program***** Design, Cengage Learning; 8th edition**

❖**Reference Textbooks**

- **C++ How to Program by Deitel and Deitel, 11th Edition, Prentice Hall Publications, 2007.**

Introduction to Computing

Course Objective

Students should achieve the following basic objectives:

- Should be able to have the know-how of computer system.
- Should be able to convert simple problems into algorithms
- Should be able to convert an algorithm into a C++ program
- Should be able to program in C++ to solve simple problems

Course Tools:

Microsoft Visual Studio

Course Learning Outcomes

CLO Statement
Understand fundamental problem-solving steps and logic constructs, by graphical representations for visualizing processes.
Solve basic problems using basics of selection statements, loops and fundamental concepts of programming language (C++).
Design and implement real-world problems using selection statements, loops, and one-dimensional arrays in C++.

Introduction to Computer Science

What is computer science?

“Everything that happens after you ask a question from Google until you get a result.”

(Lance Fortnow)

- It is a discipline that seeks to build a scientific foundation for computer design and information processing using computers

- Computer science is the study of
  - **What can be accomplished using computers, and**
  - **How to construct software to do these things**

Why study computers?

- A computer is a profoundly important technological device
  - Broadly impactful, Occasionally disruptive

- Computers have had impacts on the way we live, the way we think, and the way we do business
- But we are perhaps only 1/3 to 1/2 of the way through the process of absorbing the impact of computing in our lives

- Computers will have a substantial influence on any area of study you choose at UCP
- So, understanding computers is important

The Computer Defined

What is a computer?

The Computer Defined

What is a computer?

- The word computer comes from the word ***"compute",*** which means, ***"to calculate"***
- Thereby, a computer is an electronic device that can perform arithmetic operations at high
- A computer can be defined as a ***programmable electronic*** machine that accepts **input** (data), **processes** it and gives out **results** (information)
- It allows the user to store all sorts of data and then 'process' that data, or carry out actions with the data, such as calculating numbers or organizing words

"A computer is a machine for manipulating data according to a list of instructions known as a program"

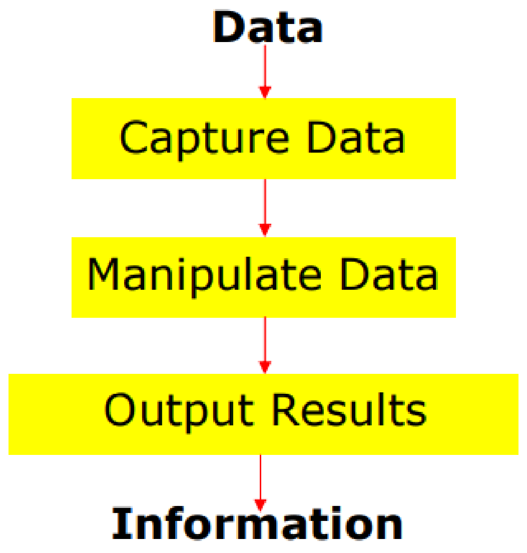
(Wikipedia)

The Computer Defined

A computer is also called a **data processor** because it can store, process, and retrieve data whenever desired

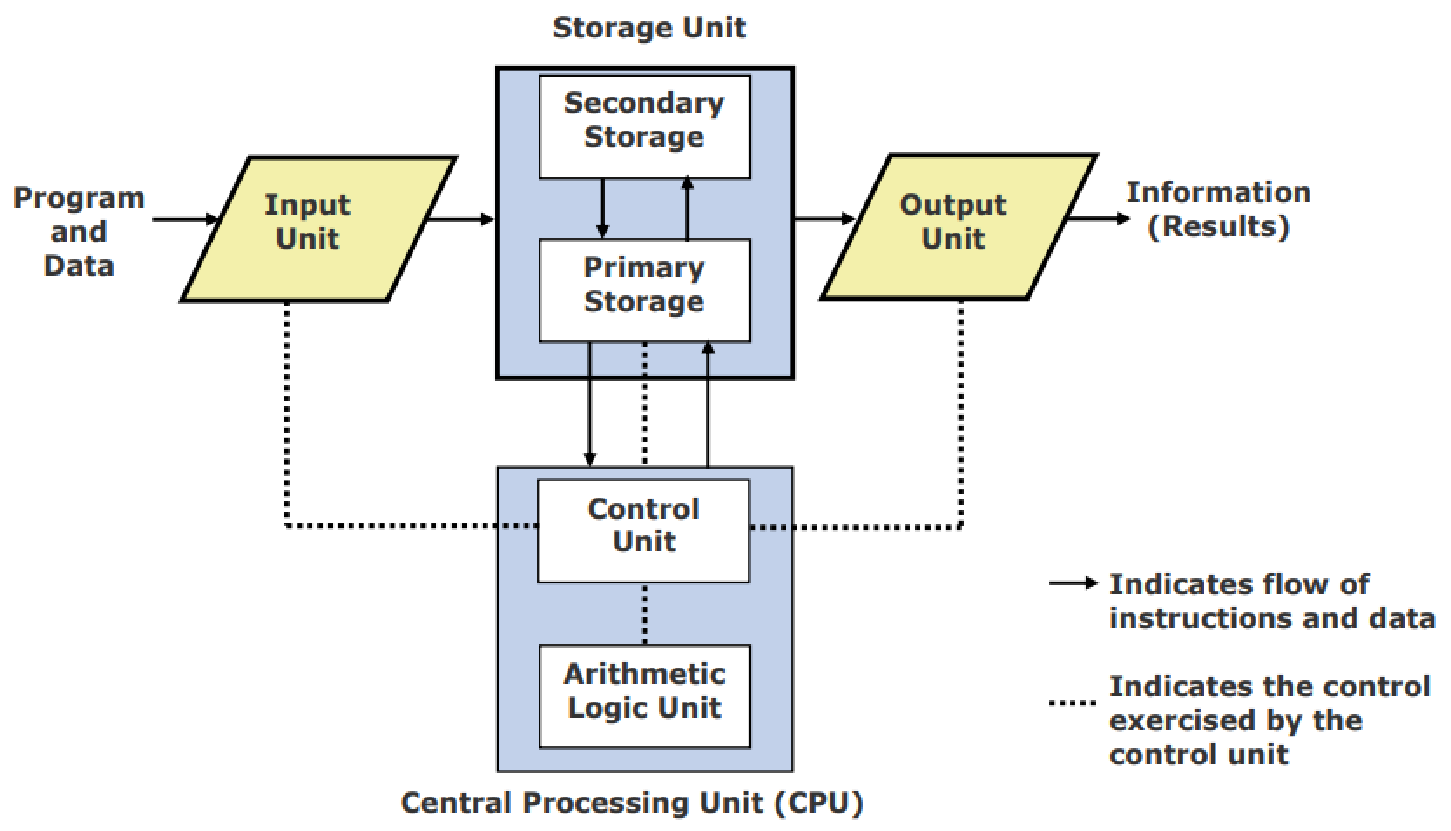
Data processing

The activity of processing data using a computer is called data processing



Data is raw material used as input and **information** is processed data obtained as output of data processing

Basic Organization of a Computer System



Input Unit

An input unit of a computer perform the following functions

- It accepts (or reads) instructions and data from outside world
- It converts these instructions and data in computer acceptable form
- It supplies the converted instructions and data to the computer for further processing

Examples:

Keyboard – Mouse - Game controller – Microphone - Touch screens - Touch sensitive pad - Biometric device - Card reader - Barcode reader – Scanner - Webcam

Output Unit

An output unit of a computer performs the following functions

- It accepts the results produced by the computer, which are in coded form and hence, cannot be easily understood by us
- It converts these coded results to human acceptable (readable) form
- It supplies the converted results to outside world

Examples:

- Monitors - LCD/LEDs - Touch screens – Printer – Speakers – Headphones – Projector - Interactive whiteboards

Storage Unit

A storage unit of a computer holds (or stores) the following,

- Data and instructions required for processing (received from input devices)
- Intermediate results for processing
- Final results of processing, before they are released to an output device

There are two types of storage

- Primary storage
- Secondary storage

Storage Unit

Primary Storage

- Also called RAM (Random Access Memory)
- Used to hold running program instructions, data and intermediate results
- Used to hold data, intermediate results, and results of ongoing processing of job(s)
- Fast in operation
- Small capacity
- Expensive
- Volatile (looses data on power dissipation)

Storage Unit

Secondary Storage

- Also called ROM (Read Only Memory)
- Used to hold stored program instructions and data
- Used to hold data and information of stored jobs
- Slower than primary storage
- Large capacity
- Lot cheaper that primary storage
- Retains data even without power

Secondary Storage Devices

- Magnetic Tape
- Magnetic Disk
- Optical Disk
- Flash Drive and Memory Cards

Central Processing Unit

CPU or Central Processing Unit is the brain of the computer

- Made of silicon and copper
- Carries out instructions from the program

CPU itself consists of

- **Arithmetic and Logic Unit** (ALU), **Control Unit** (CU), **Registers**

Arithmetic & Logic Unit is the place where the actual executions of instructions takes place

Control Unit manages and coordinates the operations of all other components of the computer

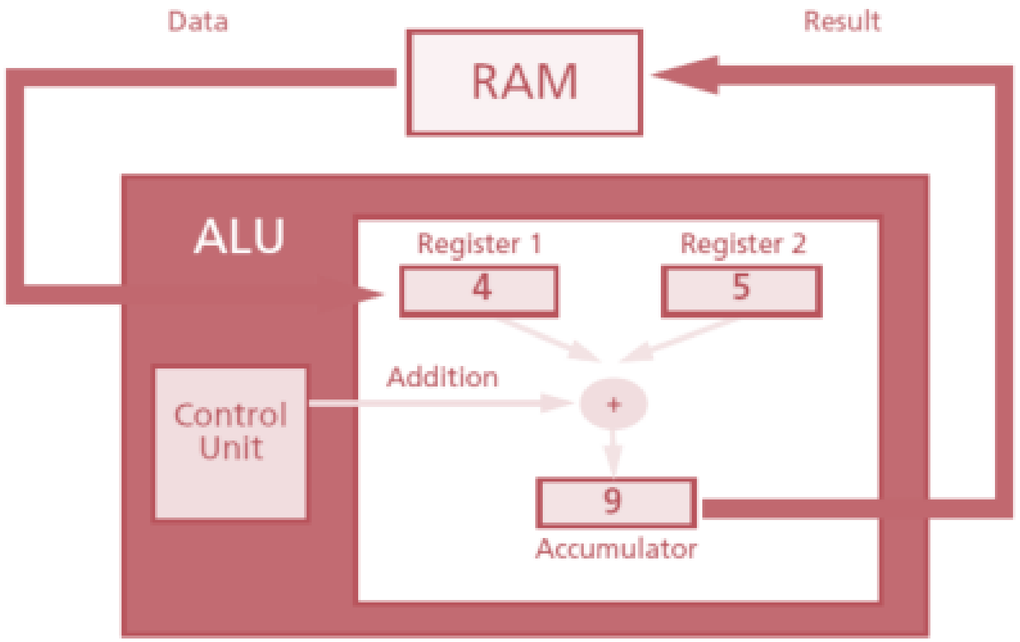
Registers are devices that hold data inside the computer's memory long enough to execute a particular function, such as indexing, calculating, sorting or otherwise manipulating data

- They are the CPU's own internal memory
- It stores location from where instruction was fetched

CPU Instruction Cycle

The CPU instruction cycle (machine cycle) has four steps

- **Fetch** - Retrieve an instruction from the memory
- **Decode** - Translate the retrieved instruction into a series of computer commands
- **Execute** - Execute the computer commands
- **Store** - Send and write the results back in memory



The System Concept

A system has following three characteristics:

- A system has more than one element
- All elements of a system are logically related
- All elements of a system are controlled in a manner to achieve the system goal

A computer is a system as it comprises of integrated components (input unit, output unit, storage unit, and CPU) that work together to perform the steps called for in the executing program

Hardware

Hardware: The physical parts of a computer

- Internal hardware
  - Located inside the main box (system unit) of the computer
- External hardware
  - Located outside the system unit
  - Connect to the computer via a wired or wireless connection
- There is hardware associated with all five computer operations

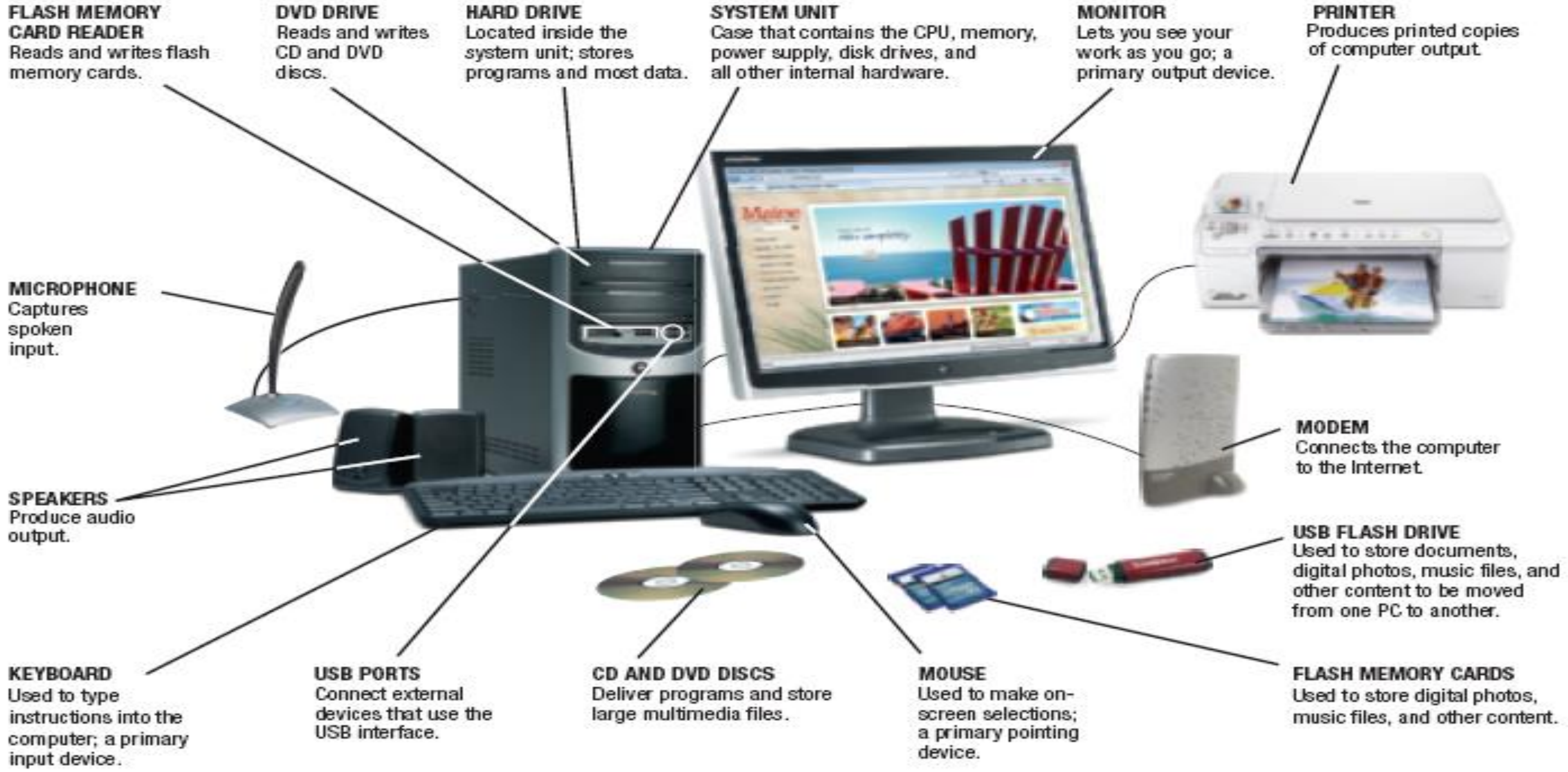

FLASH MEMORY CARD READER Reads and writes flash memory cards.
DVD DRIVE Reads and writes CD and DVD discs.
HARD DRIVE Located inside the system unit; stores programs and most data.
SYSTEM UNIT Case that contains the CPU, memory, power supply, disk drives, and all other internal hardware.
MONITOR Lets you see your work as you go; a primary output device.
PRINTER Produces printed copies of computer output.
MICROPHONE Captures spoken input.
MODEM Connects the computer to the Internet.
SPEAKERS Produce audio output.
USB FLASH DRIVE Used to store documents, digital photos, music files, and other content to be moved from one PC to another.
KEYBOARD Used to type instructions into the computer; a primary input device.
USB PORTS Connect external devices that use the USB interface.
CD AND DVD DISCS Deliver programs and store large multimedia files.
MOUSE Used to make on-screen selections; a primary pointing device.
FLASH MEMORY CARDS Used to store digital photos, music files, and other content.

FIGURE 1-9
Typical computer hardware.

Software

Software: The programs or instructions used to tell the computer hardware what to do

- **System software:** Operating system starts up the computer and controls its operation
  - Without OS computer cannot function
  - Boots the computer and launches programs at the user’s direction
- **Application software:** Performs specific tasks or applications

Computer to Fit every Need

Six basic categories of computers:

- Embedded computers
- Mobile devices
- Personal computers
- Midrange servers
- Mainframe computers
- Supercomputers

Computer to Fit every Need

Embedded computer

- Embedded into a product and designed to perform specific tasks or functions for that product
- Cannot be used as general-purpose computers

Mobile device

- A very small device with some type of built-in computing or Internet capability
- Typically based on mobile phones

Personal computer

- A small computer designed to be used by one person at a time
- Also called a microcomputer

Computer to Fit every Need

Notebook (laptop) computers

- Typically use clamshell design
- Tablet computers
  - Can be slate tablets or convertible tablets
- Netbooks
  - Small notebooks; rapidly growing type of PC
- Ultra-mobile PCs (UMPCs)
  - Handheld computers

Midrange server

- A medium-sized computer used to host programs and data for a small network
- Users connect via a network with a computer, thin client, or dumb terminal

Computer to Fit every Need

- Mainframe computer
  - Powerful computer used by several large organizations to manage large amounts of centralized data
  - Standard choice for large organizations, hospitals, universities, large businesses, banks, government offices
  - Also called high-end servers or enterprise-class servers
- Supercomputer
  - Fastest, most expensive, most powerful type of computer
  - Generally, run one program at a time, as fast as possible
  - Commonly built by connecting hundreds of smaller computers, supercomputing cluster

Embedded Computers

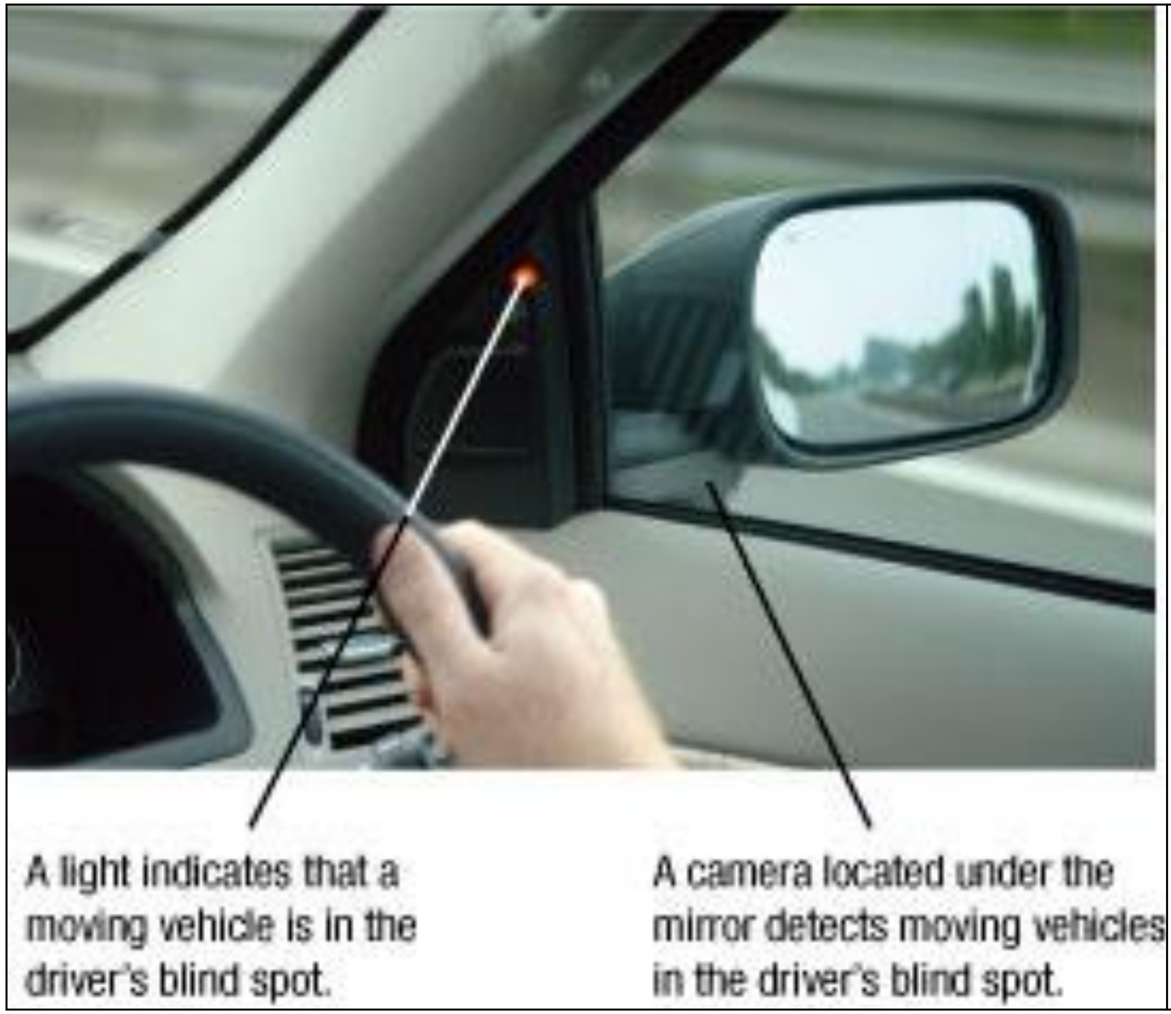

A light indicates that a moving vehicle is in the driver's blind spot.
A camera located under the mirror detects moving vehicles in the driver's blind spot.

FIGURE 1-12

Embedded computers. This car's embedded computers control numerous features, such as notifying the driver when a car enters his or her blind spot.

Midrange Server

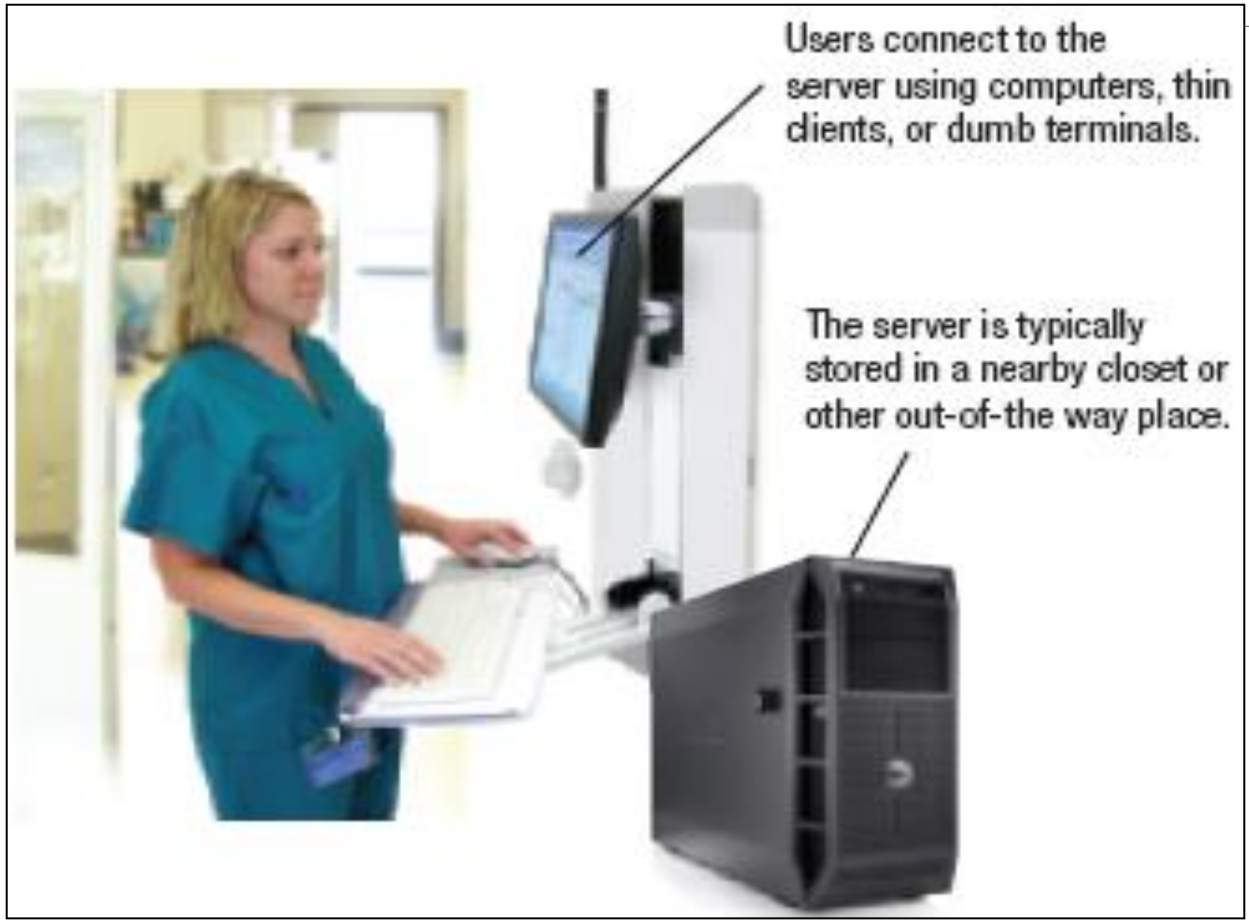
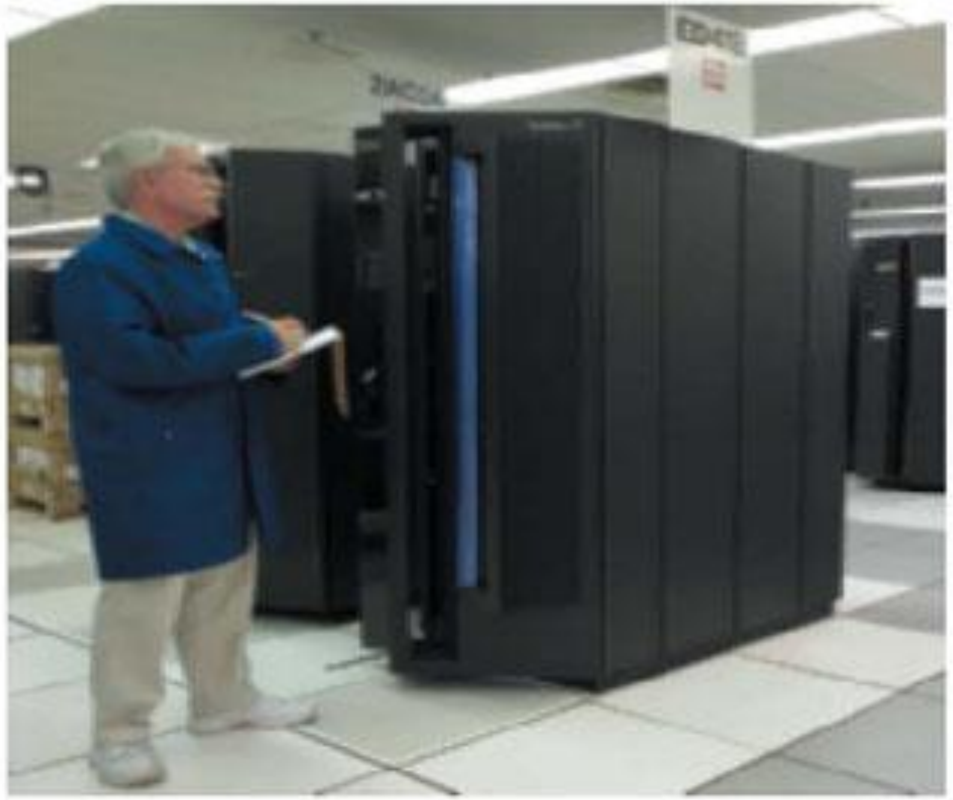

Users connect to the server using computers, thin clients, or dumb terminals.
The server is typically stored in a nearby closet or other out-of-the way place.

FIGURE 1-17

Midrange servers. Midrange servers are used to host data and programs on a small network, such as a school computer lab or medical office network.

Mainframe Computer

FIGURE 1-18
Mainframe computers.



Supercomputer

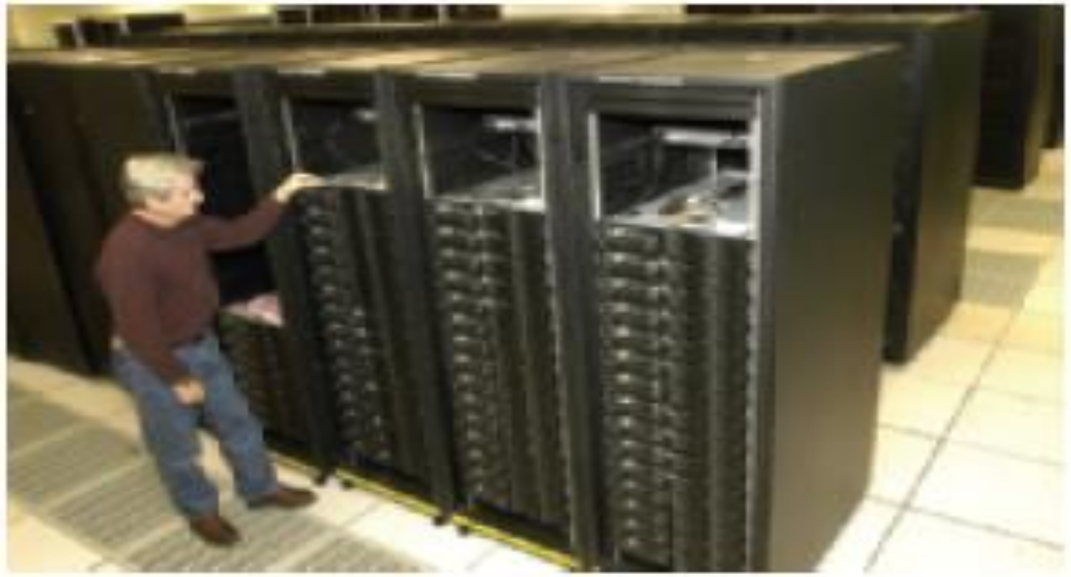


FIGURE 1-19
The Roadrunner supercomputer. Supercomputers are used for specialized situations in which immense processing speed is required.

Introduction to Problem Solving

What is a problem?

A problem is a situation that is unsatisfactory and causes difficulties for people.

What is Problem Solving?

Problem solving is the act of defining a problem; determining the cause of the problem; identifying, prioritizing, and selecting alternatives for a solution; and implementing a solution. The problem-solving process.

Steps/Stages of problem solving(PDLC)

- Problem Analysis
- Problem Design (Algorithm, Flowchart, pseudo Code)
- Program Coding
- Debugging(Errors) and Testing(Documentation)
- Implementation and maintenance

Steps/Stages of problem solving(PDLC)

- Problem Analysis
- Problem Design (Algorithm, Flowchart, pseudo Code)
- Program Coding
- Debugging(Errors) and Testing(Documentation)
- Implementation and maintenance

Problem Analysis

Problem analysis to determine both the **starting** and **ending points** for solving the problem

When determining the **ending point**, we need to describe the characteristics of a solution. In other words, ***how will we know when we're done***?

When determining the **starting point**, we need to describe what is data and what data are available? Is there any formula to solve the problem?

Problem: I need to send a birthday card to my brother, Mark.

Analysis: I don't have a card. I prefer to buy a card rather than make one myself.

Defining the Problem

A well-defined problem has **four components**:

- A clearly defined **initial situation** given.
- A clearly defined **goal**.
- A clearly defined **set of resources**.
- A clearly defined **constraints**.
- **Ownership**.

Case Study

You are at a **river** that you want to cross with all your goods. Your goods consist of a **chicken**, a **bag of grain** and your **cat**, **Rover**. You have to cross the river in your rowing boat but can only take **one passenger** with you at a time **– the chicken**, the cat or the **bag of grain**. You can't leave the **chicken alone** with the grain as the **chicken will eat the grain**. You can't leave your cat alone with the chicken as Rover will eat the chicken. However, you know that **Rover does not eat grain**. How do you get everything across the river intact?

Problem

Initial Situation: you, the chicken, the bag of grain and the cat are on one bank of a river with access to a rowing boat.

Resources: The rowing boat and your knowledge and problem-solving skills.

Constraints: You can take only one passenger, you must not leave Rover with the chicken, you must not leave the chicken with the grain.

Goal: You, the chicken, the bag of grain and the cat on the opposite bank of river.

Ownership: You will be involved in planning the solution and carrying it out.

Solution

Take the chicken across the river and leave it on the other side. Return to where you have left Rover and the grain.

Take Rover across the river and leave him on the other side. Take back the chicken.

Leave the chicken where you started. Take the bag of grain across the river and leave it with the Rover.

Go back and fetch the chicken and take it across the river with you.

Solving Problem

When faced with a problem:

1. We first clearly **define** the problem
2. Think of **possible solutions**
3. **Select** the one that we think is **the best** under the **prevailing** circumstances
4. And then **apply** that solution
5. If the solution works as desired, fine; else we **go back to step 2**

Solving Problem is Iterative Process

It is **quite common** to first solve a problem for a **particular/special case**

Then for **another**

And, possibly **another**

And watch for **patterns and trends that emerge**

And to use the knowledge form those patterns and trends in coming up with a **general solution**

Example

1 + 1 = 2

2 + 1 = 3

3 + 1 = 4

4 + 1 = 5

= 6, 7, 8…

in first 2, 3 iteration the child discover a pattern i.e. the answer is always one more than the previous answer. Despite added two numbers he discovered the trend or pattern.

That's how generally people work. 1st they solve the problem for a particular case, then another case and so on. And discover trends and patterns in the problem. And from those trends and pattern they make a general purpose solution. Which may be efficient.

If you want to **solve** a **problem**, it helps if you have **experienced** that **problem** or similar ones before.

Generally, there are **many ways** of solving a **problem**. The **experienced** people find the **method easily** to solve a problem. So again **experienced counts**.

The **process** or **set of processes** that can be used to **solve a problem** is termed as the **“algorithm”**.

Algorithm

Sequence of **steps**.

That can be taken to **solve** a **given problem**.

Steps:

When we divide a **task** into smaller task, we may call them steps.

Example: Walking

Algorithm:

Sequence of steps, these steps must be in **precise** sequence.

Example: making an egg.

Example

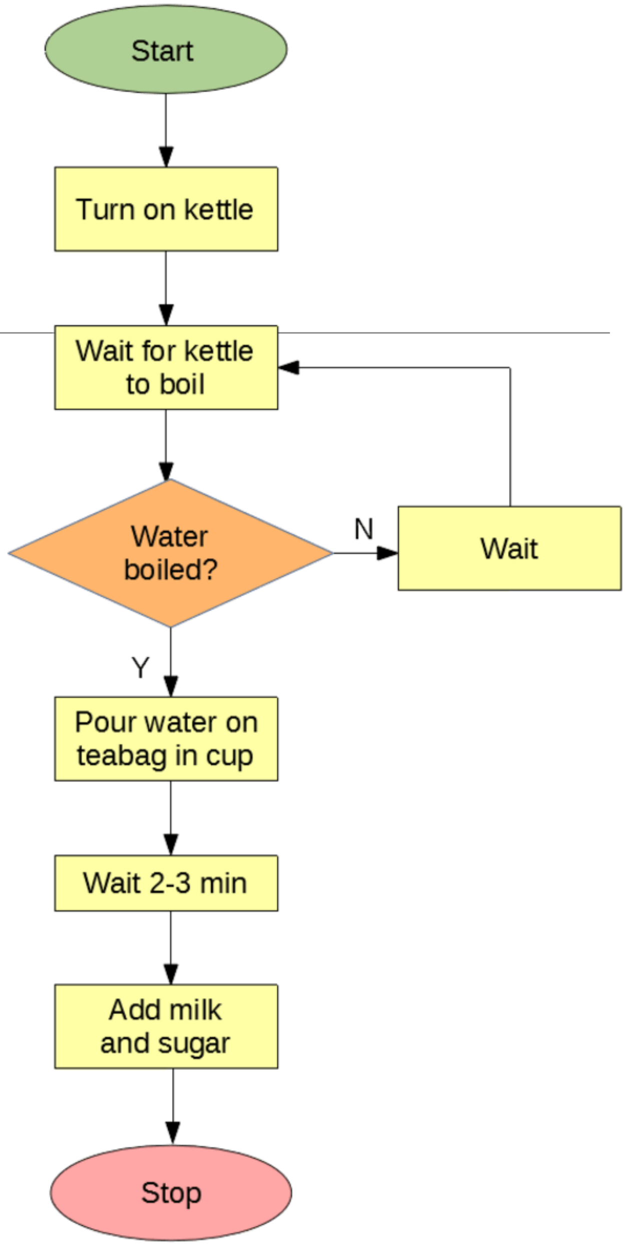
1) **Addition** (ADDING TWO NUMBERS)
2) Conversion from **decimal** to **binary**.
3) The process of **boiling an egg**.
4) The process of **mailing a letter**.
5) **Sorting** (Tallest person)
6) **Searching** (google follow an algorithm to search your appropriate page)

Let us write down the algorithm for a problem that is familiar to us

CLASS ACVTIVITIES

Problem: Make a cup of tea

1. Fill up the kettle with water.
2. Boil the kettle.
3. Place a teabag in your favourite mug.
4. Pour boiling water into your favourite mug.
5. Brew the tea for a few moments.
6. Remove and dispose of the teabag.
7. Add milk.
8. Add sugar.



Algorithm for a problem

Converting a **decimal** number into **binary**.

		Remainder
2	125	
2	62	1
2	31	0
2	15	1
2	7	1
2	3	1
2	1	1
	0	1

Algorithm in Structured English

1. Write the decimal number.

2. Divide by 2; write **quotient** and **remainder**.

3. Repeat **step 2 on the quotient**; **keep on** repeating until the quotient becomes zero.

4. Write all the **remainder** digits in the **reverse order (last remainder first)** to form the final result.

Points to Note:

1) The process consists of **repeated application** of **simple steps**.
2) All **steps** are **unambiguous** (clearly defined).
3) We are **capable** of doing all steps.(all steps are **executable**)
4) Only **finite number** of **steps** is need to be taken.
5) The **required result** will be **found**.
6) Moreover, the process will **stop at that point**.

Algorithm (Better Definition)

1st Definition:

Sequence of steps that can be taken to solve a problem.

Better Definition:

A ***precise sequence*** of a ***limited number*** of ***unambiguous***, ***executable*** steps that ***terminates*** in the form of a ***solution***.

THREE REQUIREMENT

1. **Sequence** is:
   a. Precise
   b. Consists of limited number of steps

2. Each **step** is:
   a. Unambiguous
   b. Executable

3. The **sequence** of **steps terminates** in the form of a **solution**.

Algorithm Representation

Generally, software developers represent them in one of three forms:

- **Pseudo Code**
- **Flowcharts**
- **Actual Code**

Pseudo Code

A Pseudocode is defined as a **step-by-step description** of an algorithm. Pseudocode does not use any programming language in its representation instead it uses the **simple English** language text as it is intended for **human understanding** rather than machine reading. Pseudocode is the intermediate state between an idea and its implementation(code) in a high-level language.

Pseudo Code

Language that is typically used for writing algorithms.

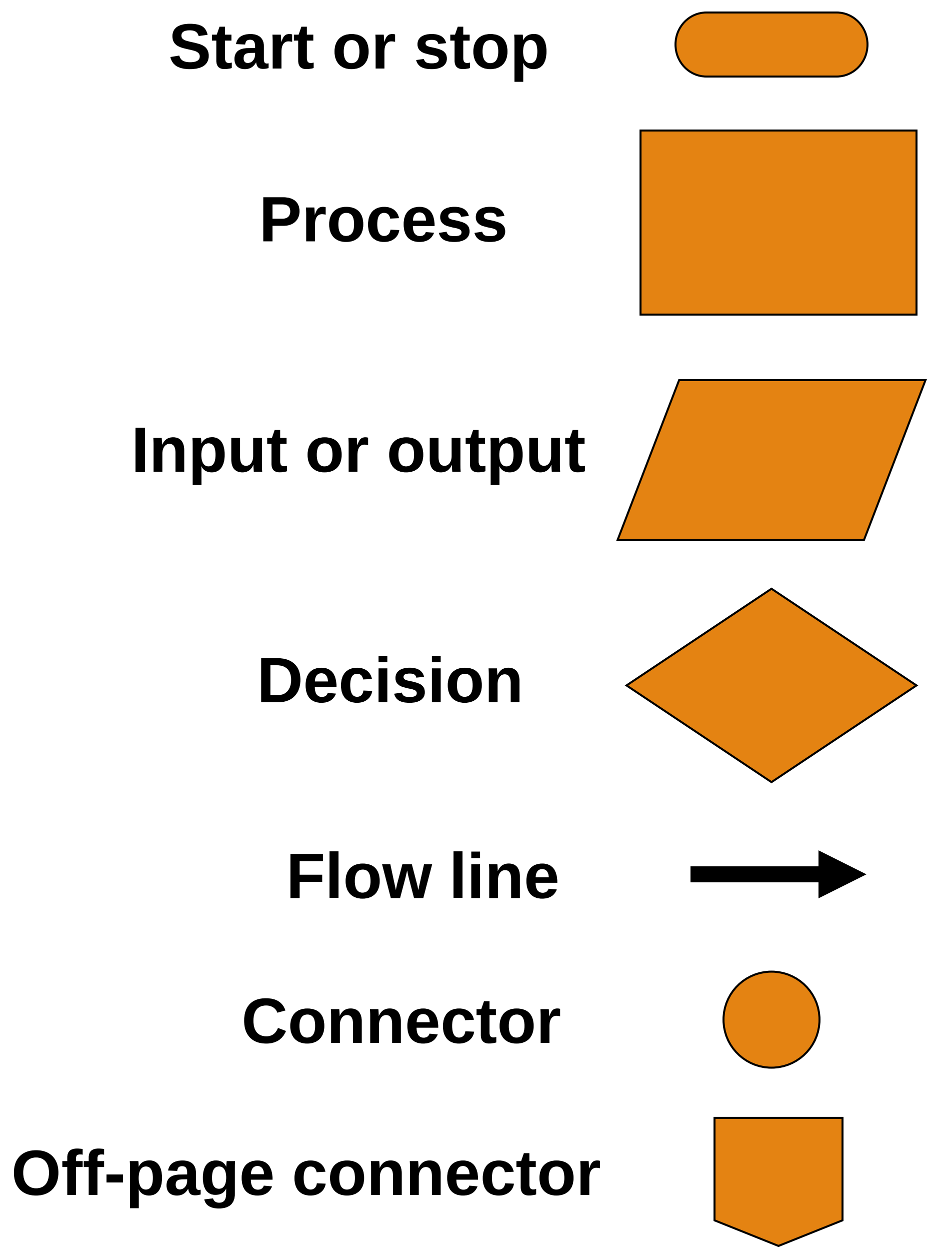
Similar to programming language, but not as rigid.

Plain English plus some formulas are used to write pseudo code.

Flowchart

A **graphical** representation of a **process** (e.g. an algorithm), in which **graphic objects** are used to indicate the **steps & decisions** that are taken as the **process** moves along from **start to finish**.

Flowchart Elements



Actual Code

Algorithm in the form of actual code written in any programming language like C++, Java etc.