

Introduction to programming and problem solving

October 4, 2023

Outline

- 1 Basic Computer Organisation
- 2 Generation of programming languages
 - First Generation
 - Second Generation:
 - Third Generation:
 - Fourth Generation
 - Fifth Generation
- 3 Algorithm
- 4 Flowchart
- 5 C Tokens

Introduction

- **Definition** It is an electronic device that accepts the data as input, processes the data in a pre-determined way and then communicates the result as output on the screen.

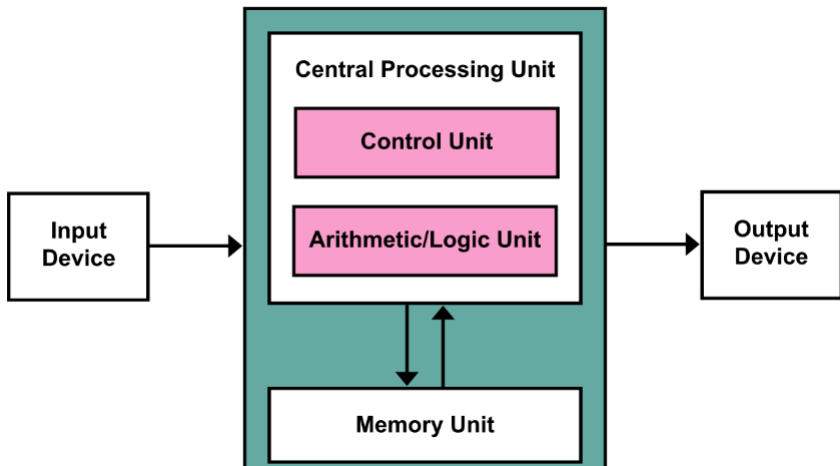
Characteristics I

-
- **Characteristics of Computer:**
- **Speed:** Executing mathematical calculation, a computer works faster and more accurately than human. Computers have the ability to process so many millions (1,000,000) of instructions per second. Computer operations are performed in micro and nano seconds. A computer is a time saving device. It performs several calculations and tasks in few seconds that we take hours to solve. The speed of a computer is measure in terms of GigaHertz and MegaHertz.
- **Diligence** A human cannot work for several hours without resting, yet a computer never tires.
- **Reliability** A computer is reliable. The output results never differ unless the input varies. the output is totally depend on the input.

Characteristics II

- **Automation** The world is quickly moving toward AI (Artificial Intelligence)-based technology. A computer may conduct tasks automatically after instructions are programmed.
- **Versatility** Versatility refers to a capacity of computer. Computer perform different types of tasks with the same accuracy and efficiency. A computer can perform multiple tasks at the same time this is known as versatility.
- **Memory** A computer can store millions of records.
- **Accuracy** When a computer performs a computation or operation, the chances of errors occurring are low.

Basic Computer Organisation



Memory Unit

- All the data that has to be processed or has been processed is stored in the memory unit.
- It transmits it to the required part of the computer whenever necessary.
- There are two types of memory **1) Primary Memory 2)Secondary Memory**

Primary Memory

Primary Memory

- This type of memory cannot store a vast amount of data.
- Therefore, it is only used to store recent data. The data stored in this is temporary.
- It can get erased once the power is switched off. Therefore, is also called temporary memory or main memory.
- RAM stands for Random Access Memory. It is an example of primary memory.
- This memory is directly accessible by the CPU. It is used for reading and writing purposes.
- For data to be processed, it has to be first transferred to the RAM and then to the CPU.

Secondary Memory

- For permanent storage purposes, secondary memory is used.
- It is also called permanent memory or auxiliary memory. The hard disk is an example of secondary memory.
- Even in a power failure data does not get erased easily.

Control Unit

- A control unit (CU) handles all processor control signals.
- It directs all input and output flow, fetches code for instructions and controls how data moves around the system.

Arithmetic and Logic Unit (ALU)

- The arithmetic logic unit is that part of the CPU that handles all the calculations the CPU may need, e.g. Addition, Subtraction and comparisons.
- It performs Logical Operations, Bit Shifting Operations, and Arithmetic operations.

Input and Output Devices

- The electromagnetic devices that accept data or a set of instructions from the outside world and then translate that data into machine-readable and understandable form are known as input devices.
- **Output Device:** Any peripheral that accepts data from a computer and prints, projects, or reproduces it is known as an output device.
- The output may be audio, video, hard copy – printed paper, etc.
- Output devices convert the computer data to human understandable form.

Software I

Software is a set of instructions, that is designed to perform a defined task, and it tells the computer how to work. Types of

Software: There are two broad categories of software 1) System Software
2) Application Software

System software is a set of computer programs that is designed to manage system resources.

System software acts as a platform for other software to work, such as antivirus software, OS, compiler, disk formatting software, etc.

Types of System Software:

- **Operating System** An Operating System is the most basic type of System Software that helps to manage computer hardware and software.
- **Programming translators** are the software that converts high-level language into machine language.

Software II

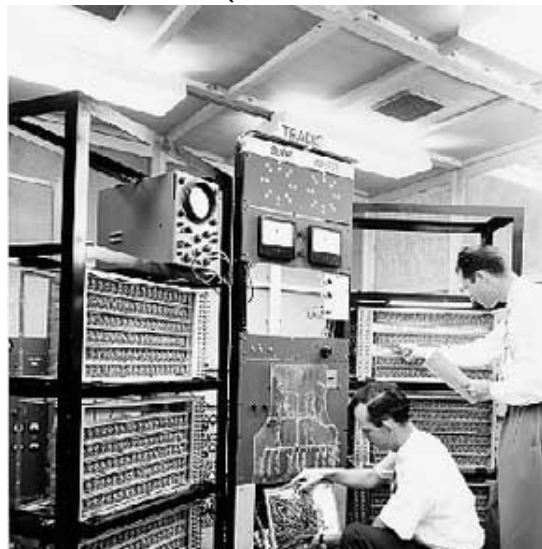
- **Device Drivers** are the types of system software that reduce the troubleshooting issues in your system.
- **Firmware Software** These are the operational software installed on the computer motherboards that help the operating system to identify the Flash, ROM, EPROM, EEPROM, and memory chips.
- **Utility Software** Utility software works as an interface between system software and application software. Utility software is a third-party tool designed to reduce maintenance issues and detect errors in the computer. Examples WinRAR, WinZip to reduce disk sizes.

Software III

- **Application software:** is a kind of software that performs specific functions for the end user by interacting directly with it. The sole purpose of application software is to aid the user in doing specified tasks. Web browsers like Firefox, and Google Chrome, as well as Microsoft Word and Excel, are examples of application software that is used on a personal computer or laptop.

Generations of Computer:

First Generation(1945-1956:



First Generation

- Thousands of vaccum tubes
- Consumes lot of power
- Generates large amount of heat
- large in size
- very expensive
- Used Machine language
- Punched cards and Paper tapes

Second Generation(1956-1963)

- Used Transistors
- In comparison to the first generation, the size of second generation was smaller.
- In comparison to computers of the first generation, the computing time taken by the computers of the second generation was lesser.

Third Generation 1964-1971

- Used Integrated Circuits
- Smaller and Faster
- Cheaper than second Generation
- The third generation computer consumed less power and also generated less heat.

Fourth Generation(1972 onwards)

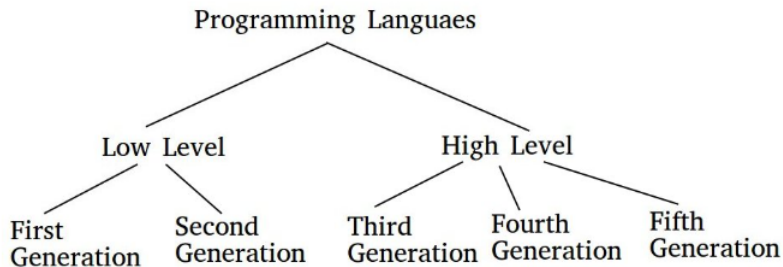
- Microprocessors with VLSI
- Thousands of IC on a single chip
- Small and POrtable
- Cheapest and work at high speed

Fifth Present and Future

- Fifth-generation computers, also known as modern computers, are still in the development stage and are based on artificial intelligence.
- In 1982, Japan was invented the FGCS (Fifth Generation Computer System).
- Computers of this generation are based on microelectronic technology with high computing power and parallel processing.

Generation of programming languages

A programming language is a vocabulary and set of grammatical rules for instructing a computer or computing device to perform specific tasks.



First Generation

- The first-generation languages are also called machine languages/ 1G languages.
- This language is machine-dependent.
- The machine language statements are written in binary code (0/1 form) because the computer can understand only binary language.

Advantages:

- Fast & efficient as statements are directly written in binary language.
- No translator is required.

Disadvantages:

- Difficult to learn binary codes.
- Difficult to understand – both programs & where the error occurred.

Second Generation:

- The second-generation languages are also called assembler languages/ 2G languages.
- Assembly language contains human-readable notations that can be further converted to machine language using an assembler.
- In the assembly language, symbolic names are used to represent the opcode and the operand part of the instruction.

Advantages:

- It is easier to understand if compared to machine language.
- Modifications are easy
- Correction & location of errors are easy.

Disadvantages:

- Assembler is required.
- This language is architecture /machine-dependent, with a different instruction set for different machines.

Third Generation

- The third generation is also called procedural language /3 GL.
- It consists of the use of a series of English-like words that humans can understand easily, to write instructions.
- It's also called High-Level Programming Language.
- For execution, a program in this language needs to be translated into machine language using a Compiler/ Interpreter.
- Examples of this type of language are C, PASCAL, FORTRAN, COBOL, etc.

Advantages:

- Use of English-like words makes it a human-understandable language.
- Lesser number of lines of code as compared to the above 2 languages.
- Same code can be copied to another machine & executed on that machine by using compiler-specific to that machine.

Disadvantages:

- Compiler/ interpreter is needed.
- Different compilers are needed for different machines.

Fourth Generation

- The fourth-generation language is also called a non-procedural language/ 4GL
- They are often used to build database systems, create user interfaces, and generate reports.
- These are languages that consist of statements that are similar to statements in the human language.
- Examples of these languages are Perl, Python, Ruby, SQL, and MatLab(MatrixLaboratory).

Advantages:

- Easy to understand & learn.
- Less time is required for application creation.
- It is less prone to errors.

Disadvantages:

- Memory consumption is high.
- Less flexible.

Fifth Generation

- A fifth-generation programming language is based on solving using constraints given to the program rather than using an algorithm written by a programmer. i.e., we make computers learn to solve any problem.
- It very closely resembles human speech. Examples are PROLOG, Mercury, OPS5, AI, etc.

Advantages:

- Machines can make decisions.
- Programmer effort reduces to solve a problem.
- Easier than 3GL or 4GL to learn and use.

Disadvantages:

- Complex and long code.
- More resources are required & they are expensive too.

The best program ever developed in the 5th generation is the Sofia robot which exactly looks like humans and is more intelligent than humans.

An algorithm is a step-by-step procedure or set of instructions designed to solve a specific problem or perform a particular task.

Characteristics of an algorithm

- **Input:** An algorithm takes zero or more inputs, which are the initial data or information on which it operates to produce a result. These inputs are well-defined and specified.
- **Output:** An algorithm produces at least one output or result based on the provided inputs. The output represents the solution to the problem.
- **Finiteness:** An algorithm must be finite, meaning it has a finite number of well-defined steps or instructions. It will eventually terminate, providing an output. It should not run indefinitely.
- **Definiteness:** Each step or instruction in an algorithm must be clear, unambiguous, and precisely defined. There should be no ambiguity or confusion about how to perform each step.
- **Effectiveness:** An algorithm is designed to solve a specific problem or perform a particular task effectively. It must lead to the correct and desired result when executed correctly.

Algorithm to calculate area of circle

Step 1. Start

Step 2. Read R

Step 3. Let $PI=3.14$

Step 4. Calculate $area=PI*R*R$

Step 5. Print area

Step 6. Stop

Algorithms Exercise

- 1 Sum of two numbers
- 2 Generate simple interest
- 3 swap two numbers
- 4 salary of an employee $\text{Gross Salary} = \text{Basic Pay} + \text{DA} + \text{HRA}$
- 5 largest of two numbers

Algorithm for Largest of two numbers

Step 1: Start

Step 2: Read number a

Step 3: Read number b






Step 4: if $a > b$, print a (Compare a and b using greater than operator)

Step 5: else print b

Step 6: Stop

flowchart

A flow chart is a **graphical or symbolic representation of an Algorithm.**

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

pseudocode

Pseudocode is an informal way of writing a program. It is not exactly a computer program.

It represents the algorithm of the program in natural language and mathematical notations.

Usually, there is no particular code syntax to write a pseudocode.

Pseudo code for Area of Rectangle

AreaOfRectangle()

Begin

Read: width, length;

Set $\text{area} = \text{width} * \text{length}$;

Print area;

End

History of C



C programming language was developed in 1972 by Dennis Ritchie at bell laboratories of AT&T (American Telephone & Telegraph), located in the U.S.A.

Dennis Ritchie is known as the founder of the c language.

It was developed to overcome the problems of previous languages such as B, BCPL, etc.

Initially, C language was developed to be used in UNIX operating system.

It inherits many features of previous languages such as B and BCPL.

Features of C

- 1 Procedural Language
- 2 Fast and Efficient
- 3 Modularity
- 4 Statically Type
- 5 General-Purpose Language
- 6 Rich set of built-in Operators
- 7 Libraries with Rich Functions
- 8 Middle-Level Language
- 9 Portability

Basic structure of a C Program

Documentation section

Link section

Definition section

Global declaration section

main () Function section

{

Declaration part

Executable part

}

Subprogram section

Function 1

Function 2

.....

.....

Function n

(User defined functions)

Structure of C Program I

Documentation Section: Consists of the description of the program, programmer's name, and creation date. These are generally written in the form of comments.

Link Section: All header files are included in this section which contains different functions from the libraries. A copy of these header files is inserted into your code before compilation.

Definition Section: we define different constants. The keyword define is used in this part.

Global Declaration: All the global variables used are declared in this part. The user-defined functions are also declared in this part of the code.

main function

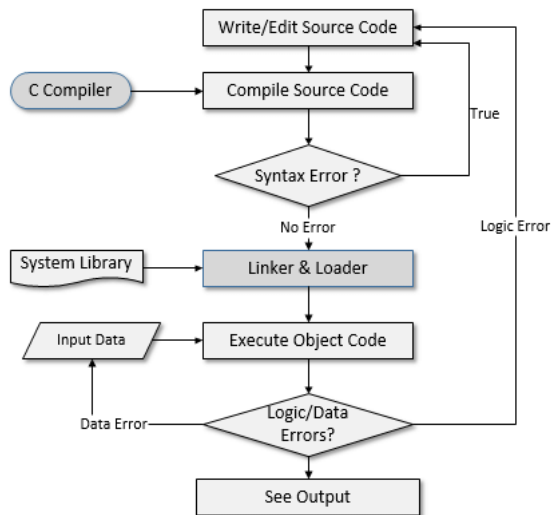
- A C program must have one main function section in its structure.
- This section contains two parts; declaration part and executable part.

Structure of C Program II

- **Declaration part:** The declaration part declares all the variables used in the executable part.
- **Executable part:** There is at least one statement in the executable part. These two parts must appear between the opening and closing braces of the main function.
- The program execution begins at the opening brace and ends at the closing brace.
- All statements in the declaration and executable part end with a semicolon.

Sub Program Section All the user-defined functions are defined in this section of the program.

Compiling and executing a program



C Tokens

A token in C can be defined as the smallest individual element of the C programming language that is meaningful to the compiler.

- 1 Keywords
- 2 Identifiers
- 3 Constants
- 4 Strings
- 5 Special Symbols
- 6 Operators

Keywords

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

- Pre-defined or reserved words in a programming language.
- Each keyword is meant to perform a specific function in a program.
- They can't be used as variable names
- C language supports 32 keywords

Identifiers

Identifier refers to name given to entities such as variables, functions, structures etc.

Rules for Naming Identifiers

- They must begin with a letter or underscore(_).
- They must consist of only letters, digits, or underscore. No other special character is allowed.
- It should not be a keyword.
- It must not contain white space.
- It should be up to 31 characters long

Constants

The constants refer to the variables with fixed values.

They are like normal variables but with the difference that their values can not be modified in the program once they are defined.

2 ways to define constant in C

const keyword

define preprocessor

Constant	Example
Decimal Constant	10, 20, 450 etc.
Real or Floating-point Constant	10.3, 20.2, 450.6 etc.
Octal Constant	021, 033, 046 etc.
Hexadecimal Constant	0x2a, 0x7b, 0xaa etc.
Character Constant	'a', 'b', 'x' etc.
String Constant	"c", "c program", "c in javatpoint" etc.

Table: Types of constants

Strings, Special Symbols and operators

Strings are nothing but an array of characters ended with a null character ('').

This null character indicates the end of the string. Strings are always enclosed in double quotes.

Whereas, a character is enclosed in single quotes in C and C++.

Special Symbols

- Brackets[]
- Parentheses()
- Braces
- Comma ,
- Semicolon
- Asterisk (*)

Operators Operators are symbols that trigger an action when applied to C variables and other objects.

The data items on which operators act are called operands.

Variables

A variable is a name of the memory location.

It is used to store data.

Its value can be changed, and it can be reused many times. **variable**

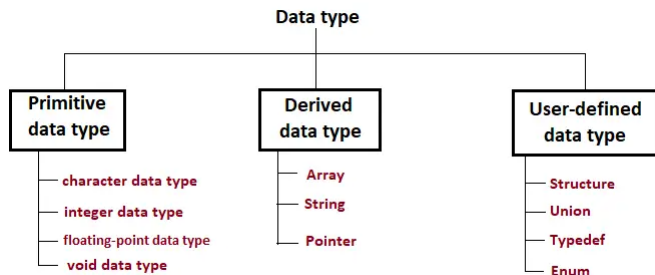
declaration: *datatype variablename;*

variable initialization: *variablename=value;*

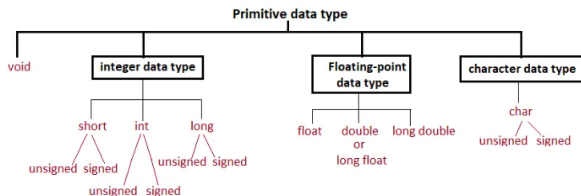
variable declaration and definition *datatype variablename=value;*

Datatypes

Each variable in C has an associated data type.
It specifies the type of data that the variable can store.
Each data type requires different amounts of memory
The data type is a collection of data with values.



Primitive Datatypes in C



Data Type	Memory (bytes)	Range	Format Specifier
short int	2	-32,768 to 32,767	%hd
unsigned short int	2	0 to 65,535	%hu
unsigned int	4	0 to 4,294,967,295	%u
int	4	-2,147,483,648 to 2,147,483,647	%d
long int	4	-2,147,483,648 to 2,147,483,647	%ld
unsigned long int	4	0 to 4,294,967,295	%lu
long long int	8	$-(2^{63})$ to $(2^{63})-1$	%lld
unsigned long long int	8	0 to 18,446,744,073,709,551,615	%llu
signed char	1	-128 to 127	%c
unsigned char	1	0 to 255	%c
float	4		%f
double	8		%lf
long double	12		%Lf

Operators

Operators in C

	Operators	Type
Unary Operator →	++, --	Unary Operator
Binary Operator {	+, -, *, /, %	Arithmetic Operator
	<, <=, >, >=, ==, !=	Relational Operator
	&&, , !	Logical Operator
	&, , <<, >>, ~, ^	Bitwise Operator
	=, +=, -=, *=, /=, %=	Assignment Operator
Ternary Operator →	?:	Ternary or Conditional Operator

Operators in C