THE UNIVERSITY OF UTAH
ASIA CAMPUS
SOUTH KOREA

Department of Electrical and Computer Engineering

Spring 2022

Accelerated Object Oriented Programming (CS 1420)

1. Introduction to Programming

Habtamu Minassie
Habtamu.aycheh@Utah.edu

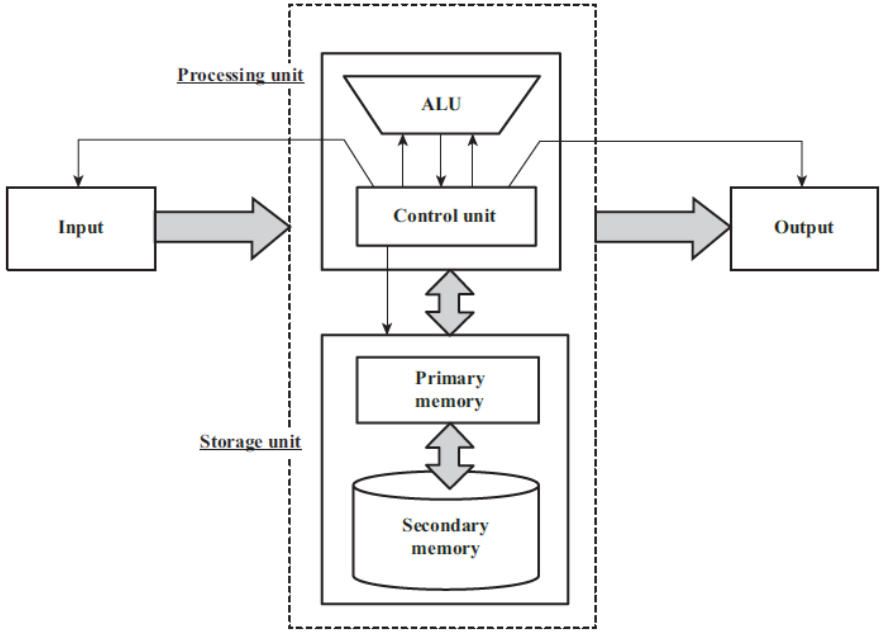
Computer and programs

- A modern computer can be defined as “a machine that stores and manipulates information under the control of a changeable program.”

We live in a highly connected world …

What is a computer program?

- A program is a set of instructions, which are followed by the machine so as to generate a desired output
  - This means that writing a computer program is giving instructions to a processor, so as to delegate a particular job to the hardware of the computer system



'Data' and 'instructions' are given as input to the system so as to perform a particular operation.

'instruction' represents the operation to be performed

'data' represents the information over which an operation is to be performed

Programming Languages

- A programming language is a computer language that is used by programmers (developers) to communicate with computers.
  - Every structure in programming language has a precise form, called syntax
  - Every structure in programming language has a precise meaning, called semantics.
- Thus, we need to understand the **syntax** and the **semantics** of a programing language to develop a software using the language.
- The process of creating this software is called programming

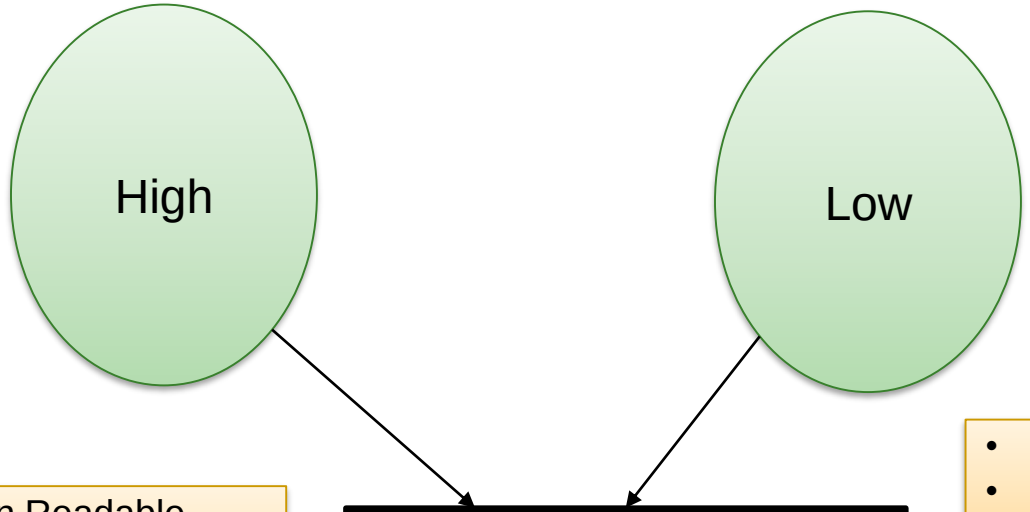
Translation

- As the hardware of a computer system is ultimately built up of electronic and semiconductor devices, it can only understand a binary language of 0's and 1'
- It is impractical for us to write programs in machine language as this will involve a detailed study on the circuitry over which the system is built upon as every machine is built up using different hardware technologies and circuitry.
- The language we use to write computer programs is called **high-level language** whereas the language that only machines understand is called as **a low-level language**.
- Thus, there is a need of a **translator** that can translate a high-level language into the 'machine language'.

Compiler vs Interpreter

- Both are system software that translates high level language to low level language (machine code)
- **Compiler**
  - The source program is translated into machine language all at once
  - Compiled programs generally run faster since the translation of the source code happens only once
  - Less portable - platform dependent
  - Example : c, c++,Fortran, …
- **interpreter**
  - An interpreter analyzes and executes the source code instruction by instruction
  - The source code and interpreter are needed each time the program runs
  - Slow
  - more portable
  - Example: python, java script,…

High Level vs Low level Programming Language



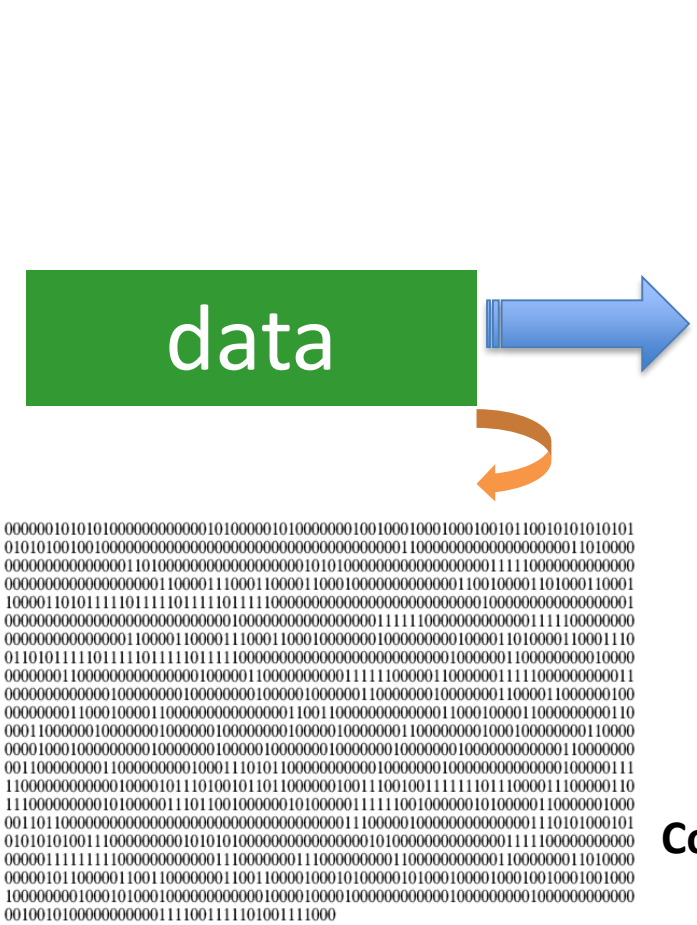
- More human Readable
- Less effort
- Slow

Python
Java
Java script
C/c++
Fortran
PHP
Ruby
…

- More machine Readable
- More effort
- Fast

Assembly Language

Data Representation



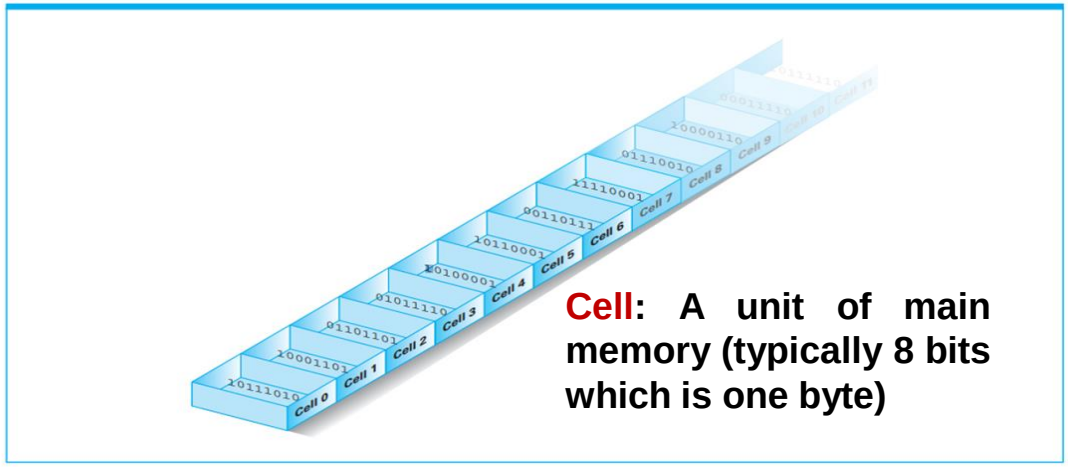
- Number
- Text
- Image
- Audio
- Video

Computer see everything in binary (0 and 1)

Note: The only language understood by computer hardware is machine language

Main Memory

- A program's instructions must be stored in main memory before they can be executed
- A primary storage is where programs and data are stored temporarily during processing



Cell: A unit of main memory (typically 8 bits which is one byte)

Memory

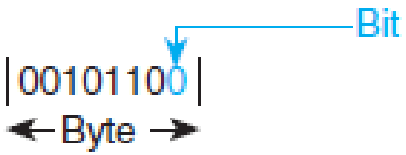
Address	Contents
0	–27.2
1	354
2	0.005
3	–26
4	H
.	.
.	.
.	.
998	X
999	75.62

Address: A "name" that uniquely identifies one cell in the computer's main memory

- Computer's main memory is organized as individual, addressable cells, the cells can be accessed independently as required
  - Random Access memory (RAM).

Bytes and Bits

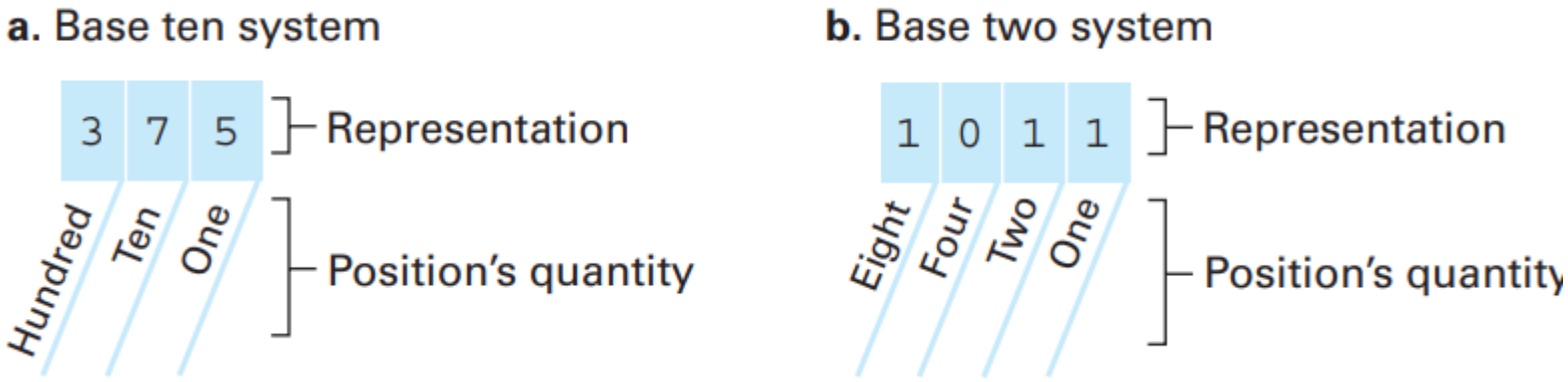
- Bit (binary digit)- 0 or 1
  - Bit Patterns are used to represent information
- Bytes
  - The amount of storage required to store a single character
  - 1 Byte = 8 bits



Bit	One binary digit
Byte	8 bits
Kilobit	1,024 or 2^{10} bits
Kilobyte	1,024 or 2^{10} bytes
Megabit	1,048,576 or 2^{20} bits
Megabyte	1,048,576 or 2^{20} bytes
Gigabit	2^{30} bits
Gigabyte	2^{30} bytes
Terabyte	2^{40} bytes
Petabyte	2^{50} bytes
Exabyte	2^{60} bytes

The Binary System

- The decimal number system is based on powers of ten.
- The Binary system is based on powers of two



Representing Numeric Values

Numbering System		
System	Base	Digits
Binary	2	0, 1
Octal	8	0,1,2,3,4,5,6,7
Decimal	10	0,1,2,3,4,5,6,7,8,9
Hexadecimal	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Hexadecimal is a shorthand notation for long bit patterns.

- Divides a pattern into groups of four bits each
- Represents each group by a single symbol

Example: 10100011 = A3

1000 1010 0101 = 8A 5

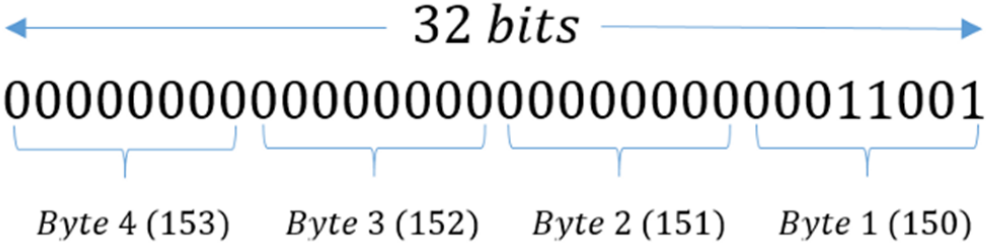
110 1110 1100 = 6 EC

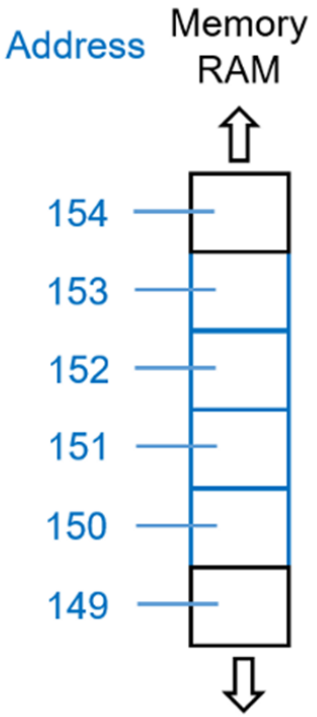
Decimal	Binary	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Example of integer value representation in memory

int $var = 25;$

$(25)_{10} = (11001)_2$





binary representation	hexadecimal	decimal
00000000 00000000 00000000 00000000	00000000	0
00000000 00000000 00000000 00000001	00000001	1
00000000 00000000 00000000 00000010	00000002	2
00000000 00000000 00000000 00000011	00000003	3
00000000 00000000 00000000 00000100	00000004	4
00000000 00000000 00000000 00000101	00000005	5
00000000 00000000 00000000 00000110	00000006	6
00000000 00000000 00000000 00000111	00000007	7
00000000 00000000 00000000 00001000	00000008	8
00000000 00000000 00000000 00001001	00000009	9
00000000 00000000 00000000 00001010	0000000A	10
00000000 00000000 00000000 00001011	0000000B	11
00000000 00000000 00000000 00001100	0000000C	12
00000000 00000000 00000000 00001101	0000000D	13
00000000 00000000 00000000 00001110	0000000E	14
00000000 00000000 00000000 00001111	0000000F	15

Representing Text

- Character data is composed of letters, symbols, and numerals that are not used in calculations.
- Character data is commonly referred to as “text.”
- Each character is assigned a unique bit pattern.
- Digital devices employ several types of codes to represent character data, including ASCII, Unicode, and their variants.
  - ASCII (American Standard Code for Information Interchange) requires seven bits for each character→ provides codes for 128 characters
    - Example: the ASCII code for an uppercase letter A is 1000001.

ASCII Table

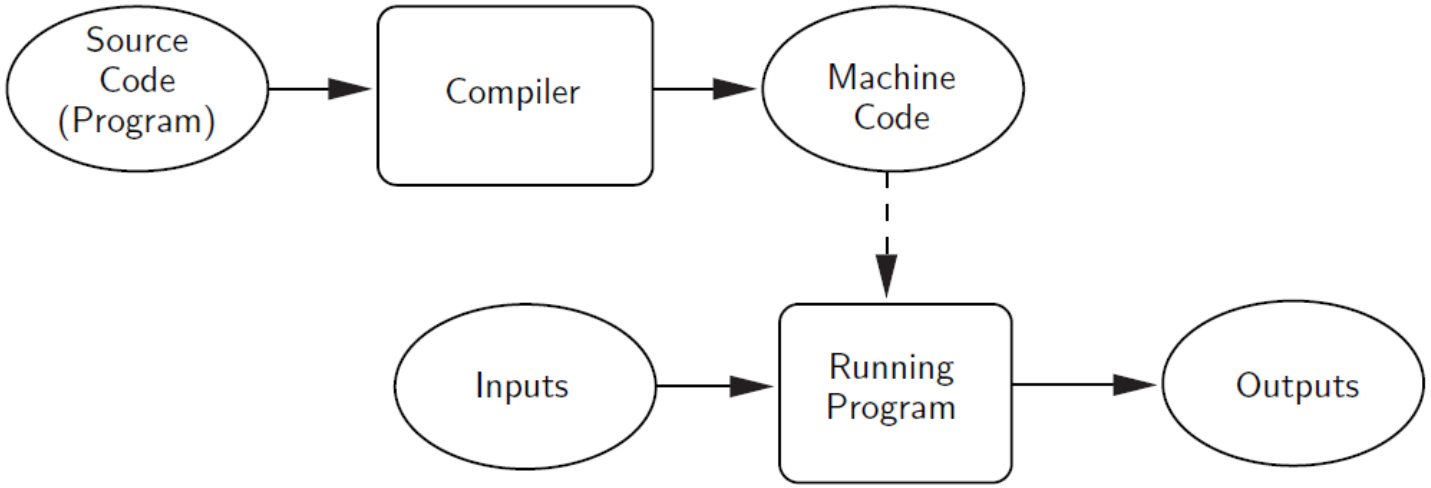
Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII
0	00000000	000	00	NUL	32	00100000	040	20	SP	64	01000000	100	40	@	96	01100000	140	60	`
1	00000001	001	01	SOH	33	00100001	041	21	!	65	01000001	101	41	A	97	01100001	141	61	a
2	00000010	002	02	STX	34	00100010	042	22	"	66	01000010	102	42	B	98	01100010	142	62	b
3	00000011	003	03	ETX	35	00100011	043	23	#	67	01000011	103	43	C	99	01100011	143	63	c
4	00000100	004	04	EOT	36	00100100	044	24	$	68	01000100	104	44	D	100	01100100	144	64	d
5	00000101	005	05	ENQ	37	00100101	045	25	%	69	01000101	105	45	E	101	01100101	145	65	e
6	00000110	006	06	ACK	38	00100110	046	26	&	70	01000110	106	46	F	102	01100110	146	66	f
7	00000111	007	07	BEL	39	00100111	047	27	'	71	01000111	107	47	G	103	01100111	147	67	g
8	00001000	010	08	BS	40	00101000	050	28	(	72	01001000	110	48	H	104	01101000	150	68	h
9	00001001	011	09	HT	41	00101001	051	29	)	73	01001001	111	49	I	105	01101001	151	69	i
10	00001010	012	0A	LF	42	00101010	052	2A	*	74	01001010	112	4A	J	106	01101010	152	6A	j
11	00001011	013	0B	VT	43	00101011	053	2B	+	75	01001011	113	4B	K	107	01101011	153	6B	k
12	00001100	014	0C	FF	44	00101100	054	2C	,	76	01001100	114	4C	L	108	01101100	154	6C	l
13	00001101	015	0D	CR	45	00101101	055	2D	-	77	01001101	115	4D	M	109	01101101	155	6D	m
14	00001110	016	0E	SO	46	00101110	056	2E	.	78	01001110	116	4E	N	110	01101110	156	6E	n
15	00001111	017	0F	SI	47	00101111	057	2F	/	79	01001111	117	4F	O	111	01101111	157	6F	o
16	00010000	020	10	DLE	48	00110000	060	30	0	80	01010000	120	50	P	112	01110000	160	70	p
17	00010001	021	11	DC1	49	00110001	061	31	1	81	01010001	121	51	Q	113	01110001	161	71	q
18	00010010	022	12	DC2	50	00110010	062	32	2	82	01010010	122	52	R	114	01110010	162	72	r
19	00010011	023	13	DC3	51	00110011	063	33	3	83	01010011	123	53	S	115	01110011	163	73	s
20	00010100	024	14	DC4	52	00110100	064	34	4	84	01010100	124	54	T	116	01110100	164	74	t
21	00010101	025	15	NAK	53	00110101	065	35	5	85	01010101	125	55	U	117	01110101	165	75	u
22	00010110	026	16	SYN	54	00110110	066	36	6	86	01010110	126	56	V	118	01110110	166	76	v
23	00010111	027	17	ETB	55	00110111	067	37	7	87	01010111	127	57	W	119	01110111	167	77	w
24	00011000	030	18	CAN	56	00111000	070	38	8	88	01011000	130	58	X	120	01111000	170	78	x
25	00011001	031	19	EM	57	00111001	071	39	9	89	01011001	131	59	Y	121	01111001	171	79	y
26	00011010	032	1A	SUB	58	00111010	072	3A	:	90	01011010	132	5A	Z	122	01111010	172	7A	z
27	00011011	033	1B	ESC	59	00111011	073	3B	;	91	01011011	133	5B	[	123	01111011	173	7B	{
28	00011100	034	1C	FS	60	00111100	074	3C	<	92	01011100	134	5C	\	124	01111100	174	7C	|
29	00011101	035	1D	GS	61	00111101	075	3D	=	93	01011101	135	5D	]	125	01111101	175	7D	}
30	00011110	036	1E	RS	62	00111110	076	3E	>	94	01011110	136	5E	^	126	01111110	176	7E	~
31	00011111	037	1F	US	63	00111111	077	3F	?	95	01011111	137	5F	_	127	01111111	177	7F	DEL

Example: The message “Hello.” in ASCII

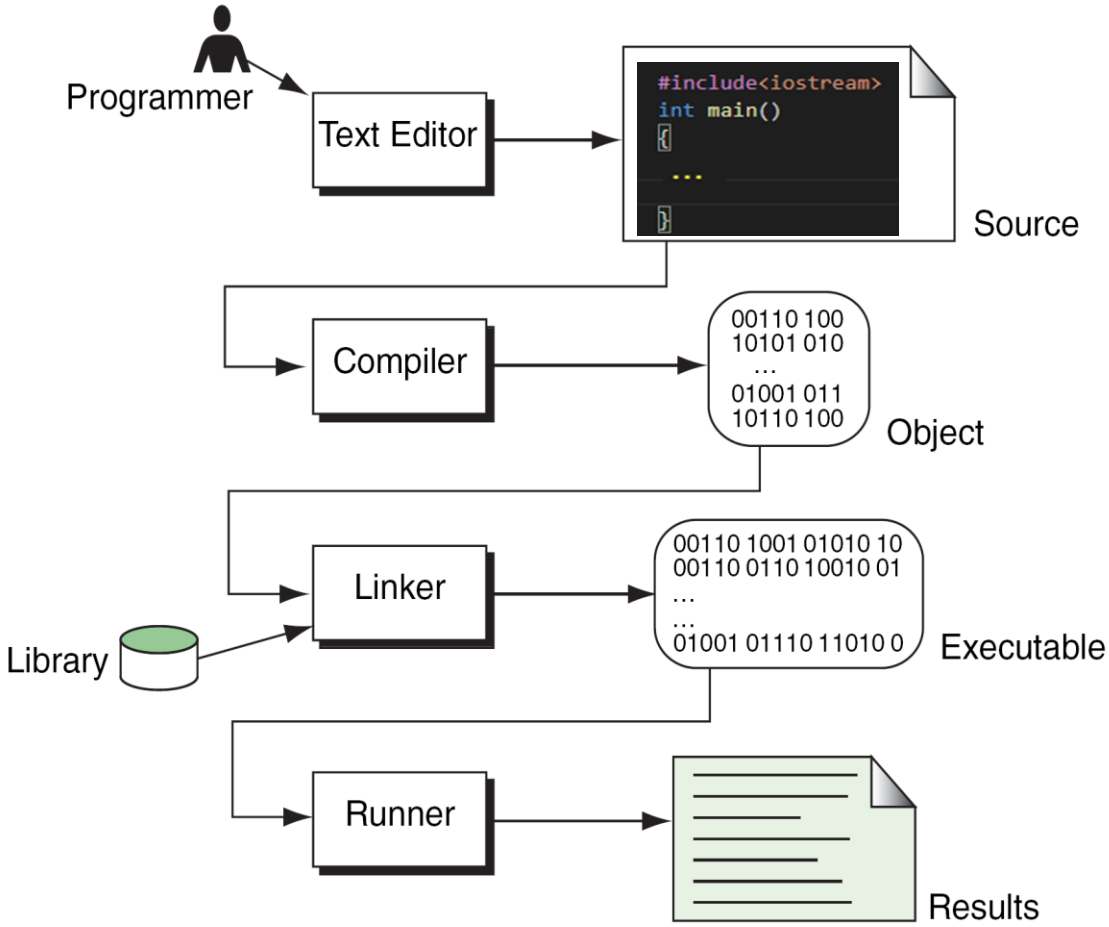
01001000	01100101	01101100	01101100	01101111	00101110
H	**e**	**l**	**l**	**o**	**.**

Creating and Running Programs

- *The procedure for turning a program written in high-level into machine language needs compiler or interpreter*



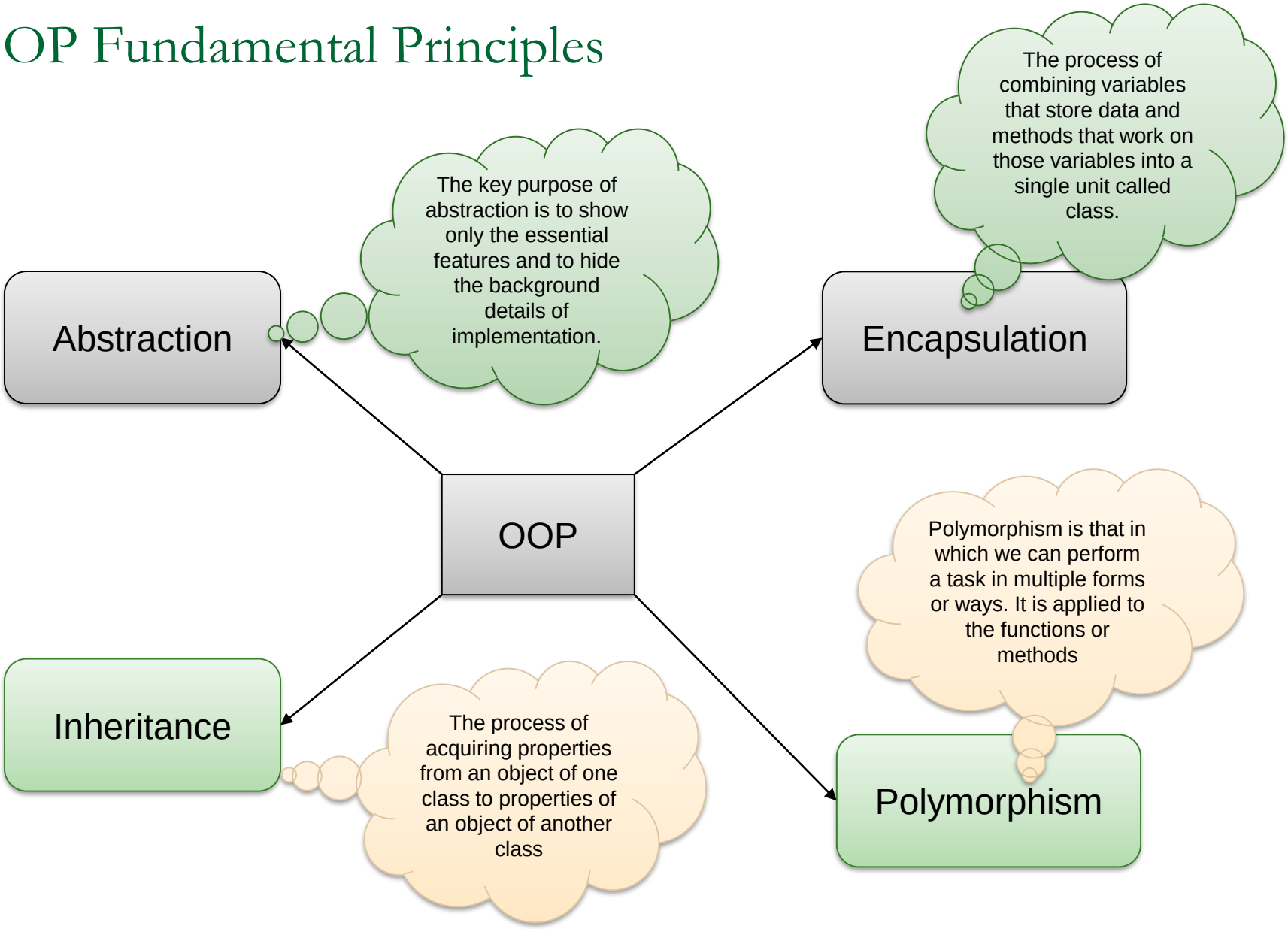
Building a C++ Program



Programing paradigms

- The two major programming paradigms are:
  - Procedural Oriented Programming(POP)
    - C, FORTRAN, Pascal , …
  - Object-Oriented Programming(OOP)
    - Java, Python, C++,…
- In procedural programming, the main program is divided into small sections known as functions, whereas in object-oriented programming, the program is divided into objects.
- In POP main focus is on functions or procedures required for computation, instead of data. The program is divided into functions, and the task is done sequentially.
- The drawbacks of procedural programming: it makes a programming too complex

OOP Fundamental Principles



The Origin of C++

The Simula languages were developed at the Norwegian Computing Center, Oslo, Norway by Ole-Johan Dahl(left) and Kristen Nygaard(right) in 1967

simula

Easy to understand

+

C

Fast execution

C was developed by Dennis Ritchie(right) & Ken Thompson(left) at the Bell Laboratories in 1972

→

C++

Easy to use and fast execution of tasks

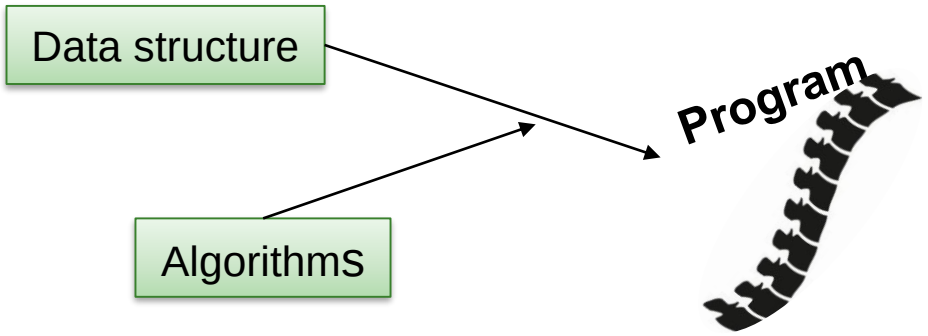
Bjarne Stroustrup

Invented by Bjarne Stroustrup in 1979 at Bell labs in Murrary hill at new Jersey.

- Initial name : c with class
- In 1993 the name was changed to c++

Algorithms

- Computer programming is all about problem solving
- The step-by-step procedure to solve a problem is known as the Algorithm
- Data structure is a way to store, organize and manage information(or data)



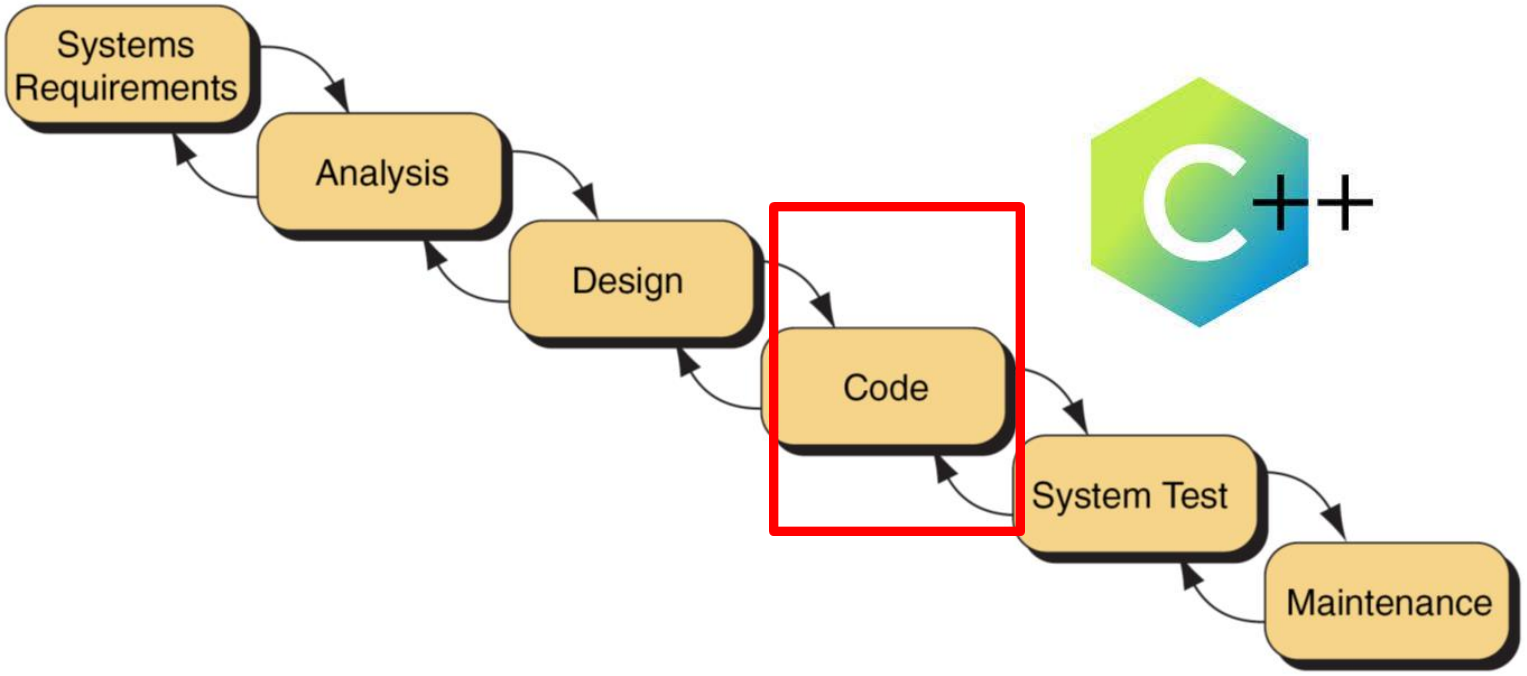
- Data structures are essential ingredients in creating fast and powerful algorithms.
- The programmer must be able to write the programs in such a way that the program should take optimum memory space and increase its execution speed
- In the world of programming, data structure and algorithms take the prime spotlight in-order to solve more complex programming issues

- **Data Structure + Algorithms = Program**

Software Development Process

- Program development is a multistep process that requires
  - Understanding a problem- requirement specification
  - Developing a solution- create a design
    - Formulate the overall structure of the program.
    - Develop your own algorithm that meets the specifications
  - Implement the design - writing the program (coding)
    - In this course we will use c++
  - Test/Debug the Program
    - Try out your program to see if it worked
    - If there are any errors (*bugs*), they need to be located and fixed. This process is called *debugging*.
    - Your goal is to find errors, so try everything that might "break" your program!
  - Maintain the Program
    - Continue developing the program in response to the needs of your users.
    - In the real world, most programs are never completely finished – they evolve over time.

Software Development Cycle




thank you