A blue decorative shape in the top-left corner, consisting of a right-angled triangle with a smaller right-angled triangle cut out of its top-right corner.

**COMPUTER
PROGRAMMING**

Computer Fundamentals

Topics covered

Computers Fundamentals: Binary Number System, Computer memory, Computer Software.

Algorithms and Programming Languages: Algorithm, Flowcharts, Generation of Programming Languages.

Course Information

Tentative Grading scheme :- 25(MST);45(EST);30(Sessional)

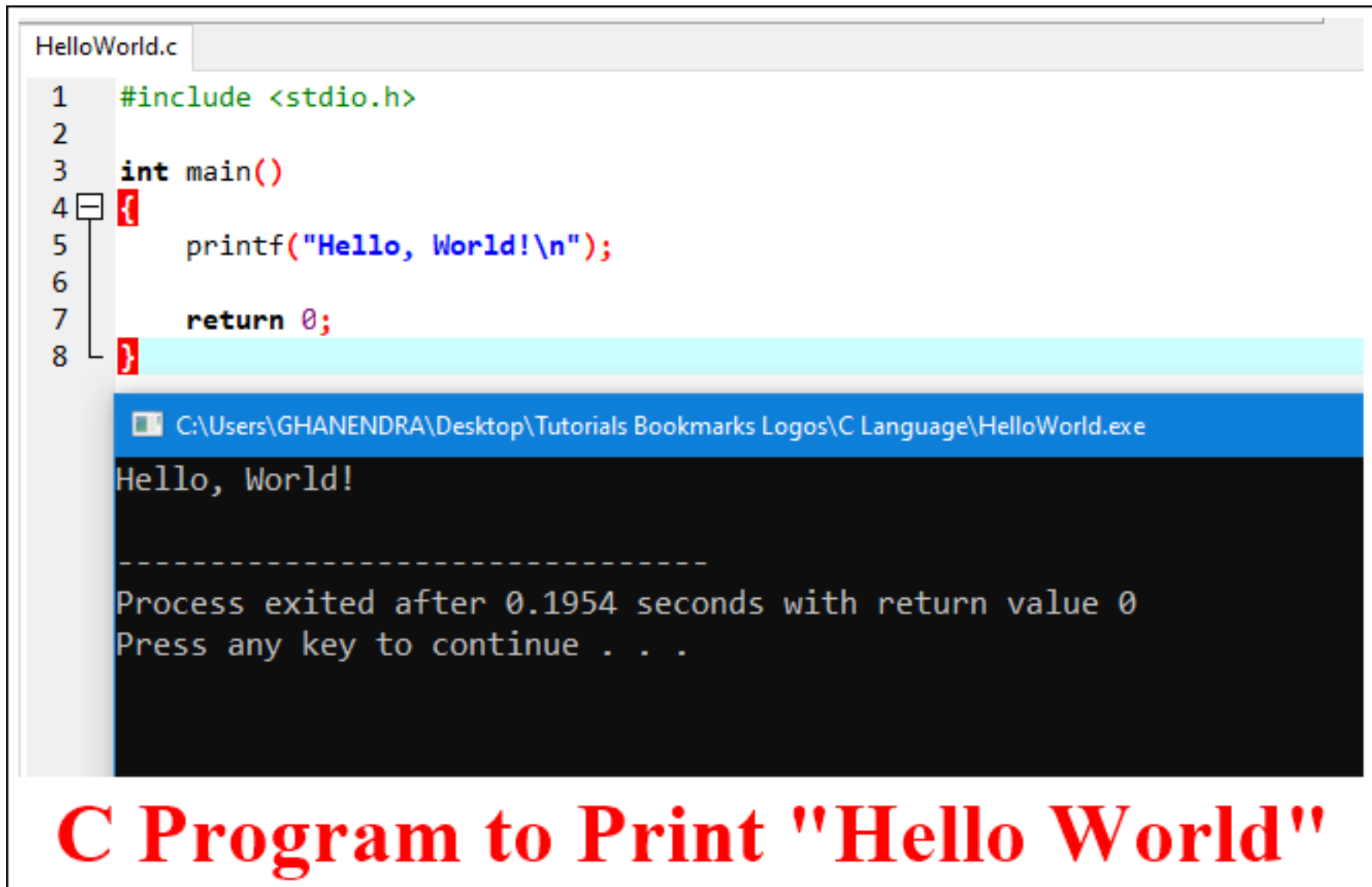
Attendance policy :- 75% including all emergencies

Punctuality :- No entry if more than 5 mins late. You should attend the class/lab in proper dress and proper place as you can be asked to switch on the video anytime.

What is programming?

Computer programming language expresses a set of detailed instructions for a computer.

Editor and Console



The image shows a code editor window titled 'HelloWorld.c' with the following C code:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      printf("Hello, World!\n");
6
7      return 0;
8  }
```

Below the editor is a console window titled 'C:\Users\GHANENDRA\Desktop\Tutorials Bookmarks Logos\C Language\HelloWorld.exe'. The console output is:

```
Hello, World!

-----
Process exited after 0.1954 seconds with return value 0
Press any key to continue . . .
```

C Program to Print "Hello World"

Compiler to run the program

https://www.onlinegdb.com/online_c_compiler

DevC++ (Find some other compiler for MAC)

CppDroid app for **Android phones**

E-box for the labs (e-box.co.in) is also for cell phones

First program

```
#include <stdio.h>
```

```
int main() {  
    printf("Hello World");  
    return 0;  
}
```

// No need to understand now, just run it

Simple integers program

```
#include <stdio.h>
```

```
int main(){  
    int a = 10, b=11,c=a+b;  
    printf("a = %d, b = %d, c = %d",a,b,c);  
    return 0;  
}
```



Computer only knows binary

Computer only knows 0 and 1

Any information can be encoded in binary strings

0 or 1 is called a bit, set of 8 bits is called a byte

Binary number system

1×2^3	1×2^2	0×2^1	1×2^0		1×2^{-1}	0×2^{-2}	1×2^{-3}	1×2^{-4}
1	1	0	1	.	1	0	1	1
8	4	0	1		0.5	0	0.125	0.0625
								
				Binary point				

$$8 + 4 + 0 + 1 + 0.5 + 0 + 0.125 + 0.0625 = 13.6875 \text{ (Base 10)}$$

<u>Decimal</u>	<u>Binary</u>
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Negative binary
numbers: 2's
complement

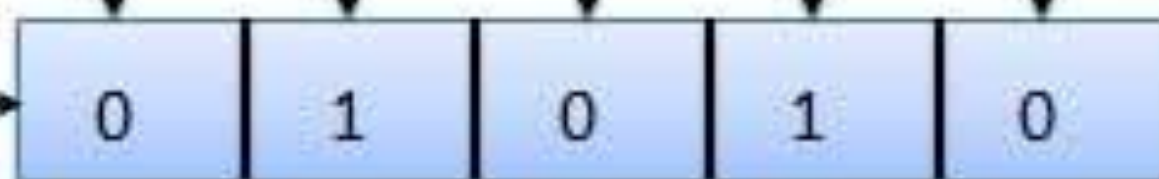
Flip zeros and
ones and add 1

decade number	binary
-8	1000
-7	1001
-6	1010
-5	1011
-4	1100
-3	1101
-2	1110
-1	1111
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111

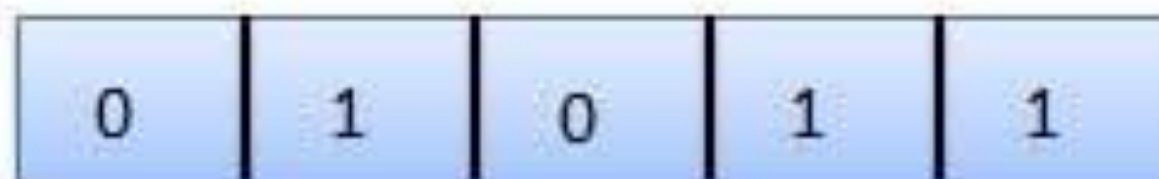
Given number →

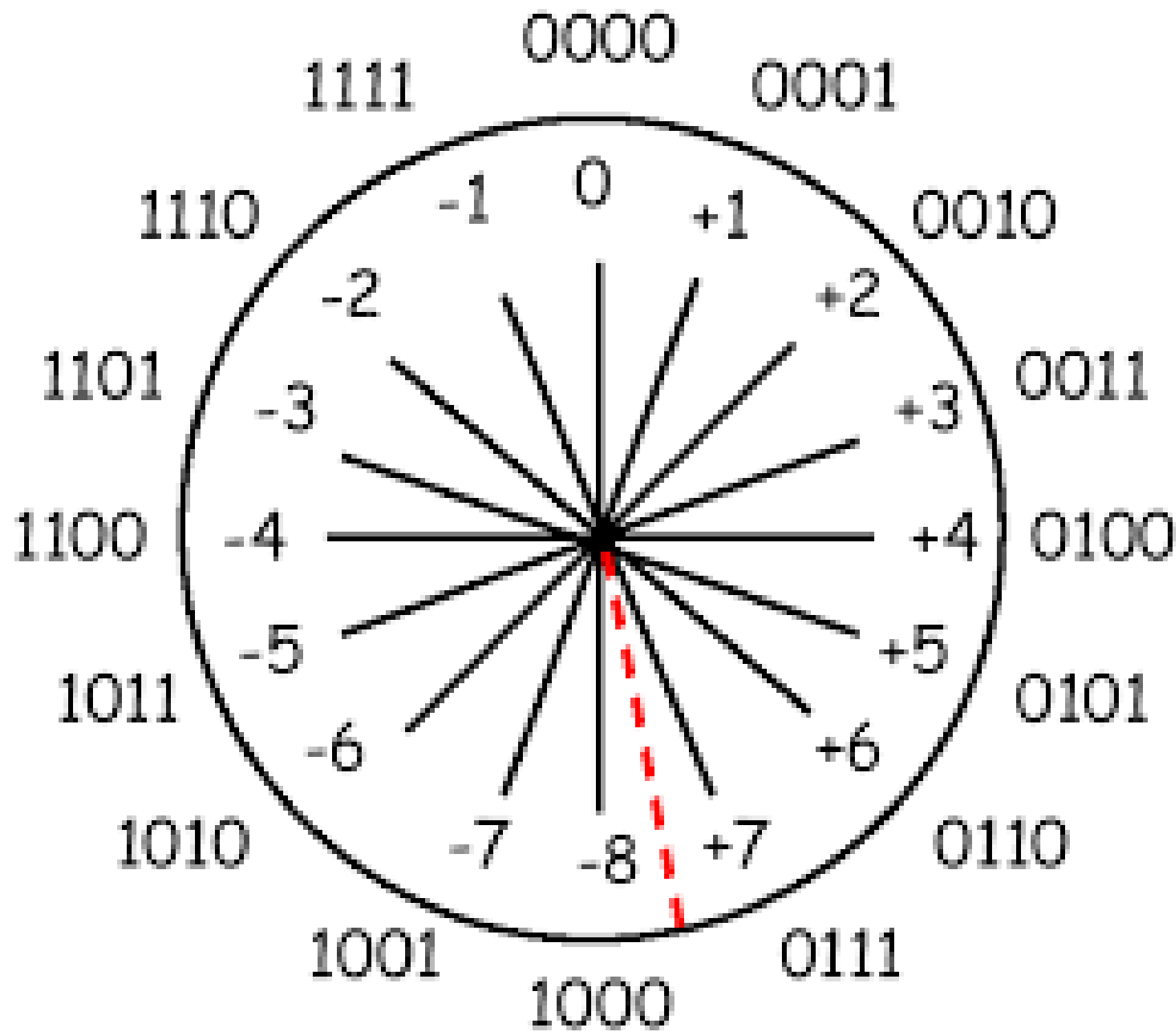


1's complement →



Add 1 +





1101 into unsigned and signed

Unsigned (+): So $1101 = 13$ in decimal

Signed (+/-): If the most significant bit (MSB) is 1 then number is negative. So 1101 means a negative number. Then find its 2's complement to find its value which is 0011 . So $1101 = -3$.

Find *signed* decimal values for 10100101 and 01111111 .

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: Decimal to Octal

$$670_{10} = 1236_8$$

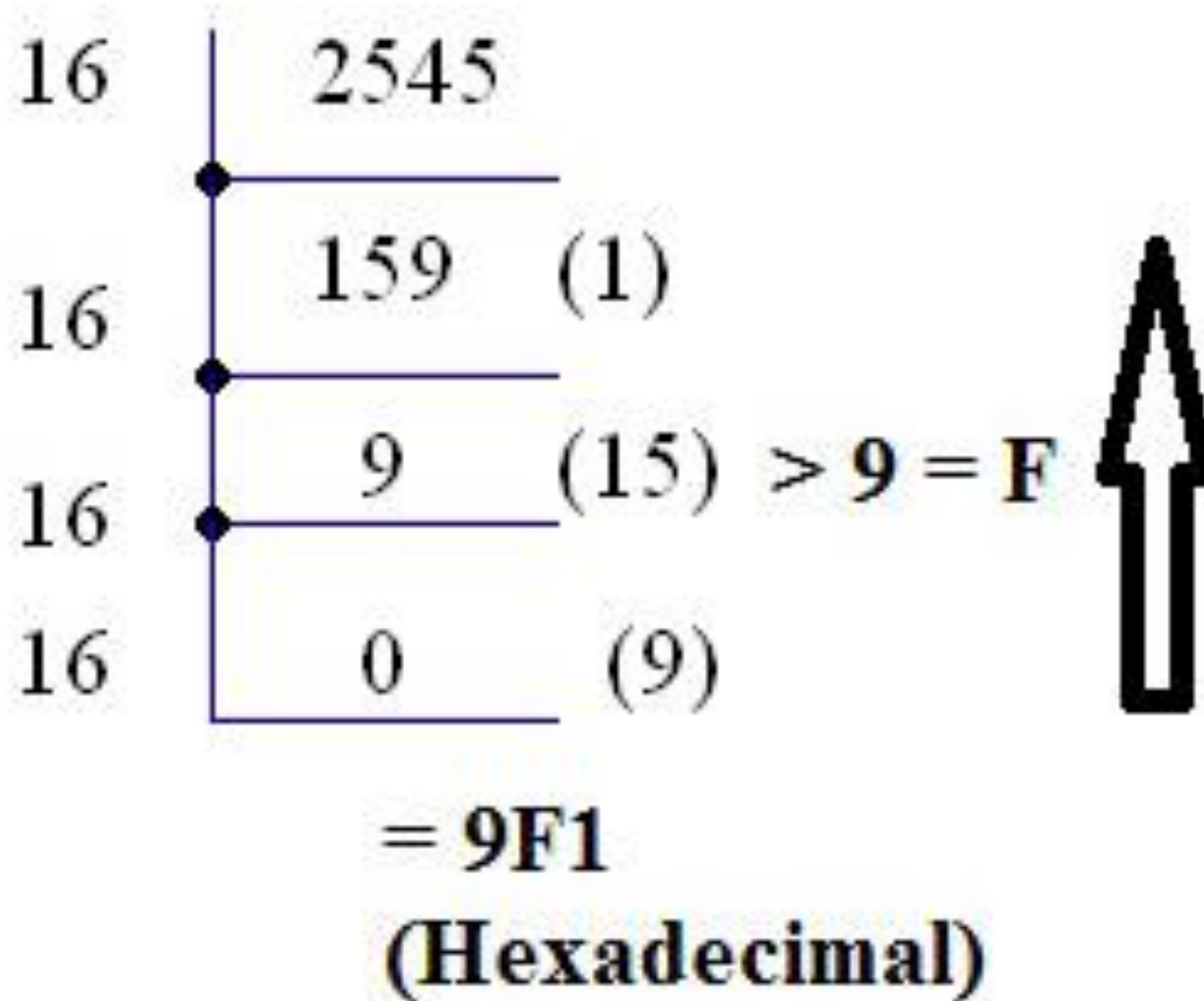
$$670 \div 8 = 83 \text{ r. } 6$$

$$83 \div 8 = 10 \text{ r. } 3$$

$$10 \div 8 = 1 \text{ r. } 2$$

$$1 \div 8 = 0 \text{ r. } 1$$

Example: Decimal to Hexadecimal



Find the Hex Equivalent
for Binary 1011010

101

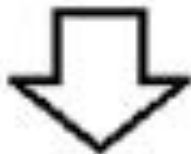
group 2

1010

group 1

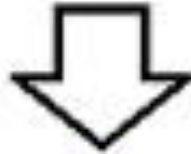
*Group 2 containing only 3 bits,
so add 0 to the left*

0101



5

1010



A

So the answer is 5A in hexadecimal

Octal to Hex Conversion

Convert the octal 752_8 to its octal equivalent.

1. Separate the digits of the given octal number, if it contains more than 1 digit.

7 5 2

2. Find the equivalent binary number for each digit of octal number. Add 0's to the left if any of the binary equivalent is shorter than 3 bits.

7 5 2

111 101 010

3. Write the all groups' binary numbers together, maintaining the same group order.

111101010

4. *Separate the binary digits into groups, each containing 4 bits or digits from right to left. Add 0s to the left, if the last group contains less than 4 bits.*

0001 1110 1010

5. *Find the hex equivalent for each group.*

0001 1110 1010

1 E A

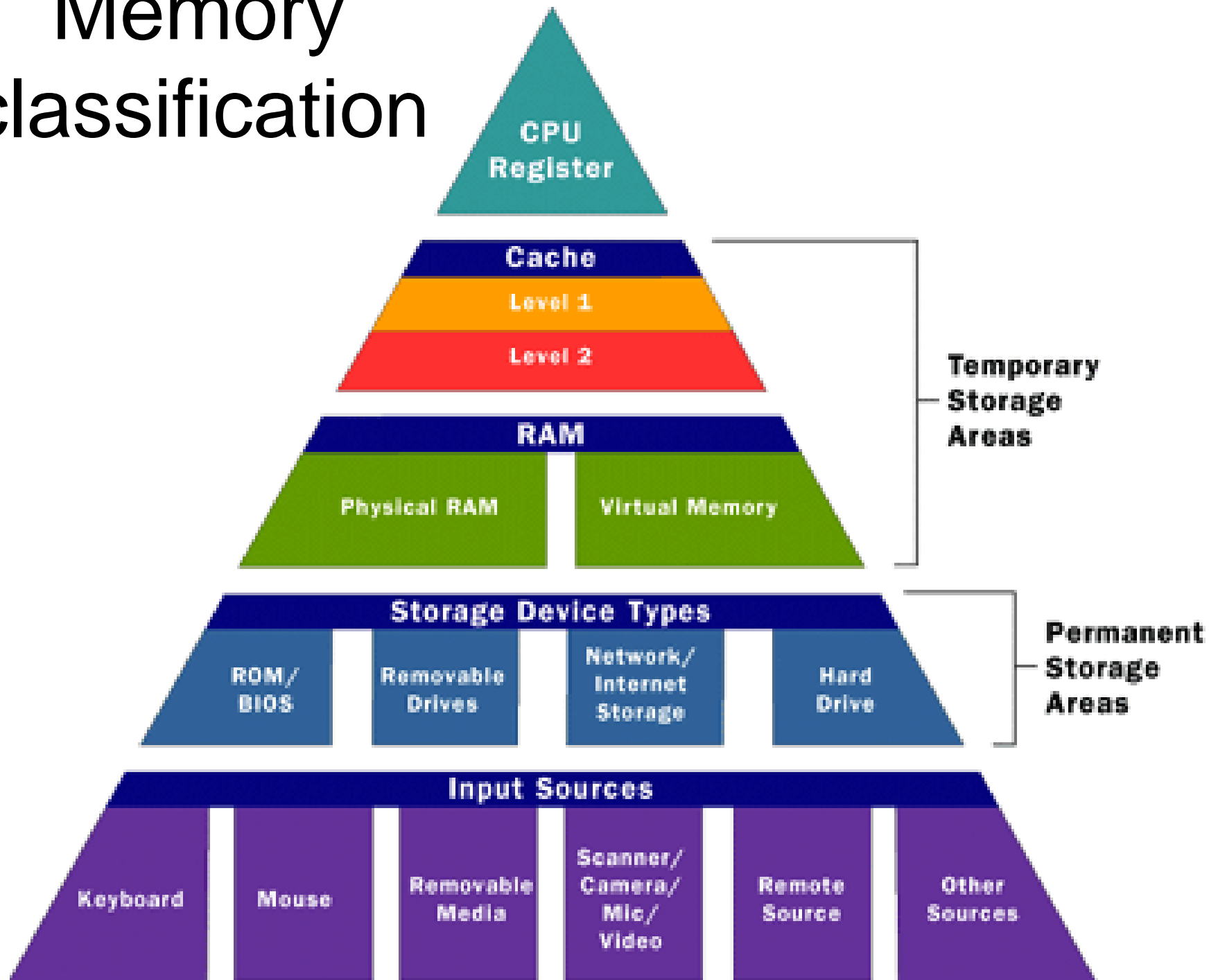
6. *Write all hex equivalent of each group together where keeping the same order provides the hex equivalent for the given octal number.*

1EA

Result

$$752_8 = 1EA_{16}$$

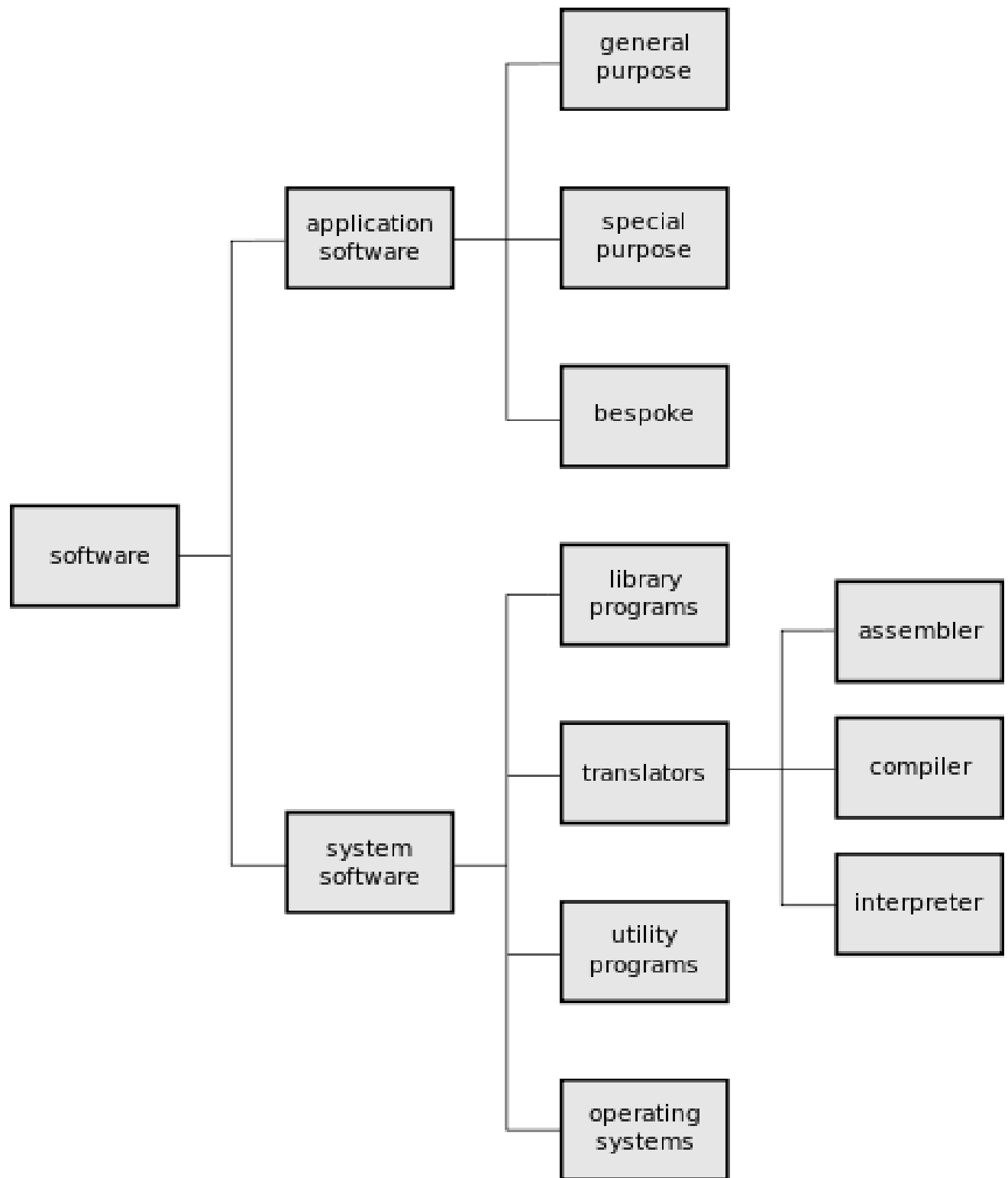
Memory classification



Memory basics

- **Registers** is small memory spaces inside processor.
- **Cache** is a high speed random access memory, integrated with the CPU.
- **Level 1** is very small (usually between 2 and 64 KB).
- **Level 2** resides on a memory card near CPU about 256 KB to 2 MB. Slower than L1 but faster than main memory.
- **Main memory** - RAM where programs and data are kept for processor
- **Virtual memory** – Part of Hard disk which is used by RAM to give an impression of larger capacity.

Software classification



Algorithms and Programming Languages:

Algorithm

Flowcharts

Generation of Programming Languages

Algorithm

Finite sequence of explicit and unambiguous instructions, which when provided with a set of input values produces an output and then terminates.

Examples Of Algorithms

Write an algorithm to add two numbers entered by user.

Step 1: Start

Step 2: Declare variables num1, num2 and sum.

Step 3: Read values num1 and num2.

Step 4: Add num1 and num2 and assign the result to sum.

$sum \leftarrow num1 + num2$

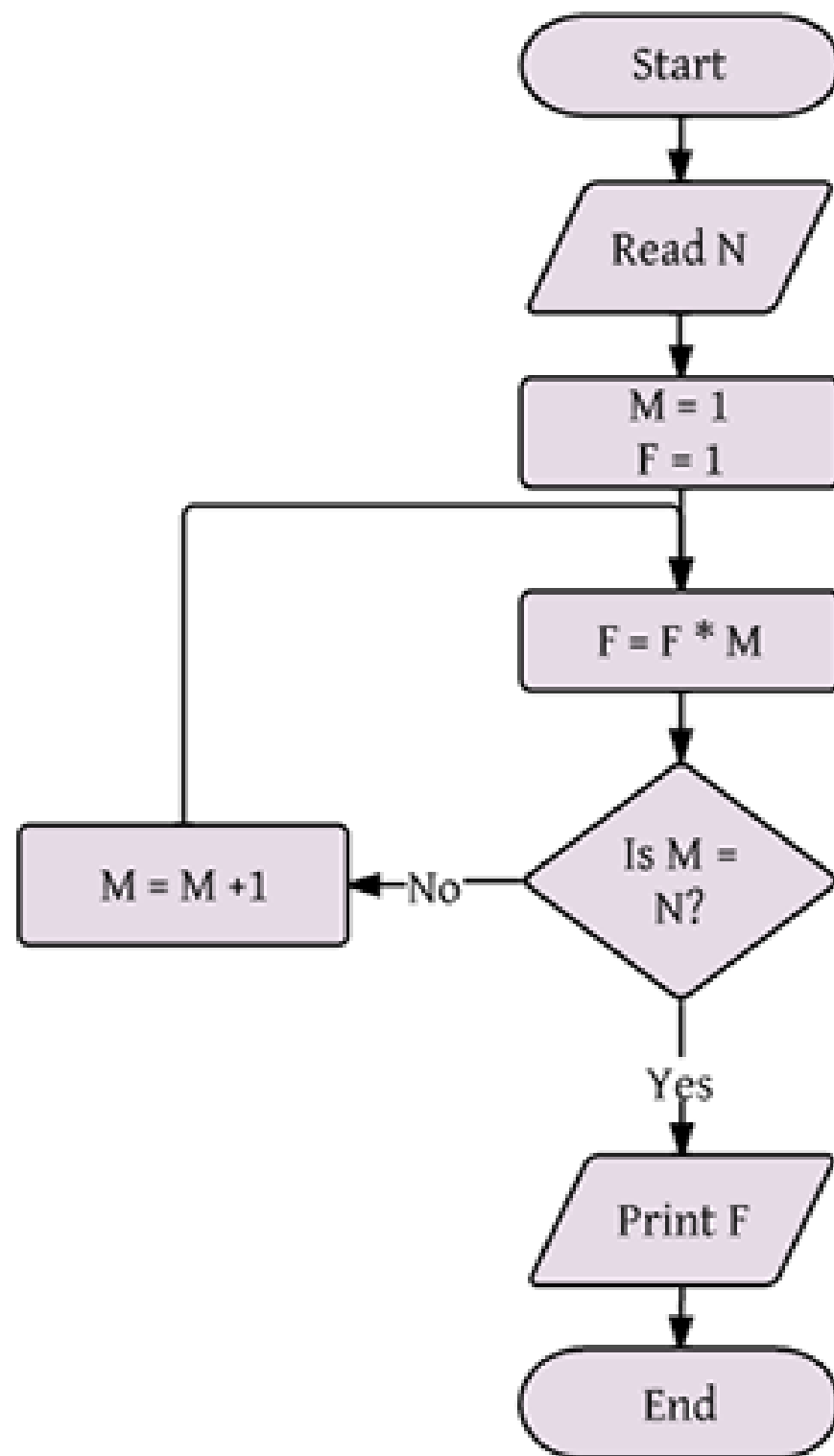
Step 5: Display sum

Step 6: Stop

Flowchart

It is a pictorial form of an algorithm.

Boxes represent operations and arrows represent sequence in which the operations are executed



Pseudo code

(1) Pseudo code is a generic way of describing an algorithm without using any specific programming language-related notations.

(2) It is an outline of a program, written in a form, which can easily be converted into real programming statements.

```
If within boundaries of search
    Calculate midpoint
    If value is at midpoint
        Return midpoint
    Else if value at midpoint is too large
        Look to the left
    Else if value at midpoint is too small
        Look to the right
    EndIf
EndIf
Return -1 if value wasn't found
```

5 generations of languages

First - machine language

Second - assembly language

Third - high-level programming languages, such as C, C++, and Java.

Forth – more close to human language, e.g. SQL

FIND ALL RECORDS WHERE NAME IS "SMITH"

Fifth - languages used for artificial intelligence and neural networks.

1-4 generation examples

Generation	First	Second	Third	Fourth
Code example	10101010011000101 10011010100000010 11111111101000101	LDA 34 ADD #1 STO 34	x = x + 1	body.top { color : red; font-style : italic }
Language	(LOW) Machine Code	(LOW) Assembly Code	(HIGH) Visual Basic, C, python etc.	(HIGH) SQL, CSS, Haskell etc.
Relation to Object Code (generally)	--	one to one	one to many	one to many

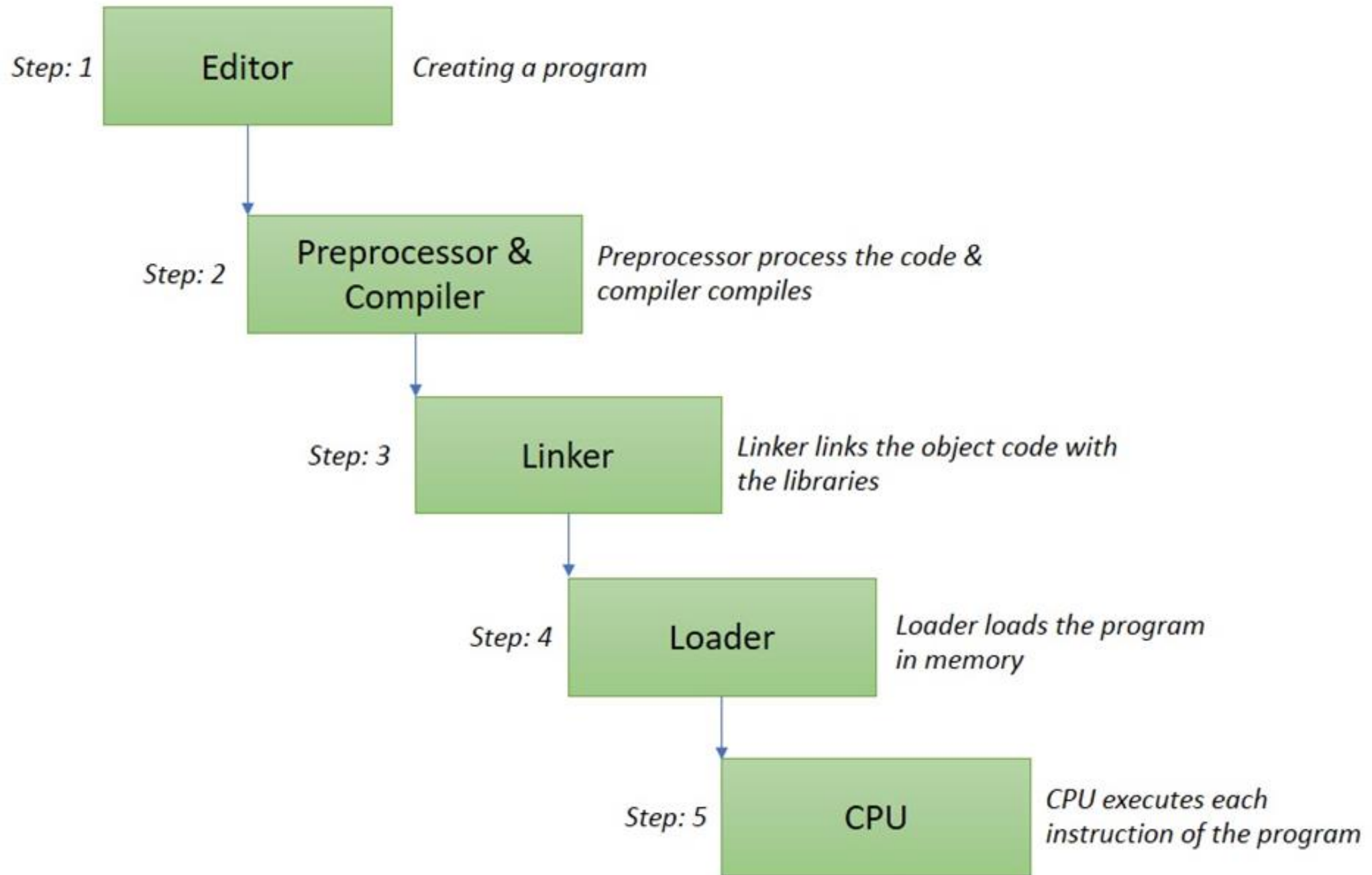
Compiler, assembler, interpreter

Compilers are used to convert high level languages (like C, C++) into machine code. **Example - GCC**

Assembler are used to convert assembly language code into machine code.

Examples - X86 assemblers

An interpreter is a computer program which executes a statement directly at runtime **Examples: Python**



Program life cycle

Basic definitions of computer
malware (malicious software)

Adware (short for advertising-supported software)

Bots automatically perform specific operations such as video gamings, online contests

Bugs are human errors in programming

Ransomware - displaying messages to demand money while locking the computer programs

Rootkit – malicious remote access to computer

Spyware – peeping Tom

Trojan horse – password stealer

Worm – exploits host computer resources

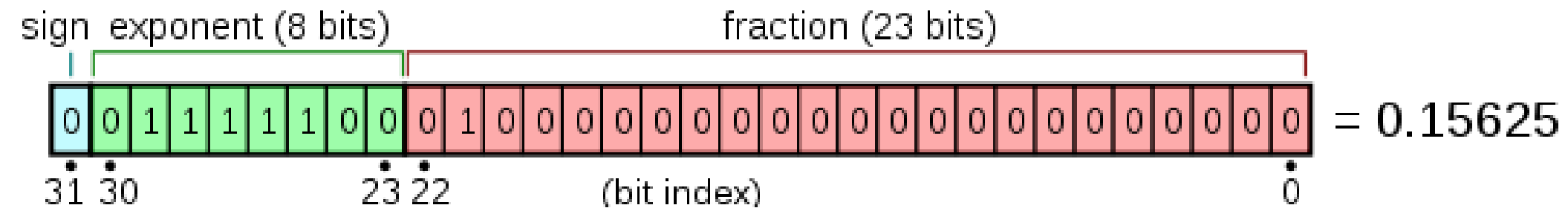
Spam – Mass of unsolicited (unwanted) emails

IEEE-754 standard

32-bit floating-point

Decimal numbers in computer memory

Basic idea – sign bit, exponent, fraction



How decimal numbers are stored in binary (4 bytes)?

Consider 5.2

5 \rightarrow 101 and 0.2 \rightarrow .00110011... (record carry for
0.2 \times 2=0.4, 0.4 \times 2=0.8, 0.8 \times 2=1.6, 0.6 \times 2=1.2)

So 5.2 = 101.00110011... = 1.010011...E+2

Conventional trick 127+2=129=10000001

0 10000001 010011001100110011001100110

SignBit 8-bit Expo 23-bits mantisa

Example 1: Suppose that IEEE-754 32-bit floating-point representation pattern is 010000000 110 0000 0000 0000 0000 0000

Sign bit $S = 0 \Rightarrow$ positive number
 $E = 1000\ 0000B = 128D$ (in normalized form)

Fraction is $1.11B$ (with an implicit leading 1) =

$$1 + 1 \times 2^{-1} + 1 \times 2^{-2} = 1.75D$$

The number is $+1.75 \times 2^{(128-127)}$
 $= +3.5D$

Exercise for floating point representation

(1) Convert -7.5D into 754 IEEE format.

(2) Convert IEEE-754 32-bit floating-point representation into floating point decimal number:

10101010 10101010 10101010 10101010

Check list – Do you know these?

Binary Number System, Computer memory, Computer Software.

Algorithm, Flowcharts, Generation of Programming Languages.