Chapter # 2
Number Systems

Student Learning Outcomes

Understand System Theory:

- The different numbering systems, including decimal, binary, hexadecimal, and octal, and their respective base values and digits.
- Why computers primarily use the binary number system for data representation.
- Machine-level representation of data, including how data is stored and processed within the computer's architecture.
- The representation of whole and real numbers in a computer, including binary encoding methods for both.
- How various arithmetic operations, such as addition, subtraction, multiplication, and division, are performed on binary representations of numbers?
- The concept of common text encoding schemes, such as ASCII and Unicode, and how they represent characters.
- How digital data representations work for various forms of multimedia, such as images, audios, videos, and other multimedia resources.
- Different file formats and their variations for specific applications.
- The concept of file extensions and their significance in identifying file types and applications.
- Key terms related to data representation, including ASCII, Unicode, binary, signed and unsigned numbers, bits, bytes, hexadecimal number systems, negatives in binary, two's complement, binary arithmetic, overflow, and underflow.
- The concept of Boolean functions, to represent logic operations and relationships between binary variables.
- How to construct Boolean expressions using variables and Boolean operators.
- Common Boolean identities and simplification techniques.
- The concept of duality in Boolean algebra, where OR becomes AND, and 0 becomes 1.
- The fundamentals of digital logic, which involves using binary digits (0 and 1) to process and store information.
- Difference between analog and digital signals and understanding their key differences.
- Various logic gates (AND, OR, NOT. NAND, XOR) and their functions in processing binary data.
- The purpose and construction of truth tables for evaluating the output of logic expressions based on input combinations.
- The concept of switches and their role in digital systems, often used to represent binary input.
- Karnaugh maps as a visual tool for simplifying Boolean expressions.
- Truth table, Boolean expression, circuit diagram of Half-adder and Full-adder.
- Half-adder and Full-adder as digital systems with specific objectives, components and interaction among those components.

2.1 NUMBERING SYSTEMS

Q.1: Explain the importance of number systems in computer science, highlighting it's conversions of using binary, decimal, octal and hexadecimal systems.

Ans. Numbering Systems:

Numbering systems are essential in computing because they form the basis for representing, storing, and processing data. Different numbering systems help computers perform tasks like calculations, data storage, and data transfer. These systems allow computers to represent various kinds of information, such as text, colors, and memory locations.

Here are four numbering systems:

1. Decimal System
2. Binary System
3. Octal System
4. Hexadecimal System

Decimal System:

The decimal number system is a base-10 number system that consists of digit from 0 to 9 and we use it in everyday life. That's why each digit of the number represents a power of 10. In the decimal system the place values starting from the right most digits are 10^0, 10^1, 10^2, and so on. For example, the decimal number 523 means:

$= 5 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$

$= 5 \times 100 + 2 \times 10 + 3 \times 1$

$= 500 + 20 + 3$

$= 523$

Binary System:

In binary, the place values are arranged from the right to left, starting with 2^0, and ending at 2^n,where each position represents a power of 2.

For example, the binary number 1011 can be converted to decimal as follows:

$= 1 \times 2^3 + 0 \times 2^2 \times 1 \times 2^1 + 1 \times 2^0$

$= 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$

$= 8 + 0 + 2 + 1$

$= 11_{10}$

Explanation:

Computers work in binary system especially because this method fits well with electronics. Digital circuits have two states: They can be either on or off. These states are easily represented by the binary digits: 1 represents ON, and 0 represnets OFF. When

typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

When you type a letter on your keyboard, the computer converts it into a binary code. Similarly, all types of data, including numbers, text, images, and sounds, are ultimately broken down into binary code.

Conversion from Decimal to Binary:

The following algorithm translate a decimal number to binary:

1. To convert decimal number to binary form, divide the decimal number by 2.
2. Record the remainder.
3. Divide the number by 2 until the quotient which is left after division is 0.
4. Meaning it is represented by the remainders and it is has read from the bottom to the top of the binary number.

Example:

83/2 = 41 remainder 1
41/2=20 remainder 1
20/2 = 10 remainder 0
10/2= 5 remainder 0
5/2 =2 remainder 1
2/2=1=1 remainder 0
1/2=0= 0 remainder 1

If the remainders are read from bottom to top then it gives the required result in binary, which is 1010011.

Octal System:

Octal is a positional numeral system with base eight, which implies that a digit to be used ranges from 0 to 7. The last digit is a single digit power of 8 while the other digits are the coefficients. In the decimal system, the place values starting from the 8^0,8^1, 8^2 and so on.

For Example: The octal number 157 means

$1 \times 8^2 + 5 \times 8^1 + 7 \times 8^0 = 64 + 40 + 7 = 107$

Explanation:

Each octal digit represents three binary digits (bits) because the octal system is base-8, and the binary system is base-2.This relationship arises from the fact that 8 is a power of 2 ($8 = 2^3$). So, each octal digit can be precisely represented by three binary digits (bits). This means that any value from 0 to 7 in octal can be converted into a 3-bit binary number. This relationship makes conversion between binary and octal straight forward.

Example:

Consider the 9-bit binary number 110101011.This number can be divided into groups of three.

Bits from right to left:

110 101 011

Each group of three bits corresponds to a single octal digit:

110=6

101=5

011=3

Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

So, the binary number 110101011 is equal to 653 in octal.

Explanation:

Note that the octal number system is not actually used in modern computers to do their work. Therefore, we can say that the binary number 110101011 is equal to 653 in octal. Whenever you have a binary number that cannot be divided into groups of a three, you will have to add zero up to the left end of it to make it appropriate.

Conversion from Decimal to Octal:

The algorithm below translates a decimal number into an octal:

1. To convert the decimal number to an equivalent octal number, divide the number by 8.
2. Write down the remainder.
3. After that divide the obtained quotient by 8.
4. Continue the divisions until one of the numbers results in 0.
5. Octal is a base eight number and the octal number is the remainder read from the bottom up to the top.

Example:

Convert 83 to octal

83/8=10 remainder 3

10/8=1 remainder 2

1/8=0 remainder 1

Going up from bottom, the remainder reading will give the desired result, that is 123 in the octal system.

Hexadecimal System:

The hexadecimal is a base 16 number system with digit number from 0 to 9 and alphabets from A to F; each digit represents 16 to the power of the position of the digit. The letters A to F stand for the numeric value of 10 to 15, The digits in hexadecimal move from right to left in place value that are 16^0, 16^1, 16^2...anothers. For Example,

The hexadecimal number 1A3 can be represented in decimal as:

$1 \times 16^2 + A \times 16^1 + 3 \times 16^0 = 1 \times 256 + 10 \times 16 + 3 \times 1 = 256 + 160 + 3 = 419$

Explanation:

The hexadecimal number system is not directly used by computers either .However, it provides an even more compact representation than octal. This makes it easier for us to read and write large binary numbers. This is because the hexadecimal system is base-16 and the binary system is base-2, therefore every single hexadecimal digit equals four binary bits. This relationship stems from the fact that 16 is a power of 2($16=2^4$).This means that any hexadecimal number between 0 and 15 then it can be converted into 4-bit binary number. Each group of four bits corresponds to a single hexadecimal digit.

Example:

The binary number 1101011010110010 equals to the hexadecimal number D6B2. In case a binary number cannot be grouped as four bits add zero(s) to the left of the number to make it fit.

1101 0110 1011 0010

Hexadecimal	Binary	Hexadecimal	Binary
0	0000	A	1010
1	0001	B	1011
2	0010	C	1100
3	0011	D	1101
4	0100	E	1110
5	0101	F	1111
6	0110		
7	0111		
8	1000		
9	1001		

Correspondence between Hexadecimal and Binary Digits:

1101=D
0110=6
1011=B
0010=2

Converting Decimal to Hexadecimal:

The following algorithm converts a decimal number to hexadecimal:

1. Convert the decimal number to an absolute value by dividing it by 16.
2. Record the quotient and the remainder.
3. Continue dividing the quotient by 16 and write down the remainder until the quotient is zero.
4. The hexadecimal number, as you might have guessed, is the remainder read from bottom to top.

Example:

Convert 2297 to hexadecimal

2297/16=143 remainder 9

143/16=8 remainder F

8/16 =0 remainder 8

Reading the remainders from bottom to top gives the required result, i.e., 8F9 in hexadecimal

2.1.1+2.1.2 DECIMAL SYSTEM + BINARY SYSTEM

Q.2: Define Decimal and Binary Number System .Explain how the decimal number system works and how to convert a decimal number to a binary number. Illustrate your explanation with an example.

Ans. Decimal System: The decimal number system is a base-10 number system that consists of digit from 0 to 9 and we use it in everyday life. That is why each digit of the number represents a power of 10. In the decimal system the place values starting from the rightmost digits are 10^0, 10^1, 10^2, and so on. For example, the decimal number 523 means:

$= 5 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$

$= 5 \times 100 + 2 \times 10 + 3 \times 1$

$= 500 + 20 + 3$

$= 523$

Binary System: In binary, the place values are arranged from the right to left, starting with 2^0, and ending at 2^n,where each position represents a power of 2.

For example, the binary number 1011 can be converted to decimal as follows:

$= 1 \times 2^3 + 0 \times 2^2 \times 1 \times 2^1 + 1 \times 2^0$

$= 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$

$= 8 + 0 + 2 + 1$

$= 11_{10}$

Explanation: Computers work in binary system especially because this method fits well with electronics. Digital circuits have two states: They can be either on or off. These states are easily represented by the binary digits: 1 represents ON, and 0 represents OFF. When typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

When you type a letter on your keyboard, the computer converts it into a binary code. Similarly, all types of data, including numbers, text, images, and sounds, are ultimately broken down into binary code.

***Conversion from* Decimal to Binary:** The following algorithm translates a decimal number to binary:

1. To convert decimal number to binary form, divide the decimal number by 2.
2. Record the remainder.
3. Divide the number by 2 until the quotient which is left after division is 0
4. Meaning it is represented by the remainders and it is read from the bottom to the

top of the binary number.

Example: Convert 83 to binary

83/2 = 41 remainder 1
41/2=20 remainder 1
20/2 = 10 remainder 0
10/2= 5 remainder 0
5/2 =2 remainder 1
2/2=1=1 remainder 0
1/2=0= 0 remainder 1

2	83
2	41 – 1
2	20 – 1
2	10 – 0
2	5 – 0
2	2 – 1
2	1 – 0
2	1 – 0
	0 – 1

If the remainders are read from bottom to top then it gives the required result in binary, which is 1010011.

2.1.3 OCTAL SYSTEM

Q.3: Explain the process of converting a decimal number to an octal number with the help of an example.

Ans. Octal System:

Octal is a positional numeral system with base eight, which implies that a digit to be used ranges from 0 to 7. The last digit is a single digit power of 8 while the other digits are the coefficients. In the decimal system, the place values starting from the 8^0,8^1, 8^2 and so on.

Example: the octal number 157 means

$= 1 \times 8^2 + 5 \times 8^1 + 7 \times 8^0$
$= 1 \times 64 + 5 \times 8 + 7 \times 1$
$= 64 + 40 + 7$
$= 111_{10}$

Explanation:

Each octal digit represents three binary digits (bits) because the octal system is base-8, and the binary system is base-2.This relationship arises from the fact that 8 is a power of 2 ($8 = 2^3$). So, each octal digit can be precisely represented by three binary digits (bits). This means that any value from 0 to 7 in octal can be converted into a 3-bit binary number. This relationship makes conversion between binary *and octal straight* forward.

Example:

Consider the 9-bit binary number 110101011.This number can be divided into groups of three

Bits from right to left: 110 101 011

Each group of three bits corresponds to a single octal digit:

110=6
101=5
011=3

So, the binary number 110101011 is equal to 653 in octal.

Explanation:

Note that the octal number system is not actually used in modern computers to do their work. Therefore, we can say that the binary number 110101011 is equal to 653 in octal. Whenever you have a binary number that cannot be divided into groups of a three, you will have to add zero up to the left end of it to make it appropriate.

Conversion from Decimal to Octal:

The algorithm below translates a decimal number into an octal.

1. To convert the decimal number to an equivalent octal number, divide the number by 8.
2. Write down the remainder.
3. After that divide the obtained quotient by 8.
4. Continue the divisions until one of the numbers results in 0.
5. Octal is a base eight number and the octal number is the remainder read from the bottom up to the top.

Example:

Convert 83 to octal

83/8=10 remainder 3
10/8=1 remainder 2
1/8=0 remainder 1

Going up from bottom, the remainder reading will give the desired result, that is 123 in the octal system.

2.1.4 HEXADECIMAL SYSETMS

Q.4: Explain the hexadecimal number system and how it can be used to represent decimal numbers. And describe the steps involved in converting a decimal number to a hexadecimal number, and illustrate your explanation with an example.

Ans. Hexadecimal System:

The hexadecimal is a base 16 number system with digit number from 0 to 9 and alphabets from A to F; each digit represents 16 to the power of the position of the digit. The letters A to F stand for the numeric value of 10 to 15, The digits in hexadecimal move from right to left in place value that are 16^0, 16^1, 16^2...anothers. For example,

The hexadecimal number 1A3 can be represented in decimal

$1 \times 16^2 + A \times 16^1 + 3 \times 16^0 = 1 \times 256 + 10 \times 16 + 3 \times 1 = 256 + 160 + 3 = 419_{10}$

1x162+Ax16'+3x16°=1x256+10x16+3x1=256+160+3=419

Explanation:

The hexadecimal number system is not directly used by computers either .However,

it provides an even more compact representation than octal. This makes it easier for us to read and write large binary numbers. This is because the hexadecimal system is base-16 and the binary system is base-2, therefore every single hexadecimal digit equals four binary bits. This relationship stems from the fact that 16 is a power of 2(16=24).This means that any hexadecimal number between 0 and 15 then it can be converted into 4 – bit binary number. Each group of four bits corresponds to a single hexadecimal digit.

Example:

The binary number 1101011010110010 equals to the hexadecimal number D6B2. In case a binary number cannot be grouped as four bits add zero(s) to the left of the number to make it fit.

1101 0110 1011 0010

Correspondence between Hexadecimal and Binary Digits:

1101=D
0110=6
1011=B
0010=2

Converting Decimal to Hexadecimal:

The following algorithm converts a decimal number to hexadecimal:

1. Convert the decimal number to an absolute value by dividing it by 16.
2. Record the quotient and the remainder.
3. Continue dividing the quotient by 16 and write down the remainder until the quotient is zero.
4. The hexadecimal number, as you might have guessed, is the remainder read from bottom to top.

Example:

Convert 2297 to hexadecimal

2297/16 = 143 remainder 9
143/16 = 8 remainder F
8/16 = 0 remainder 8

16	2297
16	143 – 9 → 9
16	8 – 15 — F
	0 – 8 — 8

Reading the remainders from bottom to top gives the required result, i.e., 8F9 in hexadecimal.

2.2 DATA REPRESENTATION IN COMPUTING SYSTEM

Q.5: Explain the binary encoding of integers and real numbers in computer systems. And how two's complement works with an example.

Ans. Data Representation in Computing Systems:

Computers can process and store a lot of information. We will discuss numeric data representation.

Binary Encoding of Integers (Z) and Real Numbers(R):

When we store data in computers, especially numbers, it is important to understand

how they are represented and stored in memory. Let us explore how different sizes of integer values are stored in 1, 2, and 4 bytes, and how both positive and negative integers are handled.

Whole Numbers (W) and Integers (Z):

Integers, also known as whole numbers, are important elements in both Mathematics and Computer Science. Knowledge of these concepts is important for primary computations, solving problems through programming, working with data and designing algorithms.

Whole Numbers(W):

Whole numbers are a set of non-negative integers. They include zero and all the positive integers. Mathematically, the set of whole numbers is:

$W=\{0,1,2,3,...\}$

In computing, whole numbers are often used to represent quantities that can't be negative.

Examples:

The number of students in a school, a person's age in years, and grades, provided there are no negative figures such as credit point balances.

A 1-byte integer has 8 bits to store values. If all 8 bits are on, it represents the maximum value, $(11111111)_2$, which is 255_{10} If all bits are off, it represents the minimum value, $(00000000)_2$, which is 0_{10}.Similarly,using 2 or 4 bytes, we get more bits to store data allowing us to store bigger values. If n is the number of bits, the maximum value that can be represented is 2^n-1

For Examples:

1- 1-Byte whole number (8 bits):Maximum value $=2^8-1= 255$·

2- 2-Byte whole number (16 bits):Maximum value $=2^{16}-1=65,535$·

3- 4-Byte whole number(32 bits):Maximum value $=2^{32}-1 = 4, 294, 967, 295$

Integers(Z):

Integers extend the concept of whole numbers to include negative numbers. In computer programming, we call them signed integers. The set of integers is represented as:

$Z = \{...-3, -2,-1,0,1,2,3,...\}$

To store both positive and negative values, one bit is reserved as the sign bit (the most significant bit). If the sign bit is ON(1), the value is negative; otherwise, it is positive. Using this system, the maximum positive value that can be stored in a 1byte signed integer is $(01111111)_2$, which is 127_{10}. As the bits available to stored a value is n -1, hence the maximum value will be $2^{n-1}-1$.We can use this formula to compute the *maximum* values for 2 and 4 bytes.

Negative Values and Two's Complement:

To store negative values, computers use a method called two's complement. To find the two's complement of a binary number, Follow these steps:

1. Invert all the bits (change Os to 1s and 1s to Os).
2. Add 1 to the Least Significant Bit (LSB).

Example:

Let us convert the decimal number -5 to an 8-bit binary number:

1. Start with the binary representation of 5:00000101 → 5: 00000101 5: 00000101_2
2. Invert all the bits:11111010_2.
3. Add1: $(11111010)_2 + (1)_2 = (11111011)_2$.

So,-5 in 8-bit two's complement is 11111011_2.

Minimum Integer Value:

For an 8-bit integer, we switch on the sign bit for the negative value and turn all bits ON. Resulting in 11111111_2. Except the first bit, we take two's complement and get 10000000_2, which is 128_{10},.Thus minimum value in 1-byte signed integer is-128,i.e,-2^7.The minimum value is computed using the formula -2^{n-1},where n is the total number of bits.

2-Byte Integer(16 bits): Minimum value= $-2^{15} = -32,768$

4-Byte Integer (32 bits): Minimum value = $-2^{31}-1 = -2,147,483,648$

Understanding how integers are stored in memory helps you appreciate the inner workings of computers and ensures you can effectively work with different data types in programming.

2.3 STORING REAL VALUES IN COMPUTER MEMORY

Q.6: Explain the process of storing real numbers (floating-point numbers) in computer memory, covering both single precision (32-bit) and double precision (64-bit) representations.

Ans. Storing Real Values in Computer Memory:

In computers, real values, also known as floating-point numbers, are used to represent number with fractions and/or decimals.

Understanding Floating-Point Representation :

Floating-point numbers (real values) are represented similarly to scientific notation as given below:

A floating-point number = sign x mantissa x $2^{exponent}$ According to the above formula, 5.75 is represented as 1.4375x2. To convert the fractional part of a real (floating-point) number from decimal (base-10) to binary (base-2) ,multiply the fractional part by 2 and write down the integral part of the result. Repeat this process with the new fractional part until the value of the fractional part becomes zero or until the required precision is achieved.

Steps for Conversion:

1. **Identify the Fractional Part:** Get the fractional part of the decimal number. For instance: in the number 4.625,the integral part is 4 and the fractional part is 0.625.
2. **Convert the Fractional Part to Binary:** Multiply the fractional part by 2,and

write down the integer that is obtained. Repeat this process with the new fractional part till it gets to 0 or until then required number of decimal places is achieved .

Example:

Converting 0.375 to Binary

1. Identify the Fractional Part: Fractional part:0.375
2. Convert the Fractional Part 0.375 to Binary:

0.375x2=0.75(Integer part:0)

0.75x2=1.5(Integer part:1)

0.5x2=1.0(Integer part:1)

The integer parts recorded are 0, 1,1.

Combine the Results:

Combine the binary representations of the integer parts from top to bottom:

$0.375_{10} = 0.011_2$

In computing, it is critical to express real numbers in a binary form since it facilitates computing and storage. This process involves converting both the integer (decimal) and the fractional parts of a given number into binary. Two commonly use standards for this representation are "Single Precision (32-bit)"and "Double Precision (64-bit)".

Single Precision(32-bit):

In this standard, 4 bytes (or 32 bits) are assigned where the 1st bit is the sign bit, and the next 8 bits are for the exponent and the remaining 23 bits are for the mantissa.

Here the exponent can be ranged between-126 and+127.

The approximate range of values from 1.4×10^{-45} to 3.4×10^{38}.

Each floating point value is broken down into three main components: the sign bit, the exponent, and the mantissa.

Value	Representation	Sign Bit	Exponent (8 bits)	Mantissa (23 bits)
Grouping		1 bit	8 bits	23 bits
5.75	1.4375×2^2	0	10000001	10111000000000000000000
–5.75	-1.4375×2^2	1	10000001	10111000000000000000000
0.15625	1.25×2^{-3}	0	01111101	01000000000000000000000
–0.15625	-1.25×2^{-3}	1	01111101	01000000000000000000000

Grouping:

This row explains the bit allocation for the 32-bit floating point format: 1 bit for the sign, 8 bits for the exponent, and 23 bits for the mantissa.

5.75: Representation: 1.4375×2^2 – Sign Bit: 0 (positive) - Exponent: 2 +127=129, which is 10000001_2 – Mantissa: The binary representation of 0.4375 is $(10111000000000000000000)_2$.

–5.75: Representation: -1.4375×2^2 –Sign Bit:1(negative)-Exponent:2+127= 129, which is 10000001_2 – Mantissa: The binary representation of 0.4375 is $(10111000000000000000000)_2$.

0.15625:Representation:1.25×2^{-3}-Sign Bit:0 (positive)-Exponent:-3+127= 124, which is 01111101_2 –Mantissa: The binary representation of 0.25 is $(01000000000000000000000)_2$.

–0.15625: Representation:–1.25×2^{-3}–Sign Bit: 1(negative) - Exponent:–3+127 =124, which is $(01111101)_2$ – Mantissa: The binary representation of 0.25 is $(01000000000000000000000)_2$

This breakdown helps illustrate how floating point values are stored and manipulated in computer systems

Double Precision(64-bit):

In double precision, the exponent is represented using 11 bits. The exponent is stored in a biased form, with a bias of 1023. The range of the actual exponent values can be determined as follows:

Bias: 1023

Exponent range: The actual exponent [illegible] range from-1022 to +1023.

Therefore, the smallest and largest [illegible]ble exponent values in double-precision are:

Minimum exponent: –1022

Maximum exponent: + 1023

We can perform the same steps given for the single-precision, except the difference of the above mentioned values

The information about how real values is stored in computer memory help us understand the precision and limitations of digital computation. With this understanding of floating-point representation, it becomes possible to control and manipulate these numbers in different ways.

2.4 BINARY ARITHMETIC OPERATIONS

Q.7: Explain how binary addition, subtraction, multiplication, and division are performed in the binary number system. Provide examples for each operation to illustrate the steps involved.

Ans. Binary Arithmetic Operations:

Arithmetic operations include addition, subtraction, multiplication and division, and are performed on two numbers at a time. Binary arithmetic operations are similar to decimal operations but follow binary rules. Here is a brief overview of the basic operations;

Addition:

Binary addition uses only two digits 0 and 1.Here,we will learn how to add binary numbers and how to handle the addition of negative binary numbers.

Binary Addition Rules:

Binary addition follows these simple rules:

1. 0 + 0 = 0

2. $0 + 1 = 1$
3. $1 + 0 = 1$
4. $1 + 1 = 0$ (with a carry of 1 to the next higher bit)

Example of Binary Addition:

Example 1:

$1101 + 1011$

In this example:

$1+1 = 0$(carry 1).

$0+1+1$(carry) $= 0$(carry 1)·

$1+0+1$(carry) $= 0$(carry 1)·

$1+1+1$(carry) $= 1$(carry 1)

Subtraction:

In binary arithmetic, subtraction can also be carried out by adding the two's complement or the value of the subtrahend to the minuend.

Example:

Subtract 6 from 9 in Binary:

Minuend $= 9_{10} = 1001_2$

Subtrahend $= 6_{10} = 0110_2$

Step 1:

Find the Two's Complement of the Subtrahend:

Invert the bits of 0110_2:

Inversion: 1001_2

Add 1 to the inverted number:

$1001_2 + 1_2 = 1010_2 = -6_{10}$

Step 2:

Add the Minuend and the Two's Complement of the Subtrahend:

$1001_2 + 1010_2 = 10011_2$

Step 3:

Discard the Carry Bit

10011_2 Discard carry $0011_2 = 3_{10}$

So, $9 - 6 = 3$.

Multiplication:

Binary numbers are base-2 numbers, consisting of only 0s and 1s. Multiplying binary numbers follows similar principles to multiplying decimal numbers, but with simpler rules. Here, we will learn how to multiply binary numbers with example.

Steps to Multiply Binary Numbers:

1. Write down the binary numbers, aligning them by the least significant bit (rightmost bit).
2. Multiply each bit of the second number by each bit of the first number, similar to the long multiplication method in decimal.
3. Shift the partial results one place to the left for each new row, starting from the

second row.

4. Add all the partial results to get the final product.

Example

Let us multiply two binary numbers: 101_2 and 11_20

$$\begin{array}{r} 101 \\ \times 11 \\ \hline 101 \\ 101\times \\ \hline 1111 \end{array}$$

So, $101_2 \times 11_2 = 1111_2$

Binary Division:

Binary division is similar to decimal division but only involves two digits: 0 and 1.

It follows steps like comparing, subtracting, and shifting, akin to long division in the decimal system.

Steps of Binary Division

Compare: Compare the divisor with the current portion of the dividend.

Subtract: Subtract the divisor from the dividend portion if the divisor is less than or equal to the dividend.

Shift: Shift the next binary digit from the dividend down to the remainder.

Repeat: Repeat the process until all digits of the dividend have been used.

Example

Step 1: Compare 10 with first two 11, subtract 10 from 11)

Step 2: Bring down the next digit 0)

Step 3: Compare 10 with 10, subtract 10 from 10)

Step 4: Bring down the next digit 0, no more digits left)

Example:

Divide 1100_2 by 10_2

$$\begin{array}{r|l} & 110 \\ \hline 10 & 1100 \\ & -10 \\ \hline & 10 \\ & -10 \\ \hline & 0 \end{array}$$

Resuet: $(1100)_2/(10)_2 = 110_2$

2.5 COMMON TEXT ENCODING SCHEMES

Q.8: Explain the different text encoding schemes commonly used in computer systems, focusing on their underlying principles, advantages, and limitations.

Ans. Common Text Encoding Schemes:

Text encoding schemes are essential for representing characters from various languages and symbols in a format that computers can understand and process. Here are

some of the most common text encoding schemes used in computers:

ASCII:

ASCII is an acronym that stands for American Standard Code for Information Interchange. It is a character encoding standard adopted for representing in devices such as computers and similar systems that use text. Each alphabet, number or symbol is given a code number between 0 and 127. ASCII enables different computers and devices to exchange text information reliably.

Let us encode the name of our country using ASCII.

1. The ASCII code for an upper case letter "P" is 80.
2. The code for letter 'a' in ASCII is 97.
3. The ASCII code for the letter 'k' is 107.
4. It is interesting to know that the ASCII code for the letter 'i' is 105.
5. In the ASCII code system, the letter 's' has a code of 115.
6. The code for 't' is 116 in ASCII.

The ASCII code is a numerical representation of characters in computer-based system, particularly for alphabetic characters.

For Example: the ASCII code of the character 'n' is 110.

Character	ASCII Code	Character	ASCII Code
SP (space)	32	!	33
"	34	#	35
$	36	%	37
(	40	)	41
*	42	+	43
,	44	-	45
.	46	/	47
0	48	1	49
2	50	3	51
4	52	5	53
6	54	7	55
8	56	9	57
:	58	;	59
<	60	=	61
>	62	?	63
@	64	A	65
B	66	C	67
D	68	E	69
F	70	G	71

H	72	I	73
J	74	K	75
L	76	M	77
N	78	O	79
P	80	Q	81
R	82	S	83
R	84	U	85
V	86	W	87
X	88	Y	89
Z	90	[	91
\	92	]	93
^	94	-	95
?	96	a	97
b	98	c	99
d	100	e	101
f	102	g	103
h	104	i	105
j	106	k	107
l	108	m	109
n	110	o	111
p	112	q	113
r	114	s	115
t	116	u	117
y	118	w	119
x	120	y	121
z	122	{	123
|	124	}	125
~	126	DEL	127

Extended ASCII:

While the standard ASCII Table includes 128 characters, there is an extended version that includes 256 characters. This extended ASCII uses 8 bits and includes additional symbols, accented letters, and other characters. *However, the original128 characters are the most commonly used and serves as the basis for text representation in computers.*

Unicode:

Unicode is an attempt at mapping all graphic characters used in any of the world's

writing system. Unlike ASCII, which is limited to 7bits and can represent only 128 characters, Unicode can represent over a million characters through different forms of encodings such as, UTF-8, UTF-16, and UTF-32. UTF is an acronym that stands for Unicode Transformation Format.

UTF-8:

It is a variable-length encoding scheme, meaning it can use a different numbers sof bytes (from 1 to 4) to represent a character. UTF-8 is backward compatible with ASCII. It means it can understand and use the older ASCII encoding scheme without any problems .Therefore, if we have a text file written in ASCII, it will work perfectly fine with UTF-8,allowing it to read both old and new texts.

Example: The letter 'A' is Unicode, represented as, U+0041,is 01000001 in the binary format and occupies 8 bits or 1 byte.

Let us look at how Urdu letters are represented in UTF-8:

Example:

The Urdu letter ب is represented in Unicode as U+0628; its binary format is 11011000 10101000,means it takes 2 bytes.

UTF-16:

UTF-16 is another variable character encoding mechanism, although it uses either 2 bytes or 4 bytes per character at most. Unlike UTF-8, it is not compatible with ASCII, meaning it cannot translate ASCII code.

Example:

The letter A in UTF-16 is equal to 00000000 01000001 in binary or 65 in decimal(2 bytes).

For Urdu:

Example: The right Urdu letter ب in UTF-16 is represented as is 0000011000101000 in binary, which occupies 2 bytes of memory.

UTF-32:

UTF-32 is a method of encoding that uses a fixed length, with all characters stored in 4 bytes per character. This makes it very simple but at the same time it may look a little complicated when it comes to space usage.

Example:

Alphabet letter 'A' in UTF-32 is represented in binary as 00000000 00000000 00000000 01000001 which is 4 bytes.

2.6 STORING IMAGES, AUDIO AND VIDEO IN COMPUTERS

Q.9: Explain in detail the process of storing images, audio, and video on a computer, including the concepts of pixels, sampling rate, and frame rate.

Ans. Storing Images, Audio, and Video in Computers:

Have you ever wondered how your favorite photos, songs, and movies are stored on your computer or phone? let us dive into the fascinating world of digital storage to understand how computers manage these different types of files:

Storing Images:

Images are made up of tiny dots called pixels. Each pixel has a color, and the combination of all these pixels forms the complete picture. Computers store images using numbers to represent these colors.

Color Representation: In a color image, each pixel's color can be represented by three numbers: Red, Green, and Blue (RGB). Each of these numbers typically ranges from 0 to 255.-For example: a pixel with RGB values (255, 0, 0) will be bright red.

Image File Formats: The following are commonly used image formats for photos-**JPEG** (Joint Photographic Expert Group): It compresses the image to save space but might lose some quality. - **PNG** (Portable Network Graphics): Supports transparency and maintains high quality without losing data. -**GIF**(Graphics Interchange Format):Used for simple animations and images with few colors.

Storing Audio:

Audio files are stored by capturing sound waves and converting them into digital data.This process involves sampling and quantization.

Sampling and Quantization:

Sampling: Recording the sound wave at regular intervals. The number of samples per second is called the **sampling rate**. Higher sampling rates result in better quality.

Quantization: Converting each sample into a number. More bits per sample provide more accurate sound representation.

Audio File Formats:

Mp3: A common format that compresses audio to save space but may lose some quality.

WAV (Wave Audio File Format): Uncompressed format that maintains high quality.

AAC (Advanced Audio Coding): Used by many streaming services for high-quality audio with efficient compression.

To explain how these units are used, consider the following examples:

1. An image file might have a size of 500 KB(Kilobytes).
2. A music file might be around 5MB(Megabytes).
3. A full-length HD movie could be approximately 2 GB (Gigabytes).
4. A large external hard drive could have a capacity of 1 TB(Terabytes).

Storing Video:

Videos are made up of many images shown rapidly in sequence, along with audio.Each image in a video is called a **frame**.

Frames and Frame Rate:

Frame Rate: The number of frames shown per second, measured in Frames Per Second (FPS). Common frame rates are 24 fps (used in movies) and 30 fps(used in TV). Higher frame rates result in smoother motion in videos.

Video File Formats:

MP4: A widely used format that efficiently compresses video while maintaining quality.

AVI: An older format that may result in larger file sizes.

MKV: Supports high-quality video and multiple audio tracks or subtitles.

How Computers Store These Files:

All these files (images, audio, and video)are stored as **binary data**, which means they are represented by sequences of Os and 1s.

Storage Devices:

Hard Disk Drive (HDD): Uses spinning disks to read/write data. They offer large storage capacities.

Solid State Drive (SSD): Uses flash memory for faster access times and better performance.

Cloud Storage: Stores files on remote servers accessible via the internet, providing flexibility and backup options.

Conceptual Long Questions

Q.1: Elaborate on the difference between whole numbers and integers, emphasizing their representation in memory. Discuss the concept of two's complement and its role in storing negative integers.

(OR) What are the two main types of numeric data representation Discuss?

Ans. Data Representation in Computing Systems:

Computers can process and store a lot of information. We will discuss numeric data representation.

Binary Encoding of Integers (Z) and Real Numbers(R):

When we store data in computers, especially numbers, it is important to understand how they are represented and stored in memory. Let us explore how different sizes of integer values are stored in 1, 2, and 4 bytes, and how both positive and negative integers are handled.

Whole Numbers (W) and Integers (Z):

Integers, also known as whole numbers, are important elements in both Mathematics and Computer Science. Knowledge of these concepts is important for primary computations, solving problems through programming, working with data and designing algorithms.

Whole Numbers(W):

Whole numbers are a set of non-negative integers. They include zero and all the positive integers. Mathematically, the set of whole numbers is:

$W=\{0,1,2,3,...\}$

In computing, whole numbers are often used to represent quantities that can't be negative.

Examples:

The number of students in a school, a person's age in years, and grades, provided there are no negative figures such as credit point balances.

A 1-byte integer has 8 bits to store values. If all 8 bits are on, it represents the maximum value, 11111111_2, which is 255_{10}. If all bits are off, it represents the minimum

value, 00000000_2, which is 0_{10}.Similarly,using 2 or 4 bytes, we get more bits to store data allowing us to store bigger values. If n is the number of bits, the maximum value that can be represented is $2^n - 1$

for examples:

1. 1– Byte whole number (8 bits):Maximum value $= 2^8 - 1 = 255$.
2. 2 – Byte whole number (16 bits):Maximum value $= 2^{16} - 1 = 65,535$.
3. 4 – Byte whole number(32 bits):Maximum value $= 2^{32} - 1 = 4,294,967,295$

Integers(Z):

Integers extend the concept of whole numbers to include negative numbers. In computer programming, we call them signed integers. The set of integers is represented as:

$Z = \{..., -3, -2, -1, 0, 1, 2, 3, ...\}$

To store both positive and negative values, one bit is reserved as the sign bit (the most significant bit). If the sign bit is ON(1), the value is negative; otherwise, it is positive. Using this system, the maximum positive value that can be stored in a 1byte signed integer is $(01111111)2$, which is 127_{10},.As the bits available to stored a value is n -1, hence the maximum value will be $2^{n-1} - 1$.We can use this formula to compute the maximum values for 2 and 4 bytes.

Negative Values and Two's Complement:

To store negative values, computers use a method called two's complement. To find the two's complement of a binary number, follow these steps:

1. Invert all the bits (change Os to 1s and 1s to Os).
2. Add 1 to the Least Significant Bit (LSB).

Example:

Let us convert the decimal number -5 to an 8-bit binary number:

1. Start with the binary representation of 5:00000101_2.
2. Invert all the bits:11111010_2.
3. Add1: $11111010_2 + 1_2 = 11111011_2$.

So,–5 in 8–bit two's complement is 11111011_2.

Minimum Integer Value:

For an 8 – bit integer, we switch on the sign bit for the negative value and turn all bits ON. resulting in 11111111_2,. Except the first bit, we take two's complement and get 10000000_2, which is 128_{10},.Thus minimum value in 1 – byte signed integer is–128,i.e.,-2^7.The minimum value is computed using the formula -2^{n-1},where n is the total number of bits.

2-Byte Integer (16 bits): Minimum value $= -2^{15} = -32,768$

4-Byte Integer (32 bits): Minimum value $= -2^{31} - 1 = -2,147,483,648$

Understanding how integers are stored in memory helps you appreciate the inner workings of computers and ensures you can effectively work with different data types in programming.

Q.2: Discuss the different components of the single and double precision representations (sign bit, exponent, mantissa), how they are used to represent the value, and the range of values that can be represented using each format.

Ans. Two commonly use standards for this representation are "Single precision (32-bit)"and "Double Precision (64-bit)".

Single Precision(32-bit):

In this standard, 4 bytes (or 32 bits) are assigned where the 1st bit is the sign bit, and the next 8 bits are for the exponent and the remaining 23 bits are for the mantissa.

Here the exponent can be ranged between-126 and+127.

The approximate range of values from 1.4x1049 to 3.4x1038 1.4×10^{-45} to3.4×10^{38}

Each floating point value is broken down into three main components: the sign bit, the exponent, and the mantissa. 1.

Grouping:

This row explains the bit allocation for the 32-bit floating point format: 1 bit for the sign, 8 bits for the exponent, and 23 bits for the mantissa.

5.75: Representation: 1.4375 x 22- Sign Bit: 0 (positive) - Exponent: 2 +127=129,

which is 100000012- Mantissa: The binary representation of 0.4375 is10111000000000000000000002.

-5.75: Representation:-1.4375 x 22-Sign Bit:1(negative)-Exponent:2+127= 129, which is 100000012-Mantissa: The binary representation of 0.4375 is10111000000000000000000002.

0.15625:Representation:1.25x23-Sign Bit:0 (positive)-Exponent:-3+127= 124, which is 011111012- Mantissa: The binary representation of 0.25 is01000000000000000000002.

-0.15625: Representation:-1.25 x2-3-Sign Bit: 1(negative) - Exponent:-3+127=124, which is 011111012-Mantissa: The binary representation of 0.25 is 01000000000000000000002

This breakdown helps illustrate how floating point values are stored and manipulated in computer systems

Double Precision(64-bit):

In double precision, the exponent is represented using 11 bits. The exponent is stored in a biased form, with a bias of 1023. The range of the actual exponent values can be determined as follows:

Bias: 1023

Exponent range: The actual exponent values range from-1022 to +1023.

Therefore, the smallest and largest possible exponent values in double-precision are:

Minimum exponent:-1022

Maximum exponent:+1023

We can perform the same steps given for the single-precision, except the difference of the abovementioned values

The information about how real values is stored in computer memory help us.

understand the precision and limitations of digital computation. With this understanding of floating-point representation, it becomes possible to control and manipulate these numbers in different ways.

Q.3: Discuss the significance of binary arithmetic operations in computer systems, including their applications in programming and digital circuit design, and how they differ from decimal operations.

Ans. Binary Arithmetic Operations:

Arithmetic operations include addition, subtraction, multiplication and division, and are performed on two numbers at a time. Binary arithmetic operations are similar to decimal operations but follow binary rules. Here's a brief overview of the basic operations:

Addition: Binary addition uses only two digits 0 and 1.Here,we will learn how to add binary numbers and how to handle the addition of negative binary numbers.

Binary Addition Rules:

Binary addition follows these simple rules:

1. 0+0=0
2. 0+1=1
3. 1+0=1
4. 1+1=0 (with a carry of 1 to the next higher bit)

Example of Binary Addition:

Example 1:

1101 + 1101 + 1011

In this example:

- 1+1=0(carry 1)
- 0+1+1(carry)=0(carry 1)
- 1+0+1(carry)=0(carry 1)
- 1+1+1(carry)=1(carry 1)

Subtraction:

In binary arithmetic, subtraction can also be carried out by adding the two's *complement* or the value of the subtrahend to the minuend.

Example:

Subtract 6 from 9 in Binary

Minuend = 9_{10} = 1001_2

Subtrahend = 6_{10} = 0110_2

Step 1:

Find the Two's Complement of the Subtrahend:

Invert the bits of 0110_2:

Inversion: 1001_2

Add 1 to the inverted number:

$1001_2 + 1_2 = 1010_2 = -6_{10}$

Step 2:

Add the Minuend and the Two's Complement of the Subtrahend

$1001_2 + 1010_2 = 10011_2$

Step 3:

Discard the Carry Bit

10011_2 Discard carry $0011_2 = 3_{10}$

So, $9 - 6 = 3$

Multiplication:

Binary numbers are base-2 numbers, consisting of only 0s and 1s. Multiplying binary numbers follows similar principles to multiplying decimal numbers, but with simpler rules. Here, we will learn how to multiply binary numbers with example.

Steps to Multiply Binary Numbers:

1. Write down the binary numbers, aligning them by the least significant bit (rightmost bit).
2. Multiply each bit of the second number by each bit of the first number, similar to the long multiplication method in decimal.
3. Shift the partial results one place to the left for each new row, starting from the second row.
4. Add all the partial results to get the final product.

Example:

Let us multiply two binary numbers: 101_2 and 11_2

$$\begin{array}{r} 101 \\ \times 11 \\ \hline 101 \\ 101\times \\ \hline 1111 \end{array}$$

So, $101_2 \times 11_2 = 11112$ (This is $101_2 \times 1_2$) 0 (This is $101_2 \times 1_2$ Shiftoed left)

$101_2 \times 11_2 = 1111_2$

Binary Division:

The Central Processing Unit (CPU) of a computer performs millions of binary multiplications every second to execute complex instructions and run programs.

Binary division is similar to decimal division but only involves two digits: 0 and 1.

It follows steps like comparing, subtracting, and shifting, akin to long division in the decimal system.

Steps of Binary Division:

Compare: Compare the divisor with the current portion of the dividend.

Subtract: Subtract the divisor from the dividend portion if the divisor is less than or equal to the dividend.

Shift: Shift the next binary digit from the dividend down to the remainder.

Repeat: Repeat the process until all digits of the dividend have been used.

Example:

(Step 1: Compare 10 with first two 11, subtract 10 from 11)
(Step 2: Bring down the next digit 0)
(Step 3: Compare 10 with 10, subtract 10 from 10)
(Step 4: Bring down the next digit 0, no more digits left)

Q.4: Discuss the significance of the encoding schemes in enabling seamless data communication and storage across diverse platforms and applications.

Ans. Common Text Encoding Schemes:

Text encoding schemes are essential for representing characters from various languages and symbols in a format that computers can understand and process. Here are some of the most common text encoding schemes used in computers:

ASCII: ASCII is an acronym that stands for American Standard Code for Information Interchange. It is a character encoding standard adopted for representing in devices such as computers and similar systems that use text. Each alphabet, number or symbol is given a code number between 0 and 127. ASCII enables different computers and devices to exchange text information reliably.

Let's encode the name of our country using ASCII.

1. The ASCII code for an upper case letter "P" is 80.
2. The code for letter 'a' in ASCII is 97.
3. The ASCII code for the letter 'k' is 107.
4. It is interesting to know that the ASCII code for the letter 'i' is 105.
5. In the ASCII code system, the letter 's' has a code of 115.
6. The code for 't' is 116 in ASCII.

The ASCII code is a numerical representation of characters in computer-based system, particularly for alphabetic characters.

For Example: the ASCII code of the character 'n' is 110.

Extended ASCII:

While the *standard* ASCII Table includes 128 characters, there is an extended version that *includes* 256 characters. This extended ASCII uses 8 bits and includes additional symbols, accented letters, and other characters. However, the original128 characters are the most commonly used and serves as the basis for text representation in computers.

Unicode: Unicode is an attempt at mapping all graphic characters used in any of the world's writing system. Unlike ASCII, which is limited to 7bits and can represent only 128 characters, Unicode can represent over a mill
12ion characters through different forms of encodings such as, UTF-8, UTF-16, and UTF-32. UTF is an acronym that stands for Unicode Transformation Format.

UTF-8:

It is a variable-length encoding scheme, meaning it can use a different number sof bytes (from 1 to 4) to represent a character. UTF-8 is backward compatible with ASCII. It means it can understand and use the older ASCII encoding scheme without any

problems .Therefore, if we have a text file written in ASCII, it will work perfectly fine with UTF-8, allowing it to read both old and new texts.

Example: The letter 'A' is Unicode, represented as, U+0041,is 01000001 in the binary format and occupies 8 bits or 1 byte.

Let's look at how Urdu letters are represented in UTF-8:

Example: The Urdu letter is represented in Unicode as U+0628; its binary format is 11011000 10101000,means it takes 2-bytes.

UTF-16:

UTF-16 is another variable character encoding mechanism, although it uses either 2 bytes or 4 bytes per character at most. Unlike UTF-8, it is not compatible with ASCII, meaning it cannot translate ASCII code.

Example: The letter A in UTF-16 is equal to 00000000 01000001 in binary or 65 in decimal(2 bytes).

For Urdu:

Example: The right Urdu letter in UTF-16 is represented as is 0000011000101000 in binary, which occupies 2 bytes of memory

UTF-32:

UTF-32 is a method of encoding that uses a fixed length, with all characters stored in 4 bytes per character. This makes it very simple but at the same time it may look a little complicated when it comes to space usage.

Example: Alphabet letter 'A' in UTF-32 is represented in binary as 00000000 00000000 00000000 01000001 which is 4 bytes.

Summary

- In computing, numbering systems are crucial as they form the foundation for representing, storing, and processing information.
- Decimal number system is a number system in which base is 10 and the digits involved are 0 to 9, which are commonly used in our daily lives.
- Binary is a base-2 number system that comprises of only the digits 0 and 1 Each digit represents a power of two.
- The Octal number system is another number system that has eight as its base; thus, it has eight digits 0 to7. Each digit represents a power of 8, this can be expressed as 8 digit.
- The Hexadecimal numbering system is another type of number system with base of 16, where the number 0 to 9 and alphabets A-F are used.
- Integers refers to the set of non-negative whole numbers, while whole numbers are the complete numbers. They include zero and all the positive integers, also positive zero.
- To store negative values, computers employ a technique commonly known as two's complement.

- In computers, real values, which are nicknamed as floating-point numbers are used to represent numbers with fraction or decimal point.
- Arithmetic operations mean addition, subtraction multiplication, and division performed on numbers in given base. Binary arithmetic involves performing these operations on numbers in binary form, or base 2.
- ASCII is an acronym for American Standard Code for Information Interchange. It is an industry standard used to encode text in computers and other devices.
- All the letters, digits or symbols are given a unique number ranging from 0 to 127.
- According to the standard ASCII table, there are both a basic version with 128 characters and an extending version with 256 characters.
- Unicode is a character encoding standard that aims to provide character codes for all the characters use a in the worlds' writing systems.
- UTF-16 on the other hand is another variable character encoding technique in which characters are represented using either two or four bytes.
- UTF-32 is a blocking encoding method of fixed block size, meaning that the size of each block is constant. Each single character is always represented using 4 bytes, regardless of its specific characteristics.
- Images under digital context are composed of small points referred to as pixel. Each of them is of certain color and the combination of such pixels makes a complete image or picture.
- Audio files are recorded using the human-computer interface to capture and then converting sound waves into digital form. Some of the key steps in this process include data sampling and quantization.

Additional MCQs

2.1 NUMBERING SYSTEMS

1. Which of the following is NOT a numbering system commonly used in computers?

(a) Decimal (b) Binary (c) Hexadecimal (d) Quaternary

2. What is The base of the decimal numbering system is:

(a) 2 (b) 8 (c) 10 (d) 16

3. What is the base of the binary numbering system?

(a) 2 (b) 8 (c) 10 (d) 16

4. What is the decimal equivalent of the binary number 1011_2?

(a) 5 (b) 11 (c) 13 (d) 15

5. **Which of the following is NOT a characteristic of the binary numbering system?**
   (a) It uses only two digits, 0 and 1. (b) It is used in digital circuits.
   (c) It is easy to represent two states, on and off.
   (d) It is more efficient than decimal for computer operations.
6. **What is the binary equivalent of the decimal number 5_{10}?**
   (a) 101_2 (b) 100_2 (c) 111_2 (d) 1001_2
7. **Which of the following is an advantage of using the binary system in computers?**
   (a) It is easier to understand than the decimal system.
   (b) It is more efficient for representing numbers.
   (c) It is more accurate for representing numbers.
   (d) It is easier to implement in digital circuits.
8. **What is the decimal equivalent of the binary number 10000000?**
   (a) 128 (b) 256 (c) 512 (d) 64
9. **Which of the following conversions is correct?**
   (a) Binary 1110 to Decimal is 14 (b) Binary 1010 to Decimal is 10
   (c) Binary 1101 to Decimal is 13 (d) All of the above
10. **What is the hexadecimal equivalent of the decimal number 255?**
    (a) FF (b) F0 (c) 00 (d) 1F
11. **Which of the following is true about the octal numbering system?**
    (a) It uses eight digits: 0-7. (b) It is base 10.
    (c) It is primarily used in modern computing.
    (d) It is less efficient than binary.
12. **What is the binary representation of the hexadecimal number A3?**
    (a) 10100011 (b) 11010011 (c) 10011011 (d) 10110011
13. **Which of the following is NOT a valid binary number?**
    (a) 101010 (b) 11012 (c) 111000 (d) 1001
14. **What is the result of adding the binary numbers 1011 and 1101?**
    (a) 11000 (b) 10100 (c) 10010 (d) 11100
15. **In a binary system, what does the leftmost bit represent?**
    (a) The least significant bit (b) The most significant bit
    (c) The middle bit (d) The parity bit
16. **What is the decimal equivalent of the hexadecimal number 1C?**
    (a) 28 (b) 30 (c) 26 (d) 24
17. **Which of the following is a characteristic of the hexadecimal system?**
    (a) It uses 16 symbols: 0-9 and A-F. (b) It is base 8.
    (c) It is less compact than binary.
    (d) It is primarily used for arithmetic operations.

18. **What is the binary equivalent of the decimal number 15_{10}?**
(a) 1110 (b) 1111 (c) 1101 (d) 1011

19. **Which of the following conversions is incorrect?**
(a) Decimal 10 to Binary is 1010 (b) Decimal 8 to Octal is 10
(c) Decimal 16 to Hexadecimal is 10 (d) Decimal 5 to Binary is 101

20. **What is the result of subtracting the binary number 1010 from 1100?**
(a) 0010 (b) 0100 (c) 1000 (d) 1110

21. **In the binary system, what does the digit "1" represent?**
(a) OFF (b) ON
(c) Both ON and OFF (d) Neither ON nor OFF

22. **What is the lowest level at which data is represented in a computer system?**
(a) Decimal (b) Binary (c) Octal (d) Hexadecimal

23. **Which of the following is the correct binary representation of the decimal number 83?**
(a) 10101001 (b) 1010011 (c) 10100101 (d) 1101001

24. **In the process of converting decimal to binary, what is the significance of the remainders?**
(a) The remainders are ignored.
(b) The remainders represent the binary digits.
(c) The remainders determine the quotient.
(d) The remainders are used to calculate the decimal equivalent.

25. **When converting decimal to binary, the remainders are read from:**
(a) Top to bottom (b) Bottom to top (c) Left to right (d) Right to left

26. **Which of the following is NOT a type of data that can be represented in binary format?**
(a) Numbers (b) Text
(c) Images (d) None of the above.

27. **How many digits are used in the binary number system?**
(a) 8 (b) 10 (c) 2 (d) 16

28. **What is the binary equivalent of the decimal number 10?**
(a) 1010 (b) 1010 (c) 1100 (d) 1110

29. **The conversion from decimal to binary is based on the principle of:**
(a) Repeated division by 2. (b) Repeated multiplication by 2.
(c) Repeated addition by 3. (d) Addition of powers of 2.

30. **Which of the following is a valid binary number?**
(a) 102 (b) 1102 (c) 1010 (d) 12

31. **In binary addition, what is the result of 1 + 1?**
(a) 0 (b) 10 (c) 1 (d) 11

32. **The purpose of a binary search algorithm is:**
(a) To sort data. (b) To find an item in a sorted list.
(c) To merge two lists. (d) To delete an item from a list.

33. **Which of the following operations is NOT performed in binary arithmetic?**
(a) Addition (b) Subtraction
(c) Division by zero (d) Multiplication

34. **What is the result of the binary operation 1101 - 101?**
(a) 1000 (b) 1000 (c) 1100 (d) 1110

35. ***In a binary tree, what is the maximum number of nodes at level 'n'?***
(a) n (b) 2^n (c) 2^n (d) n^2

36. **Which of the following is a characteristic of binary numbers?**
(a) They can only represent whole numbers.
(b) They can represent both whole numbers and fractions.
(c) They are always positive.
(d) They cannot be used in computing.

37. **What is the base of the octal system?**
(a) 8 (b) 2 (c) 10 (d) 16

38. **Which of the following is the correct binary equivalent of the octal number 157?**
(a) 1010101 (b) 1011111 (c) 1111111 (d) 1110000

39. **How many binary digits (bits) are represented by a single octal digit?**
(a) 1 (b) 2 (c) 3 (d) 4

40. **What is the octal equivalent of the binary number 110101011?**
(a) 1253 (b) 653 (c) 653 (d) 1235

41. **Which of the following conversions is incorrect?**
(a) 1010 (binary) = 10 (decimal)
(b) 111 (binary) = 7 (decimal)
(c) 12 (octal) = 10 (decimal)
(d) 100 (binary) = 4 (decimal)

42. **What is the decimal equivalent of the binary number 1010?**
(a) 8 (b) 10 (c) 12 (d) 14

43. **Which of the following is the correct binary equivalent of the octal number 157?**
(a) 1010101 (b) 1011111 (c) 1111111 (d) 1110000

44. **How many binary digits (bits) are represented by a single octal digit?**
(a) 1 (b) 2 (c) 3 (d) 4

45. **What is the octal equivalent of the binary number 110101011?**
(a) 1253 (b) 653 (c) 653 (d) 1235

46. **Which of the following activities involves converting numbers from decimal to binary?**
(a) Clock Time Conversion (b) Marks Conversion
(c) Sleeping Time Conversion (d) All of the above

47. **What is the binary equivalent of the decimal number 85?**
(a) 1010101 (b) 1101001 (c) 1001101 (d) 1110001

48. What is the binary equivalent of 3:45 PM in the Clock Time Conversion activity?

(a) 1111111111 (b) 1111101101 (c) 111000111 (d) 110111101

49. Which of the following is a characteristic of the hexadecimal system?

(a) Uses digits 0-9 only (b) Uses digits 0-7 only

(c) Uses digits 0-9 and letters A-F (d) Uses digits 0-15 only

50. What is the decimal equivalent of the binary number 1010?

(a) 8 (b) 10 (c) 12 (d) 14

51. In which system is the number 1A represented?

(a) Decimal (b) Binary (c) Hexadecimal (d) Octal

52. What is the result of adding the binary numbers 1011 and 1101?

(a) 11000 (b) 11000 (c) 10100 (d) 11110

53. Which of the following is not a valid octal digit?

(a) 0 (b) 5 (c) 7 (d) 8

54. What is the hexadecimal equivalent of the decimal number 255?

(a) FF (b) FF (c) 00 (d) 0F

55. Which of the following represents the largest value in binary?

(a) 1111 (b) 1010 (c) 11111 (d) 1000

56. Why is the octal number system not commonly used in modern computers?

(a) Because it is too complex

(b) Because it is not efficient for processing large numbers

(c) Because it is not compatible with the binary system

(d) Because it is not a standard system

57. Which of the following operations is not valid in the octal number system?

(a) Addition (b) Subtraction (c) Multiplication (d) Division by zero

58. Which of the following is true about octal numbers?

(a) They can only represent whole numbers.

(b) They can represent fractions.

(c) They are always longer than binary numbers.

(d) They are used in hexadecimal calculations.

59. Convert the decimal number 64 to octal.

(a) 100 (b) 110 (c) 120 (d) 140

60. Which number system was used in early computing systems like PDP-8?

(a) Decimal (b) Octal

(c) Binary (d) Hexadecimal

61. Why was the octal system preferred in early computing systems?

(a) Easier conversion between octal and decimal

(b) Easier conversion between octal and binary

(c) Easier conversion between decimal and binary (d) All of the above

62. What is the base of the hexadecimal number system?

(a) 2 (b) 8 (c) 10 (d) 16

63. **Which of the following is NOT a hexadecimal digit?**
(a) 9 (b) A (c) Z (d) F

64. **How many binary bits does a single hexadecimal digit represent?**
(a) 2 (b) 8 (c) 3 (d) 4

65. **What is the purpose of adding zeros to the left of a binary number when converting to hexadecimal?**
(a) To make the binary number easier to read
(b) To ensure the binary number has an even number of digits
(c) To group the binary digits into groups of four
(d) To maintain the value of the binary number

66. **How many bits are in a byte?**
(a) 4 (b) 8 (c) 16 (d) 32

67. **Which of the following is NOT a whole number?**
(a) 0 (b) 1 (c) -1 (d) 2

2.2 DATA REPRESENTATION IN COMPUTING SYSTEMS

68. **What is the purpose of binary encoding in computer systems?**
(a) To represent text characters (b) To represent images
(c) To represent numbers (d) All of the above

69. **Which of the following is an example of a quantity that cannot be negative?**
(a) Temperature (b) Altitude
(c) Bank balance (d) Number of students in a school

70. **What is the purpose of representing data in computers?**
(a) To store and process information
(b) To display images (c) To create sounds
(d) To connect to the Internet

71. **What is the maximum value that can be represented by a 1-byte integer?**
(a) 127 (b) 255 (c) 65,535 (d) 4,294,967,295

72. **Which of the following is the correct representation of the set of integers?**
(a) Z = (..., -3, -2, -1,0,1,2,3,...) (b) Z = (0, 1, 2, 3,...)
(c) Z = (..., -2, -1, 0, 1, 2, 3, ...) (d) Z = (-3, -2, -1, 0, 1, 2, 3)

73. **In computer programming, what are signed integers referred to as?**
(a) Integers (b) Whole Numbers
(c) Natural Numbers (d) Floating-Point Numbers

74. **Which bit is used to represent the sign of a value in a signed integer?**
(a) Least Significant Bit (LSB) (b) Most Significant Bit (MSB)
(c) Middle Bit (d) None of the above

75. **What method is commonly used to store negative values in computers?**
(a) One's Complement (b) Two's Complement
(c) Sign-Magnitude (d) Excess-N

76. Which of the following is a characteristic of floating-point representation?
(a) Fixed number of decimal places
(b) Variable precision
(c) Only positive values
(d) Integer-only representation

77. What is the primary purpose of the stack in computer memory?
(a) Store static data
(b) Manage function calls and local variables
(c) Hold global variables (d) Store large datasets

78. Which of the following data structures uses LIFO (Last-In, First-Out) principle?
(a) Queue (b) Stack (c) Array (d) Linked List

79. What is the time complexity of accessing an element in an array?
(a) O(n) (b) O(log n) (c) O(1) (d) O(n2)

80. What is the primary function of an operating system?
(a) To manage hardware resources (b) To compile programs
(c) To design user interfaces (d) To create databases

81. Which of the following is not a type of software?
(a) System Software (b) Application Software
(c) Middleware (d) Hardware

82. What is the minimum value that can be represented by a 4-byte integer (32 bits)?
(a) 128 (b) -2,147,483,648 (c) –32, 768 (d) 2,147,483,647

83. What is the primary purpose of storing real values in computer memory?
(a) To represent whole numbers only.
(b) To represent numbers with fractions and/or decimals.
(c) To store text and characters. (d) To store images and videos.

84. What is the process of converting a fractional part of a decimal number to binary called?
(a) Decimalization (b) Binary conversion
(c) Fractionalization (d) Floating-point conversion

85. When converting a fractional part of a decimal number to binary, what is the stopping condition?
(a) The fractional part becomes zero. (b) The fractional part becomes one.
(c) The required number of decimal places is achieved.
(d) Both a and c.

86. What is the integral part of the decimal number 4.625?
(a) 0.625 (b) 4 (c) 4.625 (d) 625

87. **Which of the following is the first step in converting a fractional part of a decimal number to binary?**
(a) Multiply the fractional part by 2.
(b) Write down the integer part of the result.
(c) Repeat the process with the new fractional part.
(d) Add the integer parts together.

88. **What is the purpose of the exponent in scientific notation for floating-point numbers?**
(a) To indicate the sign of the number.
(b) To determine the precision of the number.
(c) To scale the number by a power of ten.
(d) To represent the integer part of the number.

89. **What are the two commonly used standards for representing real numbers in binary form?**
(a) Single Precision (16-bit) and Double Precision (32-bit)
(b) Single Precision (32-bit) and Double Precision (64-bit)
(c) Single Precision (64-bit) and Double Precision (128-bit)
(d) Single Precision (128-bit) and Double Precision (256-bit)

90. **What is the approximate range of values that can be represented using a 32-bit floating-point representation?**
(a) 1.4×10^{-44} to 3.4×10^{-38} (b) 1.4×10^{-38} to 3.4×10^{-44}
(c) 1.4×10^{-44} to $3.4 \times 10^{+38}$
(d) $1.4 \times 10^{+38}$ to 3.4×10^{-44}

91. **What is the purpose of the exponent in a floating-point number?**
(a) To determine the sign of the number.
(b) To scale the mantissa.
(c) To count the number of significant digits.
(d) To represent the integer part of the number.

92. **What is the bias used in double precision floating-point representation?**
(a) 127 (b) 1023 (c) 255 (d) 511

93. **What is the range of the actual exponent values in double precision?**
(a) −126 to +127 (b) −1022 to +1023 (c) −127 to +128 (d) −254 to +255

94. **What is the smallest positive number representable in single precision?**
(a) 1.4×10^{-45} (b) 4.9×10^{-324} (c) 2.2×10^{-16} (d) 1.1×10^{-1}

95. **What is the main purpose of the sign bit in floating-point representation?**
(a) To represent the exponent of the number.
(b) To represent the mantissa of the number.
(c) To indicate whether the number is positive or negative.
(d) To indicate whether the number is a whole number or a fraction.

96. What is the total number of bits in a double precision floating-point number?

(a) 32 (b) 64 (c) 128 (d) 16

2.4 BINARY ARITHMETIC OPERATIONS

97. What is the two's complement of the binary number 0110?

(a) 1001 (b) 1010 (c) 1011 (d) 1100

98. Which of the following steps is NOT involved in multiplying binary numbers?

(a) Write down the binary numbers, aligning them by the least significant bit

(b) Multiply each bit of the second number by each bit of the first number

(c) Add all the partial results to get the final product

(d) Convert the partial results to decimal before adding them

99. What is the binary product of 101 and 11?

(a) 1001 (b) 1100 (c) 1111 (d) 1011 Answer:

2.5 COMMON TEXT ENCODING SCHEMES

100. What does ASCII stand for?

(a) American Standard Code for Information Interchange

(b) Advanced Standard Code for Information Interchange

(c) American Standard Code for Internet Interchange

(d) Advanced Standard Code for Internet Interchange

101. What is the ASCII code for an upper case letter "P"?

(a) 78 (b) 80 (c) 97 (d) 105

102. How many characters can be represented using ASCII?

(a) 128 (b) 256 (c) 1024 (d) Unlimited

103. How many bits are required to represent a single ASCII character?

(a) 4 (b) 7 (c) 8 (d) 16

104. What is the purpose of the ASCII code?

(a) To represent text in computers. (b) To encode images.

(c) To compress files. (d) To encrypt data.

105. What does the acronym "UTF" stand for?

(a) Unicode Transformation Format (b) Universal Text Format

(c) Universal Text File (d) Unicode Transfer Format

106. Which of the following statements is true regarding UTF-8?

(a) It can only encode characters from the Latin alphabet.

(b) It uses a variable number of bytes for different characters.

(c) It is not compatible with ASCII.

(d) It requires 4 bytes for all characters.

107. What is the maximum number of characters that can be represented in Unicode?

(a) 65,536 (b) 1,114,112 (c) 256 (d) 128

108. In a color image, how are colors represented?
(a) Using a single number for each pixel.
(b) Using three numbers RGB for each pixel.
(c) Using letters to represent each color.
(d) Using a combination of shapes and lines.

109. Which image file format is best for simple animations and images with few colors?
(a) JPEG (b) PNG (c) GIF (d) All of the above

110. What does DPI stand for in the context of image resolution?
(a) Digital Pixels Index (b) Dots Per Inch
(c) Data Processing Interface (d) Dynamic Picture Integration

111. Which of the following is a common use of raster images?
(a) Logos (b) Illustrations (c) Photographs (d) Fonts

112. In digital imaging, what does the term "resolution" refer to?
(a) The color depth of an image (b) The clarity and detail of an image
(c) The file size of an image (d) The format of an image

113. Which of the following audio file formats is uncompressed?
(a) MP3 (b) WAV (c) AAC (d) All of the above

114. Which of the following units is used to measure the size of an image file?
(a) Kilobytes (KB) (b) Megabytes (MB) (c) Gigabytes (GB) (d) Terabytes (TB)

Answers

1. (d)	2. (c)	3. (a)	4. (b)	5. (d)	6. (a)	7. (d)
8. (a)	9. (d)	10. (a)	11. (a)	12. (a)	13. (b)	14. (a)
15. (b)	16. (a)	17. (a)	18. (b)	19. (c)	20. (b)	21. (b)
22. (b)	23. (b)	24. (b)	25. (b)	26. (d)	27. (c)	28. (b)
29. (a)	30. (c)	31. (b)	32. (b)	33. (c)	34. (b)	35. (c)
36. (b)	37. (a)	38. (b)	39. (c)	40. (c)	41. (c)	42. (b)
43. (b)	44. (c)	45. (c)	46. (d)	47. (a)	48. (c)	49. (c)
50. (b)	51. (c)	52. (b)	53. (d)	54. (b)	55. (c)	56. (c)
57. (d)	58. (b)	59. (a)	60. (b)	61. (b)	62. (d)	63. (c)
64. (d)	65. (c)	66. (b)	67. (c)	68. (c)	69. (d)	70. (a)
71. (b)	72. (a)	73. (a)	74. (b)	75. (b)	76. (b)	77. (b)
78. (b)	79. (c)	80. (a)	81. (d)	82. (b)	83. (b)	84. (b)
85. (d)	86. (b)	87. (a)	88. (c)	89. (b)	90. (c)	91. (b)
92. (b)	93. (b)	94. (a)	95. (c)	96. (b)	97. (b)	98. (d)
99. (c)	100. (a)	101. (b)	102. (a)	103. (b)	104. (a)	105. (a)
106. (b)	107. (b)	108. (b)	109. (c)	110. (d)	111. (c)	112. (b)
113. (b)	114. (a)					

1. **In which system is the number 1A represented?**
   (a) Decimal (b) Binary (c) Hexadecimal (d) Octal
2. **Which of the following is not a valid octal digit?**
   (a) 0 (b) 5 (c) 7 (d) 8
3. **Which of the following activities involves converting numbers from decimal to binary?**
   (a) Clock Time Conversion (b) Marks Conversion
   (c) Sleeping Time Conversion (d) Both a and b
4. **Which number system was used in early computing systems like PDP-8?**
   (a) Decimal (b) Octal
   (c) Binary (d) Hexadecimal
5. **Which of the following values is NOT expressed in hexadecimal?**
   (a) Minimum Age to Cast Vote (b) Length of the Indus River
   (c) Total Districts in Pakistan (d) Height of K2
6. **Which of the following is a characteristic of floating-point representation?**
   (a) It can only represent whole numbers.
   (b) It allows for a wide range of values, including very small and very large numbers.
   (c) It is less efficient than fixed-point representation.
   (d) It uses a fixed number of bits for the integer part only.
7. **In the context of floating-point representation, what does normalization refer to?**
   (a) Adjusting the sign of the number.
   (b) Ensuring the mantissa is within a specific range.
   (c) Converting the number to an integer.
   (d) Rounding the number to the nearest whole number.
8. **Which of the following is a limitation of using floating-point representation?**
   (a) It can represent all real numbers exactly.
   (b) It can introduce rounding errors.
   (c) It requires less memory than integer representation.
   (d) It is faster than integer arithmetic.
9. **What is the main purpose of the sign bit in floating-point representation?**
   (a) To represent the exponent of the number.
   (b) To represent the mantissa of the number.
   (c) To indicate whether the number is positive or negative.
   (d) To indicate whether the number is a whole number or a fraction.
10. **Which of the following describes the term "bitrate"?**
    (a) The number of bits processed per unit of time.
    (b) The total number of bits in a file.

(c) The quality of the audio signal.
(d) The duration of the audio file.

Answers

1.	(c)	**2.**	(d)	**3.**	(b)	**4.**	(b)	**5.**	(d)	**6.**	(b)	**7.**	(b)
8.	(b)	**9.**	(c)	**10.**	(a)								

Additional Short Questions

2.1 NUMBERING SYSTEMS

1. Define numbering system.

Ans: Numbering Systems:

Numbering systems are essential in computing because they form the basis for representing, storing, and processing data. Different numbering systems help computers perform tasks like calculations, data storage, and data transfer. These systems allow computers to represent various kinds of information, such as text, colors, and memory locations.

Here are four numbering systems:

1. Decimal System
2. Binary System
3. Octal System
4. Hexadecimal System

2. Define decimal number system with example.

Ans: Decimal System:

The decimal number system is a base-10 number system that consists of digit from 0 to 9 and we use it in everyday life. That is why each digit of the number represents a power of 10. In the decimal system the place values starting from the rightmost digits are 10^0, 10^1, 10^2, and so on. For Example: the decimal number 523 means:

$5 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 = 500 + 20 + 3 = 523$

3. Define binary number system with example.

Ans: Binary System:

In binary, the place values are arranged from the right to left, starting with 2^0, and ending at 2^n, where each position represents a power of 2.

For Example: the binary number 1011 can be converted to decimal as follows:

$= 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

$= 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$

$= 8 + 0 + 2 + 1$

$= 11_{10}$

4. What are the different types of number systems discussed?

Ans: Here are four numbering systems:

1. Decimal System
2. Binary System
3. Octal System
4. Hexadecimal System

5. What is the significance of the binary system in computers?

Ans: Significance:

Computers work in binary system especially because this method fits well with electronics. Digital circuits have two states: They can be either on or off. These states are easily represented by the binary digits: 1 represents ON, and 0 represents OFF. When typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

When you type a letter on your keyboard, the computer converts it into a binary code. Similarly, all types of data, including numbers, text, images, and sounds, are ultimately broken down into binary code

6. Write the types of digital circuits.

Ans: Digital circuits have two states: They can be either on or off. These states are easily represented by the binary digits: 1 represents ON, and 0 represents OFF. When typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

When you type a letter on your keyboard, the computer converts it into a binary code. Similarly, all types of data, including numbers, text, images, and sounds, are ultimately broken down into binary code

7. How are the states ON and OFF represented in the binary system?

Ans: Digital circuits have two states: They can be either on or off. These states are easily represented by the binary digits: 1 represent ON, and 0 represents OFF. When typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

8. Write the algorithm to convert decimal into binary Or Explain the steps involved in converting a decimal number to binary.

Ans: Conversion from Decimal to Binary:

The following algorithm translates a decimal number to binary.

1. To convert decimal number to binary form, divide the decimal number by 2.
2. Record the remainder.
3. Divide the number by 2 until the quotient which is left after division is 0
4. Meaning it is represented by the remainders and it is read from the bottom to the top of the binary number.

9. *What is* binary code?

Ans: In binary code, each digit is called a "bit" and can have a value of either 0 or 1. These bits are combined to form "bytes," which are groups of 8 bits that represent a single character or number. Binary code is the foundation of all computer programming.

10. What is the binary equivalent of the decimal number 83?

Ans: Convert 83 to binary.

83/2 = 41 remainder 1

41/2=20 remainder 1

20/2 = 10 remainder 0

10/2= 5 remainder 0

5/2 =2 remainder 1

2/2=1=1 remainder 0

1/2=0= 0 remainder 1

If the remainders are read from bottom to top then it gives the required result in binary, which is 1010011.

11. Give an example of a real-world application of the octal number system.

Ans: 1-Color Codes in Web Design

In web design, colors can be represented using octal codes. For example, the color white can be represented as #FFFFFF in hexadecimal, but in octal, it would be 377777.

2-Octal codes can be used to create simple codes and ciphers. For example, substituting each letter with its corresponding octal code can create a basic encryption.

12. What is the octal equivalent of the binary number 110101011?

Ans: 110101011.

This number can be divided into groups of three

Bits from right to left: 110 101 011

Each group of three bits corresponds to a single octal digit:

110=6

101=5

011=3

So, the binary number 110101011 is equal to 653 in octal.

13. Define octal number system.

Ans: Octal System:

Octal is a positional numeral system with base eight, which implies that a digit to be used ranges from 0 to 7. The last digit is a single digit power of 8 while the other digits are the coefficients. In the decimal system, the place values starting from the 8^0,8^1, 8^2 and so on.

Example, the octal number 157 means:

$1 \times 8^2 + 5 \times 8^1 + 7 \times 8^0$

$= 1 \times 64 + 5 \times 8 + 7 \times 1$

14. What is the binary equivalent of the octal number 157?

Ans: The octal number 157 equivalent

$1 \times 8^2 + 5 \times 8^1 + 7 \times 8^0$

$= 1 \times 64 + 5 \times 8 + 7 \times 1$

15. Describe the steps involved in converting a decimal number to its octal equivalent.

Ans: Conversion from Decimal to Octal:

The algorithm below translates a decimal number into an octal.

1. To convert the decimal number to an equivalent octal number, divide the number by 8.
2. Write down the remainder.
3. After that divide the obtained quotient by 8.
4. Continue the divisions until one of the numbers results in 0.
5. Octal is a base eight number and the octal number is the remainder read from the bottom up to the top.

16. Work in pairs to convert the following decimal numbers to octal:45, 128 , 64.2.

Ans: Decimal to Octal

1. - 45 (decimal) to Octal:
   - Divide 45 by 8: 45 ÷ 8 = 5 remainder 5
   - Write the result as: 55
2. 128 (decimal) to Octal:
   - Divide 128 by 8: 128 ÷ 8 = 16 remainder 0
   - Divide 16 by 8: 16 ÷ 8 = 2 remainder 0
   - Write the result as: 200
   - Convert the whole number part (64):
   - Divide 64 by 8: 64 ÷ 8 = 8 remainder 0
   - Write the result as: 100
3. - Convert the fractional part (.2):
   - Multiply 0.2 by 8: 0.2 × 8 = 1.6
   - Take the whole number part (1) and multiply the fractional part (0.6) by 8 again: 0.6 × 8 = 4.8
   - Take the whole number part (4) and multiply the fractional part (0.8) by 8 again: 0.8 × 8 = 6.4
4. - Write the fractional part as: 0.146 (approx.)
   - Combine the whole number and fractional parts: 100.146 (approx.)

17. Convert these octal numbers to decimal: 57,124,301.

Ans; Octal to Decimal

1- 57 (octal) to Decimal:
- Multiply each digit by the corresponding power of 8 and add:
- $5 \times 8^1 = 40$
- $7 \times 8^0 = 7$
- Add: 40 + 7 = 47

2- 124 (octal) to Decimal:
- Multiply each digit by the corresponding power of 8 and add:
- $1 \times 8^2 = 64$

- $2 \times 8^1 = 16$
- $4 \times 8^0 = 4$
- Add: $64 + 16 + 4 = 84$

3- 301 (octal) to Decimal:
- Multiply each digit by the corresponding power of 8 and add:
- $3 \times 8^2 = 192$
- $0 \times 8^1 = 0$
- $1 \times 8^0 = 1$
- Add: $192 + 0 + 1 = 193$

18. Explain the process of converting 66 to octal using the algorithm.

Ans: Here is the step-by-step process to convert 66 to octal:

Step 1: Divide 66 by 8

$66 \div 8 = 8$ remainder 2

Step 2: Write the remainder as the rightmost digit

The remainder 2 is the rightmost digit of the octal number.

Step 3: Divide the quotient by 8 (if quotient is greater than 7)

Since the quotient 8 is greater than 7, divide it by 8:

$8 \div 8 = 1$ remainder 0

Step 4: Write the remainder as the next digit

The remainder 0 is the next digit of the octal number.

Step 5: Stop dividing since the quotient is less than 8

The quotient 1 is less than 8, so we stop dividing.

Step 6: Write the final octal number

Combine the digits: 102

The octal equivalent of 66 is 102.

19. Define hexadecimal number system with example.

Ans: Hexadecimal System:

The hexadecimal is a base 16 number system with digit number from 0 to 9 and alphabets from A to F; each digit represents 16 to the power of the position of the digit. The letters A to F stand for the numeric value of 10 to 15. The digits in hexadecimal move from right to left in place value that are 16^0, 16^1, 16^2...anothers.

For Example:

the hexadecimal number 1A3 can be represented in decimal

$1 \times 16^2 + A \times 16^1 + 3 \times 16^0 = 1 \times 256 + 10 \times 16 + 3 \times 1 = 256 + 160 + 3 = 419$

20. How can the hexadecimal number 1A3 be converted to its decimal equivalent?

Ans: The hexadecimal number 1A3 can be represented in decimal

$1 \times 16^2 + A \times 16^1 + 3 \times 16^0 = 1 \times 256 + 10 \times 16 + 3 \times 1 = 256 + 160 + 3 = 419$

21. What is the key reason for the hexadecimal system's usefulness in computer systems?

Ans: The hexadecimal number system is not directly used by computers either .However, it provides an even more compact representation than octal. This makes

it easier for us to read and write large binary numbers.

22. Write the steps how to convert decimal into hexadecimal with example.

Ans: Converting Decimal to Hexadecimal:

The following algorithm converts a decimal number to hexadecimal:

1. Convert the decimal number to an absolute value by dividing it by 16.
2. Record the quotient and the remainder.
3. Continue dividing the quotient by 16 and write down the remainder until the quotient is zero.
4. The hexadecimal number, as you might have guessed, is the remainder read from bottom to top.

2.2 DATA REPRESENTATION IN COMPUTING SYSTEMS

23. Briefly explain the difference between whole numbers and integers, emphasizing their relevance in computing.

Ans: Whole Numbers (W) and Integers (Z):

Whole Numbers(W): Whole numbers are a set of non-negative integers. They include zero and all the positive integers. Mathematically, the set of whole numbers is:

$W=\{0,1,2,3,...\}$

In computing, whole numbers are often used to represent quantities that can't be negative.

Examples:

The number of students in a school, a person's age in years, and grades, provided there are no negative figures such as credit point balances.

Integers(Z):

Integers extend the concept of whole numbers to include negative numbers. In computer programming, we call them signed integers. The set of integers is represented as:

$Z=\{...,-3,-2,-1,0,1;2,3,...\}$

To store both positive and negative values, one bit is reserved as the sign bit (the most significant bit). If the sign bit is ON(1), the value is negative; otherwise, it is positive. Using this system, the maximum positive value that can be stored in a 1byte signed integer is $(01111111)_2$, which is 127_{10}. As the bits available to stored a value is n -1, hence the maximum value will be $2^{n-1}-1$. We can use this formula to compute the maximum values for 2 and 4 bytes.

24. What is the significance of the sign bit in signed integers, and how does it affect the representation of positive and negative values?

Ans: To store both positive and negative values, one bit is reserved as the sign bit (the most significant bit). If the sign bit is ON(1), the value is negative; otherwise, it is positive. Using this system, the maximum positive value that can be stored in a 1byte signed integer is $(01111111)_2$, which is 127_{10},.As the bits available to stored

a value is n -1, hence the maximum value will be $2^{n-1}-1$. We can use this formula to compute the maximum values for 2 and 4 bytes.

25. Explain the concept of two's complement and its use in representing negative numbers.

Ans: Negative Values and Two's Complement:

To store negative values, computers use a method called two's complement. To find the two's complement of a binary number, follow these steps:

1. Invert all the bits (change Os to 1s and 1s to Os).
2. Add 1 to the Least Significant Bit (LSB).

Example:

Let us convert the decimal number -5 to an 8-bit binary number:

1. Start with the binary representation of 00000101_2.
2. Invert all the bits: 11111010_2.
3. Add1: $11111010_2 + 1_2 = 11111011_2$

So,-5 in 8-bit two's complement is 11111011_2.

26. How does the minimum value in a signed integer vary depending on the number of bits? Give an example.

Ans: Minimum Integer Value:

For an 8-bit integer, we switch on the sign bit for the negative value and turn all bits ON. resulting in 11111111_2,. Except the first bit, we take two's complement and get 10000000_2, which is 128_{10}. Thus minimum value in 1-byte signed integer is–128,i.e.,-2^7.The minimum value is computed using the formula -2^{n-1},where n is the total number of bits.

2 – Byte Integer(16 bits): Minimum value= $-2^{15} = -32,768$

4 – Byte Integer (32 bits): Minimum value = $-2^{31}-1 = -2,147,483,648$

2.3 STORING REAL VALUES IN COMPUTER MEMORY

27. What are the two main steps involved in converting a fractional part to binary?

Ans: 1. Identify the Fractional Part: Get the fractional part of the decimal number. For instance, in the number 4.625,the integral part is 4 and the fractional part is0.625.

2. Convert the Fractional Part to Binary: Multiply the fractional part by 2,and write down the integer that is obtained. Repeat this process with the new fractional part till it gets to 0 or until then required number of decimal places is achieved .

28. Define the terms

a. Floating point b. Single precision. c. Single bit. d. Exponent.

e. Mantissa f. Double precision

Ans: Single Precision(32-bit):

In this standard, 4 bytes (or 32 bits) are assigned where the 1st bit is the sign bit, and the next 8 bits are for the exponent and the remaining 23 bits are for the mantissa.

Here the exponent can be ranged between-126 and+127.
The approximate range of values from 1.4×10^{-45} to 3.4×10^{38}
Each floating point value is broken down into three main components: the sign bit, the exponent, and the mantissa.1.

Double Precision(64-bit):

In double precision, the exponent is represented using 11 bits. The exponent is stored in a biased form, with a bias of 1023. The range of the actual exponent values can be determined as follows:

Bias: 1023·

Exponent range: The actual exponent values range from-1022 to +1023.

Therefore, the smallest and largest possible exponent values in double-precision are:

Minimum exponent:-1022·

Maximum exponent:+1023

Exponent: The exponent is the power to which a base number is raised to represent a floating-point number. In floating-point representation, the exponent determines the magnitude of the number.

Floating-Point Number: A floating-point number is a numerical representation that uses a combination of a mantissa, exponent, and base to represent real numbers. It is a binary format that consists of three main parts: mantissa, exponent, and sign bit.

Mantissa: The mantissa (also known as the significant) is the fractional part of a floating-point number. It represents the significant digits of the number and provides the precision and accuracy of the floating-point representation.

2.4 BINARY ARITHMETIC OPERATIONS

29. Define binary addition with example.

Ans: Addition: Binary addition uses only two digits 0 and 1.Here,we will learn how to add binary numbers and how to handle the addition of negative binary numbers.

Binary Addition Rules:

Binary addition follows these simple rules:

1. 0 + 0 = 0
2. 0 + 1 =1
3. 0 + 0 =1
4. 1+1=0 (with a carry of 1 to the next higher bit)

Example of Binary Addition:

Example 1:

1101 + 1011

In this example:·

- 1+1=0(carry 1)
- 0+1+1(carry)=0(carry 1)·
- 1+0+1(carry)=0(carry 1)·

- 1+1+1(carry)=1(carry 1)

30. Define binary subtraction with example.

Ans: Subtraction: In binary arithmetic, subtraction can also be carried out by adding the two's complement or the value of the subtrahend to the minuend.

Example: Subtract 6 from 9 in Binary:

Minuend = $9_{10} = 1001_2$

Subtrahend = $6_{10} = 0110_2$

Step 1: Find the Two's Complement of the Subtrahend

Invert the bits of 0110_2:

Inversion: 1001_2

Add 1 to the inverted number:

$1001_2 + 1_2 = 1010_2 = -6_{10}$

Step 2: Add the Minuend and the Two's Complement of the Subtrahend

$1001_2 + 1010_2 = 10011_2$

Step 3: Discard the Carry Bit

10011_2 Discard carry $0011_2 = 3_2$

So, 9 - 6 = 3

31. Define binary multiplication with example.

Ans: Multiplication: Binary numbers are base-2 numbers, consisting of only 0s and 1s. Multiplying binary numbers follows similar principles to multiplying decimal numbers, but with simpler rules. Here, we will learn how to multiply binary numbers with example.

Steps to Multiply Binary Numbers:

1. Write down the binary numbers, aligning them by the least significant bit (rightmost bit).
2. Multiply each bit of the second number by each bit of the first number, similar to the long multiplication method in decimal.
3. Shift the partial results one place to the left for each new row, starting from the second row.
4. Add all the partial results to get the final product.

Example

Let us multiply two binary numbers: 101_2 and 11_2

$$\begin{array}{r} 101 \\ \times 11 \\ \hline 101 \\ 101\times \\ \hline 1111 \end{array}$$

So, $101_2 \times 11_2 = 1111_2$

32. Define binary division with example.

Ans: Binary Division:

Binary division is similar to decimal division but only involves two digits: 0 and 1. It follows steps like comparing, subtracting, and shifting, to long division in the

decimal system.

Steps of Binary Division:

Compare: Compare the divisor with the current portion of the dividend.

Subtract: Subtract the divisor from the dividend portion if the divisor is less than or equal to the dividend.

Shift: Shift the next binary digit from the dividend down to the remainder.

Repeat: Repeat the process until all digits of the dividend have been used.

Example:

Divide 1100_2 by 10_2

```
      110
   ┌─────
10 │ 1100
   │ -10
   │ ---
   │  10
   │ -10
   │ ---
   │   0
```

Result: $1100_2/10_2 = 110_2$

(Step 1: Compare 10 with first two 11, subtract 10 from 11)

(Step 2: Bring down the next digit 0)

(Step 3: Compare 10 with 10, subtract 10 from 10)

(Step 4: Bring down the next digit 0, no more digits left)

33. What is the fundamental principle behind binary subtraction using the two's complement method?

Ans: In binary arithmetic, subtraction can also be carried out by adding the two's complement or the value of the subtrahend to the minuend.

Example: Subtract 6 from 9 in Binary:

Minuend = $9_{10} = 1001_2$

Subtrahend = $6_{10} = 0110_2$

Step 1: Find the Two's Complement of the Subtrahend

Invert the bits of 0110_2:

Inversion: 1001_2

Add 1 to the inverted number:

$1001_2 + 1_2 = 1010_2 = -6_{10}$

Step 2: Add the Minuend and the Two's Complement of the Subtrahend

$1001_2 + 1010_2 = 10011_2$

Step 3: Discard the Carry Bit

10011_2 Discard carry $0011_2 = 3$ $0011_2 = 3_{10}$

So, 9 - 6 = 3

34. Write down the steps involved in binary subtraction.

Ans: Step 1: Find the Two's Complement of the Subtrahend

Invert the bits of 0110_2:

Inversion: 1001_2

Add 1 to the inverted number:

$1001_2 + 1_2 = 1010_2 = -6_{10}$

Step 2: Add the Minuend and the Two's Complement of the Subtrahend

$1001_2 + 1010_2 = 10011_2$

Step 3: Discard the Carry Bit

10011_2 Discard carry $0011_2 = 3_{10}$

So, $9 - 6 = 3$

35. What are the steps to follow for binary multiplication?

Ans: Steps to Multiply Binary Numbers:

1- Write down the binary numbers, aligning them by the least significant bit (rightmost bit).

2- Multiply each bit of the second number by each bit of the first number, similar to the long multiplication method in decimal.

3- Shift the partial results one place to the left for each new row, starting from the second row.

4- Add all the partial results to get the final product.

36. Write steps how to perform binary division?

Ans: Compare: Compare the divisor with the current portion of the dividend.

Subtract: Subtract the divisor from the dividend portion if the divisor is less than or equal to the dividend.

Shift: Shift the next binary digit from the dividend down to the remainder.

Repeat: Repeat the process until all digits of the dividend have been used.

2.5 COMMON TEXT ENCODING SCHEMES

37. Define ASCII and ASCII codes for some latters.

Ans: ASCII: ASCII is an acronym that stands for American Standard Code for Information Interchange. It is a character encoding standard adopted for representing in devices such as computers and similar systems that use text. Each alphabet, number or symbol is given a code number between 0 and 127.ASCII enables different computers and devices to exchange text information reliably.

Let us encode the name of our country using ASCII:

1- The ASCII code for an upper case letter "P" is 80.

2- The code for letter 'a' in ASCII is 97.

3- The ASCII code for the letter 'k' is 107.

4- It is interesting to know that the ASCII code for the letter 'i' is 105.

5- In the ASCII code system, the letter 's' has a code of 115.

6- The code for 't' is 116 in ASCII.

The ASCII code is a numerical representation of characters in computer-based system, particularly for alphabetic characters.

For Example: *the ASCII code of the character 'n' is 110.*

38. Define Extended ASCII.

Ans: Extended ASCII: While the standard ASCII Table includes 128 characters, there is an extended version that includes 256 characters. This extended ASCII uses 8

bits and includes additional symbols, accented letters, and other characters. However, the original 128 characters are the most commonly used and serves as the basis for text representation in computers.

39. Define Unicode.

Ans: Unicode: Unicode is an attempt at mapping all graphic characters used in any of the world's writing system. Unlike ASCII, which is limited to 7bits and can represent only 128 characters, Unicode can represent over a million characters through different forms of encodings such as, UTF-8, UTF-16, and UTF-32. UTF is an acronym that stands for Unicode Transformation Format.

40. Define UTF-8 with example.

Ans: UTF-8: It is a variable-length encoding scheme, meaning it can use a different numbers of bytes (from 1 to 4) to represent a character. UTF-8 is backward compatible with ASCII. It means it can understand and use the older ASCII encoding scheme without any problems .Therefore, if we have a text file written in ASCII, it will work perfectly fine with UTF-8,allowing it to read both old and new texts.

Example: The letter 'A' is Unicode, represented as, U+0041,is 01000001 in the binary format and occupies 8 bits or 1 byte.

Let us look at how Urdu letters are represented in UTF-8:

Example: The Urdu letter ب is represented in Unicode as U+0628; its binary format is 11011000 10101000,means it takes 2 bytes.

41. Define UTF-16 with example.

Ans: UTF-16: UTF-16 is another variable character encoding mechanism, although it uses either 2 bytes or 4 bytes per character at most. Unlike UTF-8, it is not compatible with ASCII, meaning it cannot translate ASCII code.

Example: The letter A in UTF-16 is equal to 00000000 01000001 in binary or 65 in decimal(2 bytes).

For Urdu:

Example: The right Urdu letter ب in UTF-16 is represented as is 0000011000101000 in binary, which occupies 2 bytes of memory

42. Define UTF-32 with example.

Ans: UTF-32: UTF-32 is a method of encoding that uses a fixed length, with all characters stored in 4 bytes per character. This makes it very simple but at the same time it may look a little complicated when it comes to space usage.

Example: Alphabet letter 'A' in UTF-32 is represented in binary as 00000000 00000000 00000000 01000001 which is 4 bytes.

2.6 STORING IMAGES, AUDIO AND VIDEO IN COMPUTERS

43. What is the primary difference between JPEG and PNG image file formats?

Ans: JPEG (Joint Photographic Expert Group): It compresses the image to save space but might lose some quality. while,

PNG (Portable Network Graphics): Supports transparency and maintains high quality without losing data.

44. What are the three commonly used image file formats?

Ans: Image File Formats: The following are commonly used image formats for photos-**JPEG (Joint Photographic Expert Group).** It compresses the image to save space but might lose some quality. -**PNG (Portable Network Graphics):** Supports transparency and maintains high quality without losing data. -**GIF(Graphics Interchange Format):** Used for simple animations and images with few colors.

45. What is the difference between sampling and quantization in audio storage?

Ans: Sampling and Quantization:

Sampling: Recording the sound wave at regular intervals. The number of samples per second is called the sampling rate. Higher sampling rates result in better quality.

Quantization: Converting each sample into a number. More bits per sample provide more accurate sound representation.

46. How are the video files are stored in computer?

Ans: Storing Video: Videos are made up of many images shown rapidly in sequence, along with audio.Each image in a video is called a frame.

Frames and Frame Rate:

Frame Rate: The number of frames shown per second, measured in Frames Per Second (FPS). Common frame rates are 24 fps (used in movies) and 30 fps(used in TV). Higher frame rates result in smoother motion in videos.

Video File Formats:

MP4: A widely used format that efficiently compresses video while maintaining quality.

AVI: An older format that may result in larger file sizes.

MKV: Supports high-quality video and multiple audio tracks or subtitles.

Conceptual Short Questions

1. Why is the octal number system not used in modern computers for calculations?

Ans: The octal number system is not actually used in modern computers to do their work. Therefore, we can say that the binary number 110101011 is equal to 653 in octal. Whenever you have a binary number that cannot be divided into groups of a three, you'll have to add zero up to the left end of it to make it appropriate.

2. Which numbering system do we use in everyday life?

Ans: The decimal number system is a base-10 number system that consists of digit from 0 to 9 and we use it in everyday life. That is why each digit of the number represents a power of 10

3. Explain why binary numbering systems are suitable for computers.

Ans: Computers work in binary system especially because this method fits well with electronics. Digital circuits have two states: They can be either on or off. These states are easily represented by the binary digits: 1 represent ON, and 0represnets OFF. When typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

When you type a letter on your keyboard, the computer converts it into a binary code. Similarly, all types of data, including numbers, text, images, and sounds, are ultimately broken down into binary code.

4. How does the keyboard interact with the computer's binary system?

Ans: When typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

When you type a letter on your keyboard, the computer converts it into a binary code. Similarly, all types of data, including numbers, text, images, and sounds, are ultimately broken down into binary code.

5. What is the essence of representing data like letters, numbers, and images in binary?

Ans: Computers work in binary system especially because this method fits well with electronics. Digital circuits have two states: They can be either on or off. These states are easily represented by the binary digits: 1 represent ON, and 0 represents OFF. When typing on the keyboard, the computer translates every letter to a binary. Similarly, number, text, images, and sound are all, at their lowest level, reduced to binary.

When you type a letter on your keyboard, the computer converts it into a binary code. Similarly, all types of data, including numbers, text, images, and sounds, are ultimately broken down into binary code.

6. What is the significance of understanding the binary encoding of integers in computer science?

Ans: Computers can process and store a lot of information .When we store data in computers, especially numbers, it is important to understand how they are represented and stored in memory.

7. How does understanding the representation of integers in memory help in programming?

Ans: Integers(Z):

Integers extend the concept of whole numbers to include negative numbers. In computer programming, we call them signed integers. The set of integers is represented as:

$Z=\{...-3,-2,-1,0,1,2,3,...\}$

To store both positive and negative values, one bit is reserved as the sign bit (the

most significant bit). If the sign bit is ON(1), the value is negative; otherwise, it is positive. Using this system, the maximum positive value that can be stored in a 1byte signed integer is (01111111)2, which is $(127)_{10}$,.As the bits available to stored a value is n -1, hence the maximum value will be $2^{n-1}-1$.We can use this formula to compute the maximum values for 2 and 4 bytes.

8. Explain why expressing real numbers in a binary format is crucial for computer systems.

Ans: Storing Real Values in Computer Memory:

In computers, real values, also known as floating-point numbers, are used to represent number with fractions and/or decimals. The information about how real values is stored in computer memory help us understand the precision and limitations of digital computation. With this understanding of floating-point representation, it becomes possible to control and manipulate these numbers in different ways.

9. How does the binary representation of a real number differ in "Single Precision (32-bit)" compared to standard binary representation?

Ans: Single Precision(32-bit): In this standard, 4 bytes (or 32 bits) are assigned where the 1st bit is the sign bit, and the next 8 bits are for the exponent and the remaining 23 bits are for the mantissa.

Here the exponent can be ranged between-126 and+127.

The approximate range of values from 1.4x1049 to 3.4x1038.

10. What is ASCII's main purpose in computers?

Ans: ASCII: ASCII is an acronym that stands for American Standard Code for Information Interchange. It is a character encoding standard adopted for representing in devices such as computers and similar systems that use text. Each alphabet, number or symbol is given a code number between 0 and 127. ASCII enables different computers and devices to exchange text information reliably.

The ASCII code is a numerical representation of characters in computer-based system, particularly for alphabetic characters.

For Example: the ASCII code of the character 'n' is 110.

Exercise Questions

A. Multiple Choice Questions.

1. What does ASCII stand for?

(a) American Standard Code for Information Interchange

(b) Advanced Standard Code for Information Interchange

(c) American Standard Communication for Information Interchange

(d) Advanced Standard Communication for Information Interchange

2. Which of the following numbers is a valid binary number?

(a) 1101102 (b) 11011 (c) 110.11 (d) 1101A

3. **How many bits are used in the standard ASCII encoding?**
(a) 7 bits (b) 8 bits (c) 16 bits (d) 32 bits

4. **Which of the following is a key advantage of Unicode over ASCII?**
(a) It uses fewer bits per character.
(b) It can represent characters from many different languages.
(c) It is backward compatible with binary.
(d) It is specific to the English language.

5. **How many bytes are used to store a typical integer?**
(a) 1 byte (b) 2 bytes (c) 4 bytes (d) 8 bytes

6. **What is the primary difference between signed and unsigned integers?**
(a) Unsigned integers cannot be negative
(b) Signed integers have a larger range
(c) Unsigned integers are stored in floating-point format
(d) Signed integers are only used for positive numbers

7. **In the single precision, how many bits are used for the exponent?**
(a) 23 bits (b) 8 bits (c) 11 bits (d) 52 bits

8. **What is the approximate range of values for single-precision floating-point numbers?**
(a) 1.4×10^{-45} to 3.4×10^{38} (b) 1.4×10^{-38} to 3.4×10^{45}
(c) 4.9×10^{-324} to 1.8×10^{308} (d) $4..9 \times 10^{-308}$ to 1.8×10^{308}

9. **What are the tiny dots that make up an image called?**
(a) Pixels (b) Bits (c) Bytes (d) Nodes

10. **In an RGB color model, what does RGB stand for?**
(a) Red, Green, Blue (b) Red, Gray, Black
(c) White Green, Blue (d) Red, Green, Brown

ANSWERS:

1. (a)	2. (b)	3. (a)	4. (b)	5. (c)
6. (a)	7. (b)	8. (a)	9. (a)	10. (a)

B. Write short answers.

1. **What is the primary purpose of the ASCII encoding scheme?**

Ans: ASCII: ASCII is an acronym that stands for American Standard Code for Information Interchange. It is a character encoding standard adopted for representing in devices such as computers and similar systems that use text. Each alphabet, number or symbol is given a code number between 0 and 127.ASCII enables different computers and devices to exchange text information reliably.

2. **Explain the difference between ASCII and Unicode.**

Ans: ASCII: ASCII is an acronym that stands for American Standard Code for Information Interchange. It is a character encoding standard adopted for representing in devices such as computers and similar systems that use text. Each

alphabet, number or symbol is given a code number between 0 and 127.ASCII enables different computers and devices to exchange text information reliably.

Unicode: Unicode is an attempt at mapping all graphic characters used in any of the world's writing system. Unlike ASCII, which is limited to 7bits and can represent only 128 characters, Unicode can represent over a million characters through different forms of encodings such as, UTF-8, UTF-16, and UTF-32. UTF is an acronym that stands for Unicode Transformation Format.

3. How does Unicode handle characters from different languages?

Ans: Unicode: Unicode is an attempt at mapping all graphic characters used in any of the world's writing system. Unlike ASCII, which is limited to 7bits and can represent only 128 characters, Unicode can represent over a million characters through different forms of encodings such as, UTF-8, UTF-16, and UTF-32. UTF is an acronym that stands for Unicode Transformation Format.

UTF-8: It is a variable-length encoding scheme, meaning it can use a different numbers of bytes (from 1 to 4) to represent a character. UTF-8 is backward compatible with ASCII. It means it can understand and use the older ASCII encoding scheme without any problems .Therefore, if we have a text file written in ASCII, it will work perfectly fine with UTF-8,allowing it to read both old and new texts.

Example: The letter 'A' is Unicode, represented as, U+0041,is 01000001 in the binary format and occupies 8 bits or 1 byte.

Let us look at how Urdu letters are represented in UTF-8:

Example: The Urdu letter 'ب' is represented in Unicode as U+0628; its binary format is 11011000 10101000,means it takes 2 bytes.

4. What is the range of values for an unsigned 2-byte integer?

Ans: 2-Byte Integer(16 bits): Minimum value= $-2^{15} = -32,768$

5. Explain how a negative integer is represented in binary.

Ans: Negative Values and Two's Complement:

To store negative values, computers use a method called two's complement. To find the two's complement of a binary number, follow these steps:

1. Invert all the bits (change Os to 1s and 1s to Os).
2. Add 1 to the Least Significant Bit (LSB).

Example:

Let us convert the decimal number – 5 to an 8-bit binary number:

1. Start with the binary representation of 5:00000101_2.
2. Invert all the bits:11111010_2.
3. Add1: $11111010_2 + 1_2 = 11111011_2$.

So,-5 in 8-bit two's complement is 11111011_2.

6. What is the benefit of using unsigned integers?

Ans: Integers(Z): Integers extend the concept of whole numbers to include negative numbers. In computer programming, we call them signed integers. The set of

integers is represented as:

$Z=\{...,-3,-2,-1,0,1,2,3,...\}$

To store both positive and negative values, one bit is reserved as the sign bit (the most significant bit). If the sign bit is ON(1), the value is negative; otherwise, it is positive. Using this system, the maximum positive value that can be stored in a 1byte signed integer is $(01111111)_2$, which is 127_{10}. As the bits available to stored a value is n -1, hence the maximum value will be $2^{n-1}-1$.We can use this formula to compute the maximum values for 2 and 4 bytes.

7. How does the number of bits affect the range of integer values?

Ans: Integers(Z): Integers extend the concept of whole numbers to include negative numbers. In computer programming, we call them signed integers. The set of integers is represented as:

$Z=\{...-3,-2,-1,0,1,2,3,...\}$

To store both positive and negative values, one bit is reserved as the sign bit (the most significant bit). If the sign bit is ON(1), the value is negative; otherwise, it is positive. Using this system, the maximum positive value that can be stored in a 1byte signed integer is $(01111111)_2$, which is 127_{10},.As the bits available to stored a value is n -1, hence the maximum value will be $2^{n-1}-1$.We can use this formula to compute the maximum values for 2 and 4 bytes.

8. Why are whole numbers commonly used in computing for quantities that cannot be negative? What are whole numbers?

Ans. Whole Numbers(W): Whole numbers are a set of non-negative integers. They include zero and all the positive integers. Mathematically, the set of whole numbers is:

$W=\{0,1,2,3,...\}$

In computing, whole numbers are often used to represent quantities that can't be negative.

Examples: The number of students in a school, a person's age in years, and grades, provided there are no negative figures such as credit point balances.

9. How is the range of floating-point numbers calculated for single precision?

Ans: Single Precision(32-bit): In this standard, 4 bytes (or 32 bits) are assigned where the 1st bit is the sign bit, and the next 8 bits are for the exponent and the remaining 23 bits are for the mantissa.

Here the exponent can be ranged between-126 and+127.

The approximate range of values from 1.4×10^{-45} to 3.4×10^{38}

Each floating point value is broken down into three main components: the sign bit, the exponent, and the mantissa.

10. Why is it important to understand the limitations of floating-point representation in scientific computing?

Ans: Floating-Point Number: A floating-point number is a numerical representation that uses a combination of a mantissa, exponent, and base to represent real numbers. It is a binary format that consists of three main parts: mantissa, exponent, and sign bit.

C. Long Questions

1. Explain how characters are encoded using Unicode. Provide examples of characters from different languages and their corresponding Unicode code points.

Ans. For Answer See Q.8

2. Describe in detail how integers are stored in computer memory.

Ans. For Answer See Q.6

3. Explain the process of converting a decimal integer to its binary representation and vice versa. Include examples of both positive and negative integers.

Ans. For Answer See Q.2

4. Perform the following binary arithematic operations.

(a) Multiplications of 101 by 11

Ans:

$$\begin{array}{r} 101 \\ \times\ 11 \\ \hline 101 \\ 101\times \\ \hline 1111 \end{array} = (1111)_2$$

(b) Division of 1100 by 10

Ans.

$$\begin{array}{r|l} & 110 \\ \hline 10 & 1100 \\ & -10 \\ \hline & 10 \\ & 10 \\ \hline \end{array} = (110)_2$$

5. Convert the following numbers to 4-binary and add them.

(a)

$$\begin{array}{r} 1\ 0\ 1 \\ +\ 1\ 1\ 0 \\ \hline 1\ 0\ 1\ 1 \end{array} = (1011)_2$$

(b)

$$\begin{array}{r} 1\ 1\ 0\ 0 \\ +\ 1\ 0\ 1\ 1 \\ \hline 1\ 0\ 1\ 1\ 1 \end{array} = (1011)_2$$

6. Convert the following numbers to 4-binary and ad them:

(a) 7 + (–4)

Ans: 7 in 4– bit binary: 0111

–4 in 4 bit binary (Two's complement): 1100

$$\begin{array}{r} 0\ 1\ 1\ 1 \\ +\ 1\ 1\ 0\ 0 \\ \hline 1\ 0\ 0\ 1\ 1 \end{array} = (10011)_2$$

(b) –5 + 3

Ans. –5 in 4 - bit (Two's complement): 0011

$$\begin{array}{r} 1\ 0\ 1\ 1 \\ +\ 0\ 0\ 1\ 1 \\ \hline 1\ 1\ 1\ 0 \end{array} = (1110)_2$$

7. Solve the following:

(a) $(1101)_2 - (10100)_2$

Ans.

$$\begin{array}{r} 1\ 1\ 0\ 1 \\ -\ 0\ 1\ 0\ 0 \\ \hline 1\ 0\ 0\ 1 \end{array}$$

(b) $(1010)_2 - (0011)_2$

Ans.

$$\begin{array}{r} 1\ 0\ 1\ 0 \\ -\ 0\ 0\ 1\ 1 \\ \hline 0\ 1\ 1\ 1 \end{array}$$

(c) $(1000)_2 - (0110)_2$

Ans.

$$\begin{array}{r} 1\ 0\ 0\ 0 \\ -\ 0\ 1\ 1\ 0 \\ \hline 0\ 0\ 1\ 0 \end{array}$$

(d) $(1110)_2 - (100)_2$

Ans.

$$\begin{array}{r} 1\ 1\ 1\ 0 \\ -\ 1\ 0\ 0 \\ \hline 1\ 0\ 1\ 0 \end{array}$$