Exercise

✓ Multiple Choice Questions – Correct Answers

1. **What does ASCII stand for?**
   → (a) American Standard Code for Information Interchange
2. **Which of the following numbers is a valid binary number?**
   → (b) 11011
   (Explanation: A binary number can only contain digits 0 and 1. Option (a) contains "2", and (d) contains "A", so both are invalid.)
3. **How many bits are used in the standard ASCII encoding?**
   → (a) 7 bits
4. **Which of the following is a key advantage of Unicode over ASCII?**
   → (b) It can represent characters from many different languages
5. **How many bytes are used to store a typical integer?**
   → (c) 4 bytes
6. **What is the primary difference between signed and unsigned integers?**
   → (a) Unsigned integers cannot be negative
7. **In single precision, how many bits are used for the exponent?**
   → (b) 8 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}
9. **What are the tiny dots that make up an image called?**
   → (a) Pixels
10. **In an RGB color model, what does RGB stand for?**
    → (a) Red, Green, Blue

✍ Short Questions – Simple Answers

1. **What is the primary purpose of the ASCII encoding scheme?**
   → ASCII gives a unique numeric code to each character (letter, digit, symbol) so computers can store and process them as numbers.
2. **Explain the difference between ASCII and Unicode.**
   → ASCII uses 7 bits and supports 128 characters (mostly English). Unicode supports many languages and uses more bits (like 16 or 32).
3. **How does Unicode handle characters from different languages?**
   → Unicode assigns a unique code to every character, including letters from Arabic, Chinese, Urdu, and others.

4. **What is the range of values for an unsigned 2-byte integer?**
   → 0 to 65535
   (Explanation: 2 bytes = 16 bits, so range = 0 to 2^{16} - 1 = 65535)
5. **Explain how a negative integer is represented in binary.**
   → It is represented using two's complement form, where the most significant bit (MSB) is 1.
6. **What is the benefit of using unsigned integers?**
   → They allow a larger range of positive values since no bits are reserved for sign.
7. **How does the number of bits affect the range of integer values?**
   → More bits mean a larger range. For example, 8 bits can represent 0–255, but 16 bits can represent 0–65535.
8. **Why are whole numbers commonly used in computing for quantities that cannot be negative?**
   → Because negative values are not meaningful for things like age, quantity, and file size, so unsigned integers are used.
9. **How is the range of floating-point numbers calculated for single precision?**
   → By using 1 sign bit, 8 exponent bits, and 23 mantissa bits according to IEEE 754 standard.
10. **Why is it important to understand the limitations of floating-point representation in scientific computing?**
    → Because floating-point values can be imprecise and may cause rounding errors in calculations.

📕 Long Questions – Detailed Answers

1. Explain how characters are encoded using Unicode. Provide examples.
→ Unicode gives every character a unique code point, even from different languages.
Example:

- English letter 'A' = U+0041
- Arabic letter 'ب' = U+0628
- Chinese character '你' = U+4F60
  Unicode can use different encoding formats like UTF-8, UTF-16, and UTF-32. UTF-16 uses 2 or 4 bytes to store each character.

---

2. Describe in detail how integers are stored in computer memory.
→ Integers are stored in binary using a fixed number of bits like 8, 16, or 32.

- **Unsigned integers**: All bits represent the number (e.g., 8-bit can store 0–255)
- **Signed integers**: Use one bit for sign (positive or negative), and remaining bits for the value.
  Negative numbers are stored using **two's complement**.

3. Explain the process of converting a decimal integer to binary and vice versa.

Decimal to Binary Example (Positive):
→ Convert 13 to binary:
13 ÷ 2 = 6 R1
6 ÷ 2 = 3 R0
3 ÷ 2 = 1 R1
1 ÷ 2 = 0 R1 → Binary = 1101

Binary to Decimal Example:
→ Convert 1101 to decimal:
1×8 + 1×4 + 0×2 + 1×1 = 13

Decimal to Binary (Negative using 2's complement):
→ -5 in 4-bit binary:
+5 = 0101
Step 1: Invert = 1010
Step 2: Add 1 = 1011 → So -5 = 1011

---

4. Perform the following binary arithmetic operations:

a. 101 × 11
→ 101
× 011
————
→ 101 (×1)
+1010 (×10)
= 1111

b. 1100 ÷ 10
→ 1100 = 12
→ 10 = 2
→ 12 ÷ 2 = 6 → Binary of 6 = 110

---

6. Add the following binary numbers:

a. 101 + 110
→ 101 = 5, 110 = 6 → 5 + 6 = 11 → Binary = 1011

b. 1100 + 1011
→ 1100 = 12, 1011 = 11 → 12 + 11 = 23 → Binary = 10111

7. Convert the following numbers to 4-bit binary and add them:

a. 7 + (-4)
→ 7 = 0111
→ -4 = 1100 (in 2's complement)
→ 0111 + 1100 = 0011 (Overflow ignored) = 3

b. -5 + 3
→ -5 = 1011
→ 3 = 0011
→ 1011 + 0011 = 1110 (in 2's complement, this is -2)

8. Solve the following binary subtractions:

a. $\mathbf{1101_2 - 0100_2}$ → 1101 - 0100 = 1001
b. $\mathbf{1010_2 - 0011_2}$ → 1010 - 0011 = 0111
c. $\mathbf{1000_2 - 0110_2}$ → 1000 - 0110 = 0010
d. $\mathbf{1110_2 - 0100_2}$ → 1110 - 0100 = 1010