C Programming

Topics:

1. Input & Output
2. Variables & Data Types
3. Comments

Data types

Data Type	Typical Size	Description	Value Range
`char`	1 byte	Stores a single character. Can also store small integers (-128 to 127).	-128 to 127
`unsigned char`	1 byte	Only positive values (0 to 255).	0 to 255
`int`	4 bytes	Stores integer values. Usually 32-bit signed.	-2,147,483,648 to 2,147,483,647
`unsigned int`	4 bytes	Only positive integers (0 to 4,294,967,295).	0 to 4,294,967,295
`short`	2 bytes	Smaller integer (-32,768 to 32,767).	-32, 768 to 32, 767
`unsigned short`	2 bytes	Only positive short integers (0 to 65,535).	0 to 65,535
`long`	4 or 8 bytes	Larger integer, depending on system. 32-bit or 64-bit.	4 byte: (-2,147,483,648 to 2,147,483,647) or 8 byte: (-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807)
`unsigned long`	4 or 8 bytes	Positive long integers only.	4 byte: (0 to double of long) B byte: (0 to double of long)
`long long`	8 bytes	Very large integer.	9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`unsigned long long`	8 bytes	Positive very large integer.	0 to 18,446,744,073,709,551,615
`float`	4 bytes	Single-precision floating point number (approx 6-7 digits of precision).	Range: (1.2×10^{-38} to 3.4×10^{38}) and Precision: 6-7 digits
`double`	8 bytes	Double-precision floating point number (approx 15-16 digits of precision).	Range: (2.3×10^{-308} to 1.7×10^{308}) and Precision: 15-16 digits
`long double`	12–16 bytes	Extended precision floating point (depends on compiler).	Range: (3.4×10^{-4999} to 1.1×10^{4999}) and Precision: 18-19 digits

Format Specifiers:

Format Specifier	Data Type	Description	Example
%d	int	Signed decimal integer	int a = 10; printf("%d", a); → 10
%i	int	Signed decimal integer (same as %d)	int a = 10; printf("%i", a); → 10
%u	unsigned int	Unsigned decimal integer	unsigned int b = 20; printf("%u", b); → 20
%o	unsigned int	Octal (base 8)	printf("%o", 10); → 12
%x	unsigned int	Hexadecimal (lowercase)	printf("%x", 15); → f
%X	unsigned int	Hexadecimal (uppercase)	printf("%X", 15); → F
%f	float	Floating-point number (decimal)	float f = 3.14; printf("%f", f); → 3.140000
%lf	double	Double-precision floating-point	double d = 3.14159; printf("%lf", d); → 3.141590
%Lf	long double	Extended precision floating-point	long double ld = 3.1415926; printf("%Lf", ld); → 3.141593
%c	char	Single character	char ch = 'A'; printf("%c", ch); → A
%s	char[] / string	String of characters	char str[]="Hi"; printf("%s", str); → Hi
%p	Pointer	Memory address	int *ptr = &a; printf("%p", ptr); → 0x7ffee3...
%e / %E	float / double	Scientific notation	printf("%e", 1234.5); → 1.234500e+03
%g / %G	float / double	Automatically selects %f or %e	printf("%g", 1234.5); → 1234.5
%%	—	Prints the % symbol	printf("%%"); → %

Problem List on basic

1. Declare the variables: name (character), age (integer), weight(float), salary(double), assign values to each variable and print them using the right specifiers (%c or %s, %d, %f, %lf).
2. Write a C program to declare the following variables: name (character), age (integer), weight(float), salary(double). Take input for each variable and print them using the right specifiers (%c or %s, %d, %f, %lf). [***Hints: Take input using scanf function***]
3. Input a name, age, roll and grade of a student, then print them in a single line.
4. Input a name, age, roll and grade of a student, then print them in multiple lines.
5. Write a C program to store the following employee information in separate variables:

    - Employee ID (integer)
    - Employee Age (integer)
    - Employee Salary (float)
    - Employee Grade (character)
    - Employee Department Code (integer)

Then, print all the values using the correct format specifiers.

6. Input a float value from the user and display it with 2 decimal places.
7. Print a sentence with tabs (\t) between values.
8. Print a sentence with new lines (\n) between values.
9. Print a sentence mixing integers, floats, and characters in formatted style.
10. Input two integers and calculate their sum, difference, product, and quotient.
11. Input two integers and calculate the remainder using modulus operator %.
12. Input three integers and calculate their average.
13. Input length and width of a rectangle and find its area and perimeter.
14. Input radius of a circle and calculate area and circumference using 3.1416.
15. Input base and height of a triangle and calculate its area.
16. Input two numbers and swap them using a temporary variable.
17. Input two numbers and swap them without using a temporary variable.
18. Convert temperature from Celsius to Fahrenheit. [$F=(9/5\times C)+32$]
19. Convert temperature from Fahrenheit to Celsius. [$C=5/9\times(F-32)$]
20. Input marks of 5 subjects, calculate total and average, and display percentage.
21. Input a number and display its square and cube.

22. Write a program that takes input for Principal amount, Rate of interest, and Time (in years) from the user, and then calculates and prints the Simple Interest. [***SI=(P×R×T)/100***]
23. Write a C program that takes input of **two numbers** from the user.
    - Update each number by multiplying it by 5.
    - Then, input another number and subtract it from both updated numbers.
    - Finally, print the resulting values.