C STRINGS IN C PROGRAMMING (WITHOUT POINTERS)

1. DECLARING AND INITIALIZING STRINGS

- Strings in C are arrays of characters terminated by a null character ('\0').

- They are declared as character arrays, with a fixed size.

2. READING AND WRITING STRINGS

- Use scanf() to read strings (input stops at the first whitespace).

- Use printf() to print strings.

3. STRING HANDLING FUNCTIONS

a. strlen()

- This function returns the length of the string, excluding the null terminator.

b. strcpy()

- Copies the contents of one string into another.

c. strncpy()

- Copies a specified number of characters from one string to another.

d. strcmp()

- Compares two strings lexicographically, character by character.

e. strncmp()

- Compares the first n characters of two strings.

f. strcat()

- Appends one string to the end of another string.

g. strncat()

- Appends the first n characters of one string to another string.

h. strchr()

- Finds the first occurrence of a character in a string.

i. strstr()

- Finds the first occurrence of a substring within another string.

j. strrev()

- Reverses the contents of a string (in some compilers).

4. STRING COMPARISON

- Use strcmp() to compare two strings. It returns:
  - 0 if the strings are equal.
  - A negative value if the first string is lexicographically smaller.
  - A positive value if the first string is lexicographically greater.

5. COMMON STRING OPERATIONS

- Copying: Use strcpy() or strncpy() to copy one string into another.
- Concatenation: Use strcat() or strncat() to append one string to another.
- Searching: Use strchr() or strstr() to find characters or substrings within a string.
- Comparison: Use strcmp() or strncmp() to compare two strings.

6. COMMON MISTAKES WITH STRINGS

- Buffer Overflow: Ensure that the destination string has enough space to accommodate the source string during operations like copying or concatenation.
- Missing Null Terminator: Always ensure that strings are properly terminated with the null character ('\0').

7. BEST PRACTICES

- Always ensure the size of the destination string is large enough to hold the source string and the null terminator.
- Use safe functions like fgets() instead of gets() to avoid buffer overflow.