

# Strings

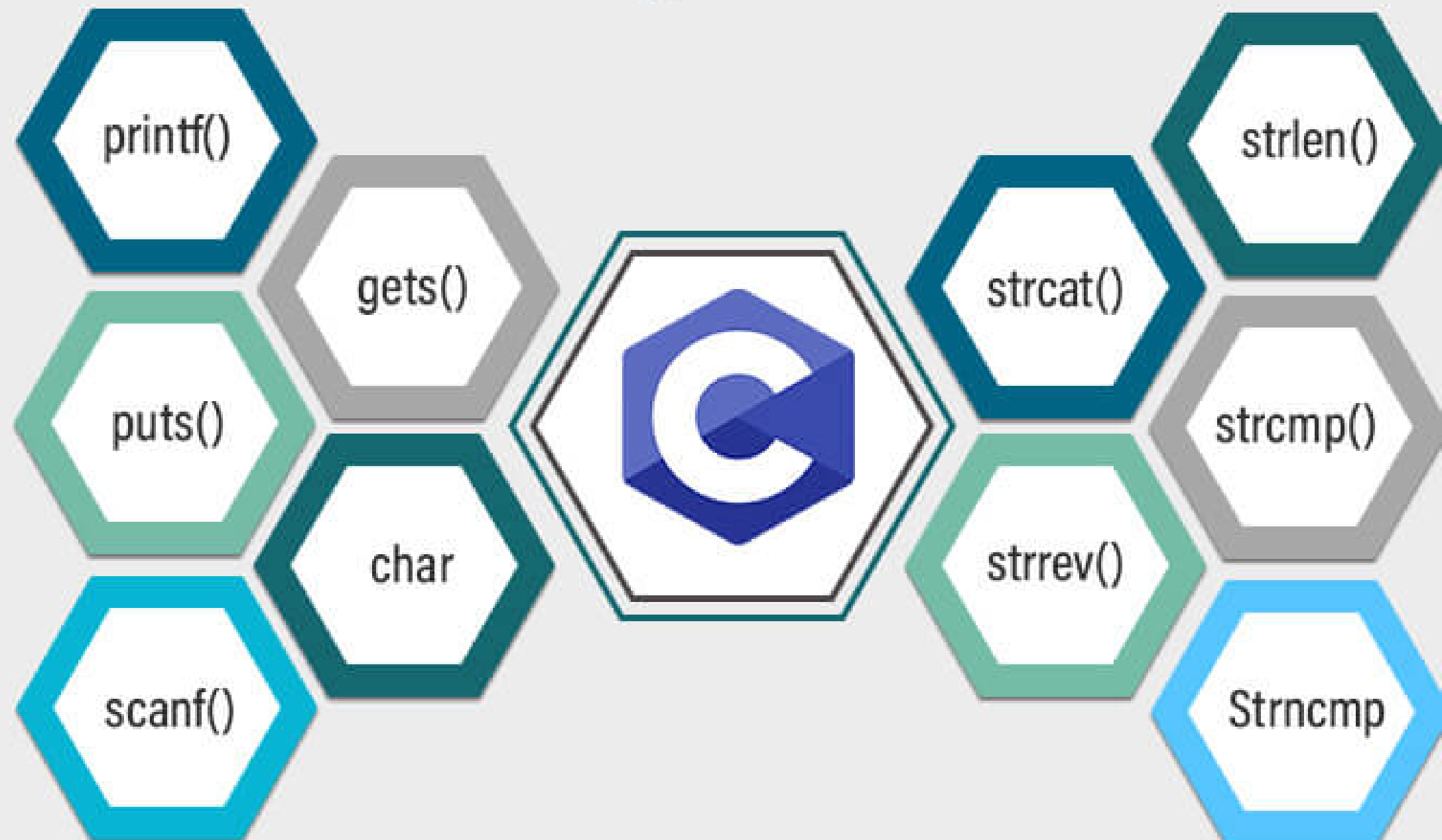


## Defnition

fundamental data type used to represent textual data. They are sequences of characters and are enclosed in double quotes (" ")



# C String Functions



## Creating Strings

Strings in C are sequences of characters represented as arrays of characters (char arrays). They are defined using double quotes (" "), and the last character is a null character '\0' to indicate the end of the string.

```
// Declare a character array for the string  
char greeting[] = "Hello, World!";
```

Variable


H	e	l	l	o	\0
---	---	---	---	---	----

## String Indexing

Strings are ordered sequences, and you can access individual characters using indexing. C uses zero-based indexing, where the first character has an index of 0.

Index	0	1	2	3	4	5
Variable	H	e	l	l	o	\0

# String Manipulation Techniques

- **Appending:** Concatenating one string to the end of another.
  - **Truncating:** Removing characters from the end of a string.
  - **Extracting Substrings:** Selecting a portion of a string as a separate substring
- 

# Appending

```
char str1[50] = "Hello, ";  
char str2[] = "World!";  
strcat(str1, str2);  
printf("Appended String: %s\n", str1);
```

“Hello” + “World” = “ HelloWorld ”

String 1   String 2   Result


# Truncating

```
char text[] = "Carrot";  
text[3] = '\0';  
printf("Truncated String: %s\n", text);
```

Carrot

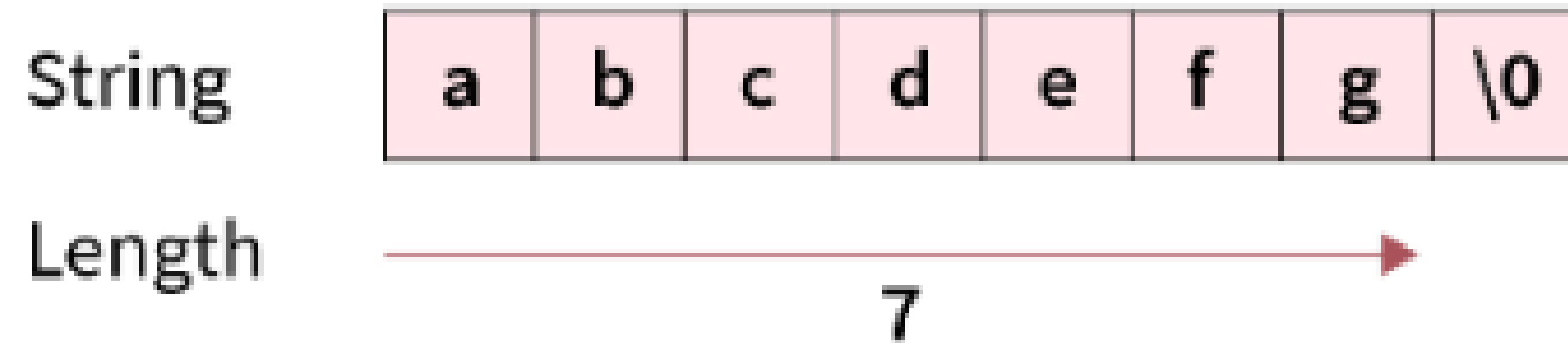
Car

# String Functions and Operations

- **strlen()**: Calculates the length of a string.
  - **strcpy()**: Copies one string into another.
  - **strcat()**: Concatenates one string onto the end of another.
  - **strcmp()**: Compares two strings for equality.
- 

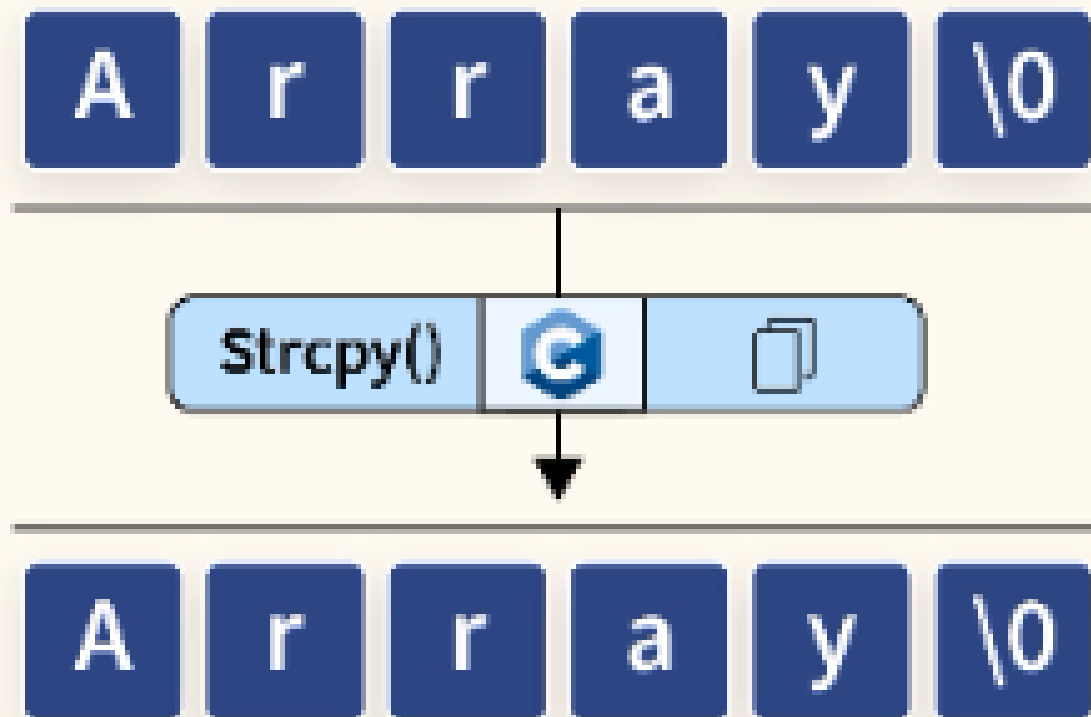
# strlen()

```
char text[] = "abcdefg!";  
int length = strlen(text);  
printf("Length of the string: %d\n", length);
```



# strcpy()

```
char source[] = "Array";  
char destination[20];  
strcpy(destination, source);  
printf("Copied String: %s\n", destination);
```



# strcat()

```
char str1[20] = "Hello, ";  
char str2[] = "World!";  
strcat(str1, str2);  
printf("Concatenated String: %s\n", str1);
```

“Hello” + “World” = “Hello World”


String 1   String 2   Result

# strcmp()

```
char str1[] = "string";  
char str2[] = "strong";  
int result = strcmp(str1, str2);
```

s	t	r	i	n	g
s	t	r	o	n	g

# Handling Escape Sequences in Strings

- `\n`: Newline
  - `\t`: Tab
  - `\\`: Backslash
  - `\"`: Double quote
- 

# Handling Escape Sequences in Strings

```
printf("This is a string with escape sequences:\n");  
printf("Line 1\tTabbed\n");  
printf("Line 2\\Backslash\n");  
printf("Line 3\"Double-quoted\"");
```

```
This is a string with escape sequences:  
Line 1    Tabbed  
Line 2\\Backslash  
Line 3"Double-quoted"
```

# String Formatting

- Using format specifiers with `printf()` and `sprintf()` to format strings.
- Examples include `%s` for strings, `%d` for integers, and `%f` for floating-point numbers.

# Pattern Matching and Searching in Strings

- Utilizing functions like `strstr()` to search for a substring within a string.
- **Regular expressions** for advanced pattern matching

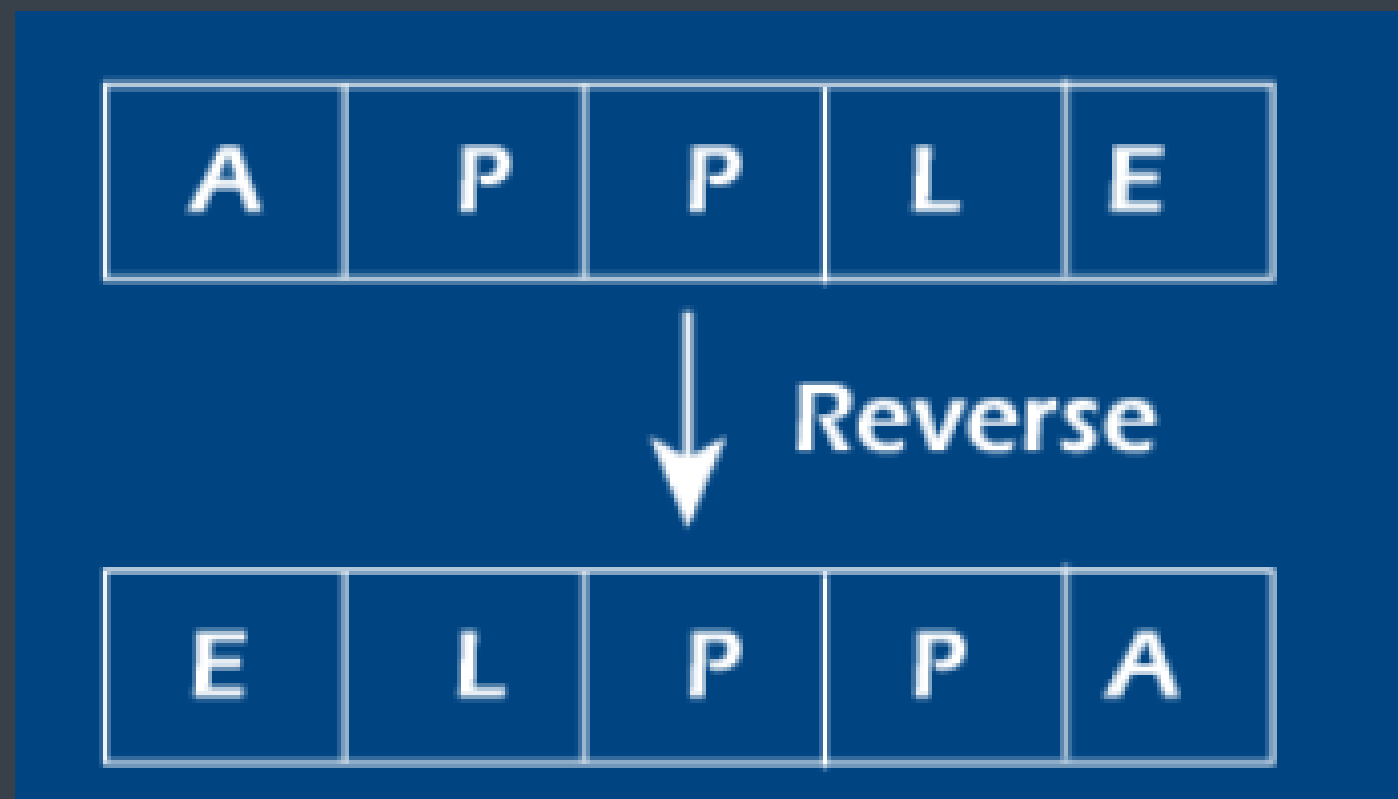
# Checking for Palindromes

- A palindrome is a string that reads the same forwards and backward (e.g., "racecar").
- To check for palindromes, you can compare characters from both ends of the string.



# Reversing Strings

- Reversing a string involves swapping the characters from the start with those from the end.
- This can be done using loops or recursion.



# Decimal to Binary and Binary to Decimal Conversion

- Converting decimal numbers to binary involves repeatedly dividing by 2 and recording remainders.
  - Converting binary numbers to decimal involves summing powers of 2 based on the binary digits.
- 