Char in depth

Size:

- Typically **1 byte (8 bits)**.
- Can store **-128 to 127** (if `signed char`) or **0 to 255** (if `unsigned char`).

Usage:

- To store characters (like `'a'`, `'A'`, `'$'`).
- Internally, characters are stored as **ASCII values**.
  Example: `'A'` → 65, `'a'` → 97.

Special forms:

- `signed char` → values from **-128 to 127**.
- `unsigned char` → values from **0 to 255**.
- `char` (default) → compiler-dependent (may be signed or unsigned).

1. ASCII Character Table (Common Set)

Character	Decimal Value	Character	Decimal Value
`A`	65	`a`	97
`B`	66	`b`	98
`C`	67	`c`	99
`D`	68	`d`	100
`E`	69	`e`	101
`F`	70	`f`	102
`G`	71	`g`	103
`H`	72	`h`	104

I	73	i	105
J	74	j	106
K	75	k	107
L	76	l	108
M	77	m	109
N	78	n	110
O	79	o	111
P	80	p	112
Q	81	q	113
R	82	r	114
S	83	s	115
T	84	t	116
U	85	u	117
V	86	v	118
W	87	w	119
X	88	x	120
Y	89	y	121
Z	90	z	122

2. Digits and Special Characters

Character	Decimal Value	Character	Decimal Value
0	48	:	58
1	49	;	59

2	50	<	60
3	51	=	61
4	52	>	62
5	53	?	63
6	54	@	64
7	55	[	91
8	56	\	92
9	57	]	93

3. Control Characters

Name	Char	Decimal Value
Null	`\0`	0
Backspace	`\b`	8
Tab	`\t`	9
Newline	`\n`	10
Carriage Return	`\r`	13
Space	`' '`	32

Valid vs Invalid

```
char c1 = 'A';     // ✅ valid
char c2 = '9';     // ✅ valid
char c3 = '#';     // ✅ valid
char c4 = '\n';    // ✅ valid (newline character)
char c5 = ' ';     // ✅ valid (space)

char c6 = 'AB';    // ❌ invalid (multi-character)
char c7 = "A";     // ❌ invalid (string, not char)
```

Scanf in C

What is `scanf`?

- `scanf` is a **standard input function** in C, defined in `<stdio.h>`.
- It is used to **read data** (from keyboard input, by default) and store it in variables.

Syntax:

```
scanf("format specifiers", &variables);
```

Important points

1. **`format specifiers`** tell `scanf` the type of data to read.
2. **`&` (address-of** operator**)** is required, because `scanf` needs the memory location of the variable to store the input.

Example Code:

```
#include <stdio.h>

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    printf("You entered: %d\n", num);
    return 0;
}
```

More Operators

Assignment Operator:-

refer to session2 notes

Relational Operator:-

refer to session2 notes

Logical Operators:-

refer to session2 notes

Bitwise Operator:-

refer to session2 notes

Introduction to Function

Functions

A function is a subprogram that can be defined by the user in their program. It is a complete program in itself, in the sense that its structure is similar to the **main()** function, except that the name **main** is replaced by the name of the function.

Syntax:

<type> <name>(arguments)

<type>: It is the type of value to be returned by the function. If no value is returned, then keyword **void** should be used.

<name>: It is a user defined name of the function. The function can be called from another function by this name.

Arguments: It is a list of parameters (parameter is the data that the function may receive when called from another function). The list can be omitted by leaving the parameters empty.

Example:

```
// Function to calculate area of a triangle
float triangleArea(float base, float height)
 { //Open braces
   float area = 0.5f * base * height;
   return area;
} //Closed braces
```

Note: The program segment enclosed within the opening brace and closing brace is known as the function body.

Calling a Function

Function can be called or invoked from another function by using its name. The function name must be followed by a set of actual parameters, enclosed in parentheses and separated by commas.

A function call to function **triangleArea** from the **main** program/function can be written as:

triangleArea(**float** base, **float** height);

Example:

```
#include<stdio.h>
//Function to calculate area of triangle
//Function definition
float triangleArea(float base, float height){
   float area = 0.5 * base * height;
   return area;
}
//Main function
int main(){
   float base, height,area;
   printf("Enter base of the triangle: ");
   scanf("%f", &base);
   printf("Enter height of the triangle: ");
   scanf("%f", &height);
   area = triangleArea(base, height); //Function call
   printf("Area of the triangle: %.2f\n", area);
   return 0;
}
```