# Introduction to Computing and Programming

Introduction to Programming, Identifiers and Constants

# Content

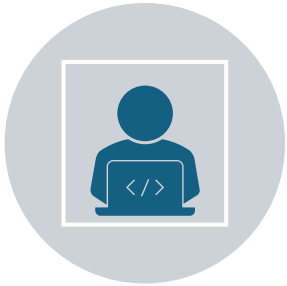
- Quick Recap
- Expressions
- Conditional Statements

# Recap

- Structure of C program
- Character Set
- Delimiters
- Keywords

# Introduction to C

C is a **general purpose and structured programming** language.

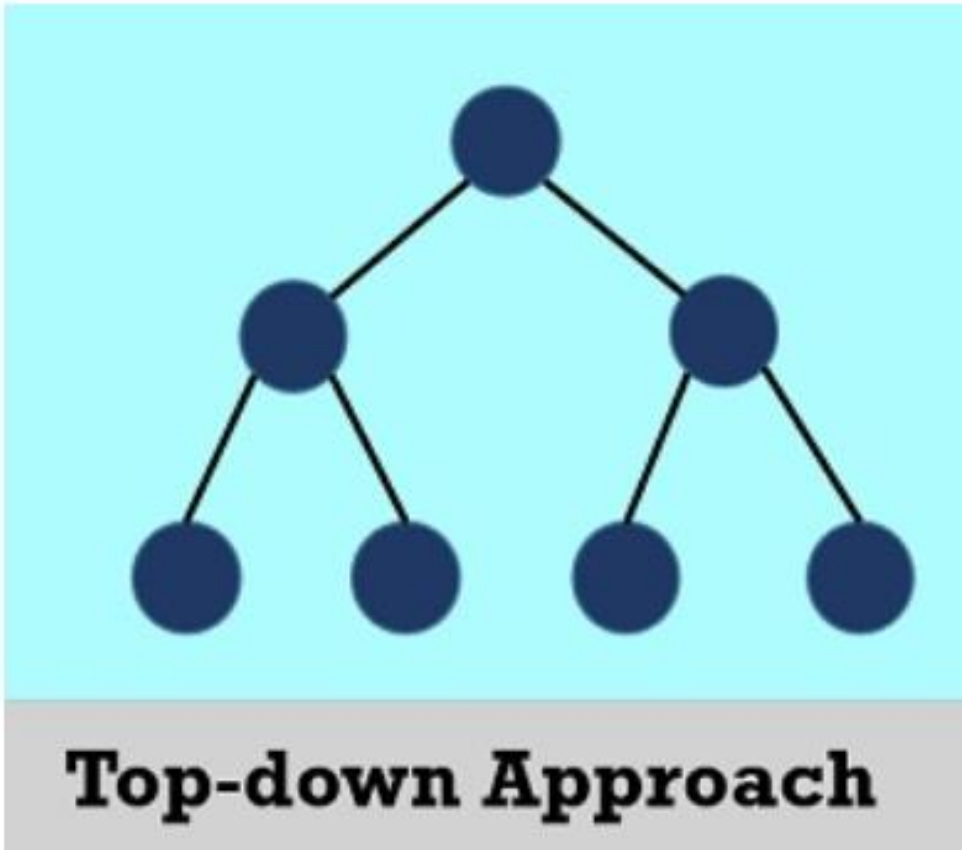


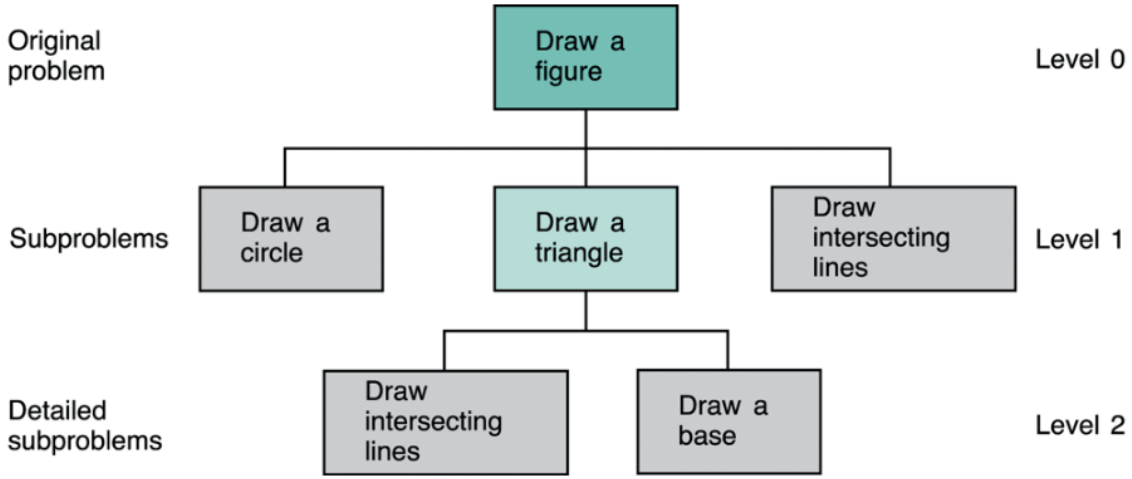
C can be used for **system programming and for application programming**.

It can be used to write very concise source program due to the availability of extensive libraries.

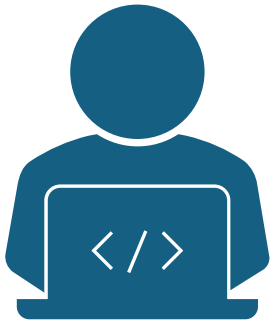
It is highly **portable**.

C Language follows **Top-down approach**: Divides the problem into subproblem then solves it



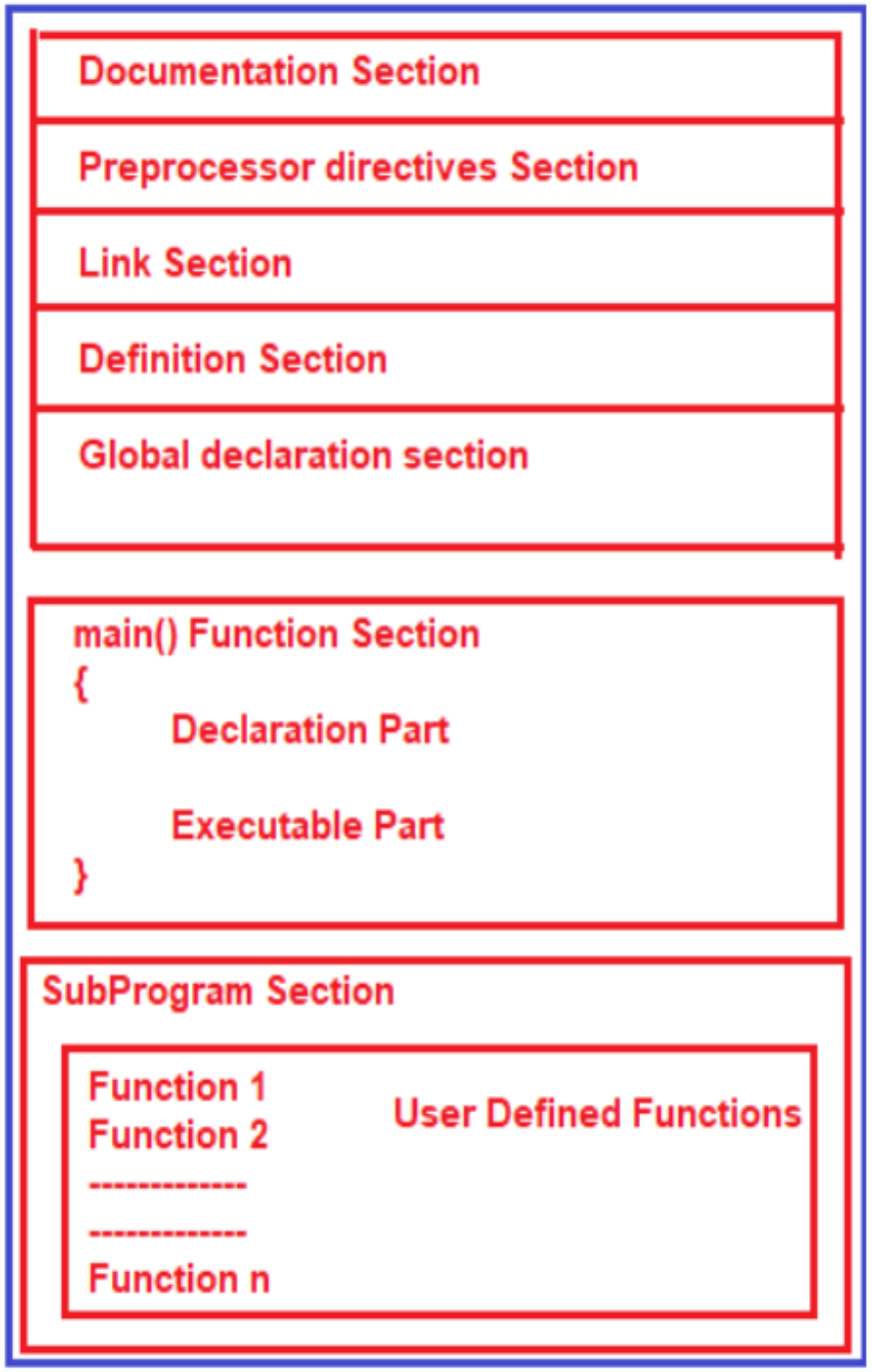



# Machine independent and Platform independent



A program that **run at any OS** means, that is called platform independent and those programs **run at any architecture** of computer (hardware) are called machine independent.

C is a Machine Independent language but it is not a platform independent language.




# Structure of C Program

# C Program Example

```c
#include <stdio.h>
int addition(int num1, int num2)
{
    int sum;
    sum = num1+num2;
    return sum;
}
```

```c
int main()
{
    int var1, var2;
    printf("Enter number 1: ");
    scanf("%d",&var1);
    printf("Enter number 2: ");
    scanf("%d",&var2);
    int res = addition(var1, var2);
    printf ("Output: %d", res);

    return 0;
}
```

# Elements of C

- Character Set
- Execution Characters/Escape Sequences
- Trigraph Characters
- Delimiters
- Reserved Words/ Keywords
- Identifiers
- Data types
- Constants
- Variables
- Expressions
- Statements

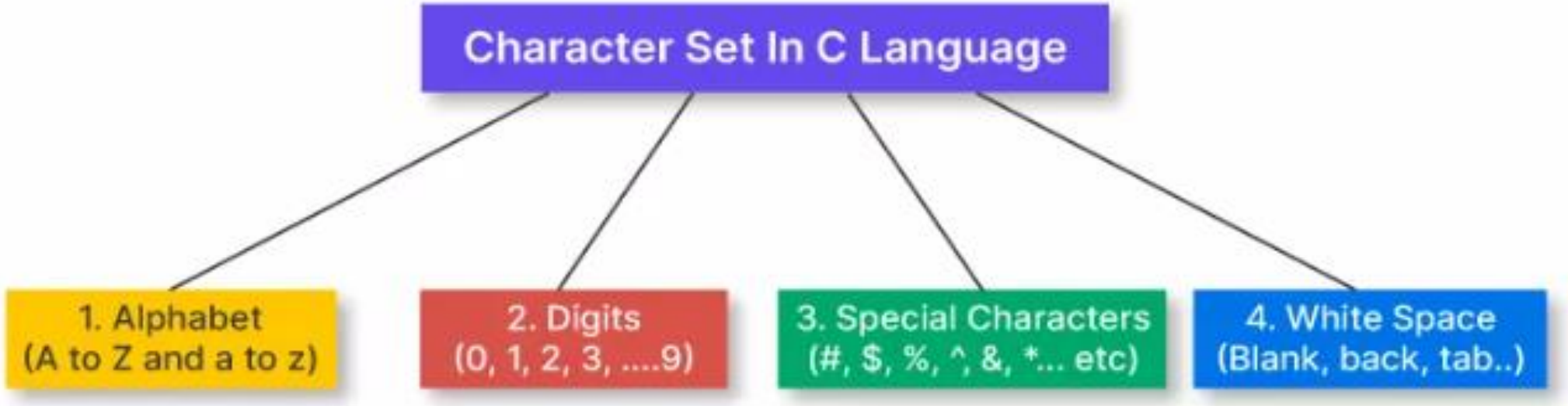
# ASCII: American Standard Code for Information Interchange (Letter representation)

| Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | [NULL] | 32 | 20 | [SPACE] | 64 | 40 | @ | 96 | 60 | ` |
| 1 | 1 | [START OF HEADING] | 33 | 21 | ! | 65 | 41 | A | 97 | 61 | a |
| 2 | 2 | [START OF TEXT] | 34 | 22 | " | 66 | 42 | B | 98 | 62 | b |
| 3 | 3 | [END OF TEXT] | 35 | 23 | # | 67 | 43 | C | 99 | 63 | c |
| 4 | 4 | [END OF TRANSMISSION] | 36 | 24 | $ | 68 | 44 | D | 100 | 64 | d |
| 5 | 5 | [ENQUIRY] | 37 | 25 | % | 69 | 45 | E | 101 | 65 | e |
| 6 | 6 | [ACKNOWLEDGE] | 38 | 26 | & | 70 | 46 | F | 102 | 66 | f |
| 7 | 7 | [BELL] | 39 | 27 | ' | 71 | 47 | G | 103 | 67 | g |
| 8 | 8 | [BACKSPACE] | 40 | 28 | ( | 72 | 48 | H | 104 | 68 | h |
| 9 | 9 | [HORIZONTAL TAB] | 41 | 29 | ) | 73 | 49 | I | 105 | 69 | i |
| 10 | A | [LINE FEED] | 42 | 2A | * | 74 | 4A | J | 106 | 6A | j |
| 11 | B | [VERTICAL TAB] | 43 | 2B | + | 75 | 4B | K | 107 | 6B | k |
| 12 | C | [FORM FEED] | 44 | 2C | , | 76 | 4C | L | 108 | 6C | l |
| 13 | D | [CARRIAGE RETURN] | 45 | 2D | - | 77 | 4D | M | 109 | 6D | m |
| 14 | E | [SHIFT OUT] | 46 | 2E | . | 78 | 4E | N | 110 | 6E | n |
| 15 | F | [SHIFT IN] | 47 | 2F | / | 79 | 4F | O | 111 | 6F | o |
| 16 | 10 | [DATA LINK ESCAPE] | 48 | 30 | 0 | 80 | 50 | P | 112 | 70 | p |
| 17 | 11 | [DEVICE CONTROL 1] | 49 | 31 | 1 | 81 | 51 | Q | 113 | 71 | q |
| 18 | 12 | [DEVICE CONTROL 2] | 50 | 32 | 2 | 82 | 52 | R | 114 | 72 | r |
| 19 | 13 | [DEVICE CONTROL 3] | 51 | 33 | 3 | 83 | 53 | S | 115 | 73 | s |
| 20 | 14 | [DEVICE CONTROL 4] | 52 | 34 | 4 | 84 | 54 | T | 116 | 74 | t |
| 21 | 15 | [NEGATIVE ACKNOWLEDGE] | 53 | 35 | 5 | 85 | 55 | U | 117 | 75 | u |
| 22 | 16 | [SYNCHRONOUS IDLE] | 54 | 36 | 6 | 86 | 56 | V | 118 | 76 | v |
| 23 | 17 | [END OF TRANS. BLOCK] | 55 | 37 | 7 | 87 | 57 | W | 119 | 77 | w |
| 24 | 18 | [CANCEL] | 56 | 38 | 8 | 88 | 58 | X | 120 | 78 | x |
| 25 | 19 | [END OF MEDIUM] | 57 | 39 | 9 | 89 | 59 | Y | 121 | 79 | y |
| 26 | 1A | [SUBSTITUTE] | 58 | 3A | : | 90 | 5A | Z | 122 | 7A | z |
| 27 | 1B | [ESCAPE] | 59 | 3B | ; | 91 | 5B | [ | 123 | 7B | { |
| 28 | 1C | [FILE SEPARATOR] | 60 | 3C | < | 92 | 5C | \ | 124 | 7C | \| |
| 29 | 1D | [GROUP SEPARATOR] | 61 | 3D | = | 93 | 5D | ] | 125 | 7D | } |
| 30 | 1E | [RECORD SEPARATOR] | 62 | 3E | > | 94 | 5E | ^ | 126 | 7E | ~ |
| 31 | 1F | [UNIT SEPARATOR] | 63 | 3F | ? | 95 | 5F | _ | 127 | 7F | [DEL] |

# Code for Character to Integer conversion

```
#include <stdio.h>
int main() {
        char ch; int ascii_value; // Prompt user to enter a character
        printf("Enter a character: ");
        scanf("%c", &ch); // Convert character to its ASCII integer value
        ascii_value = (int)ch; // Print the result
        printf("The ASCII value of '%c' is: %d\n", ch, ascii_value);
        return 0;
}
```

# Character Set




| Special Characters | | | | |
|---|---|---|---|---|
| + | > | / | [ | \ |
| ! | ; | “ | ] | { |
| < | * | . | % | } |
| : | ^ | , | ~ | # |
| - | ( | = | - | \| |
| ? | ) | ' | & | |

Execution Character / Escape Sequence

| Escape Sequences | Meaning |
|---|---|
| \' | Single Quote |
| \" | Double Quote |
| \\ | Backslash |
| \0 | Null |
| \a | Bell |
| \b | Backspace |
| \f | form Feed |
| \n | Newline |
| \r | Carriage Return |
| \t | Horizontal Tab |
| \v | Vertical Tab |

# Trigraph Character


| Trigraph Sequence | Translation |
|---|---|
| ??= | # |
| ??( | [ |
| ??) | ] |
| ??< | { |
| ??> | } |
| ??! | \| |
| ??/ | \ |
| ??' | ^ |
| ??- | ~ |

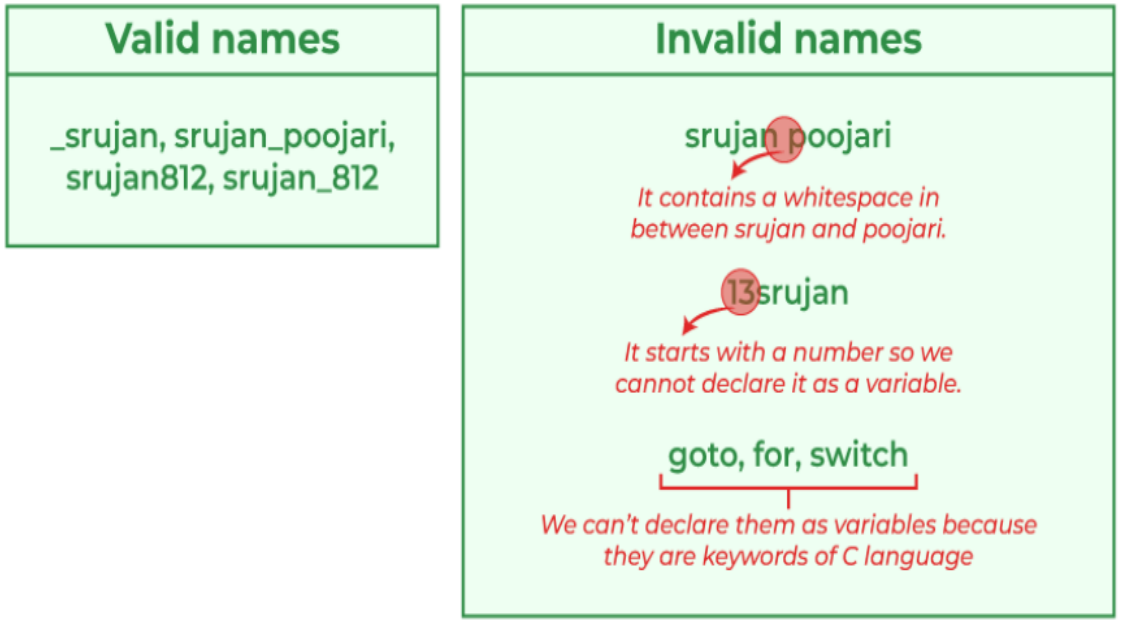
# Delimiters

| | | |
|---|---|---|
| : | colon | used for label |
| ; | semicolon | end of statement |
| ( ) | parentheses | used in expression |
| [ ] | square brackets | used for array |
| { } | curly braces | used for block of statements |
| # | hash | preprocessor directive |
| , | comma | variable delimiter |

# Reserved Words / Keywords (Total: 44)

# Identifiers: **User defined names**

1. Consists of letters (a-z or A-Z), and digits (0-9).
2. Exclude special characters except the '_' underscore.
3. Spaces are not allowed while naming an identifier.
4. Can only begin with an underscore or letters.
5. Cannot name identifiers the same as keywords
6. The identifier must be **unique** in its namespace.
7. C language is case-sensitive so, 'name' and 'NAME' are different identifiers.



# Data types

| Type | Size (bits) | Size (bytes) | Range |
|---|---|---|---|
| char | 8 | 1 | -128 to 127 |
| unsigned char | 8 | 1 | 0 to 255 |
| int | 16 | 2 | $-2^{15}$ to $2^{15}-1$ |
| unsigned int | 16 | 2 | 0 to $2^{16}-1$ |
| short int | 8 | 1 | -128 to 127 |
| unsigned short int | 8 | 1 | 0 to 255 |
| long int | 32 | 4 | $-2^{31}$ to $2^{31}-1$ |
| unsigned long int | 32 | 4 | 0 to $2^{32}-1$ |
| float | 32 | 4 | 3.4E-38 to 3.4E+38 |
| double | 64 | 8 | 1.7E-308 to 1.7E+308 |
| long double | 80 | 10 | 3.4E-4932 to 1.1E+4932 |

# Basic Format Specifiers:

- Format specifiers are essential for **proper data representation in input/output operations.**
- They ensure that data types are handled correctly in C programs.
- **Basic Format Specifiers are as follows:**
  1. %d: Integer (decimal)
  2. %c: Character
  3. %f: Floating-point number
  4. %s: String (array of characters)
  5. %lf: Double
  6. %u: Unsigned integer
  7. %x / %X: Unsigned hexadecimal integer

## How to Declare Constants

const int var; ✗

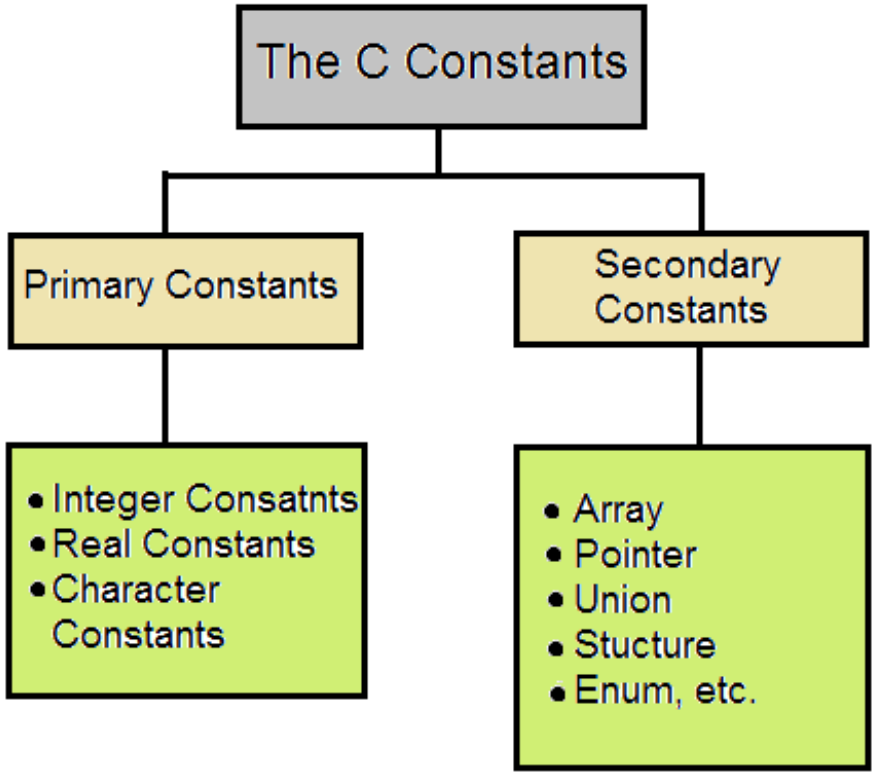
const int var;
var=5 ✗

Const int var = 5; ✓

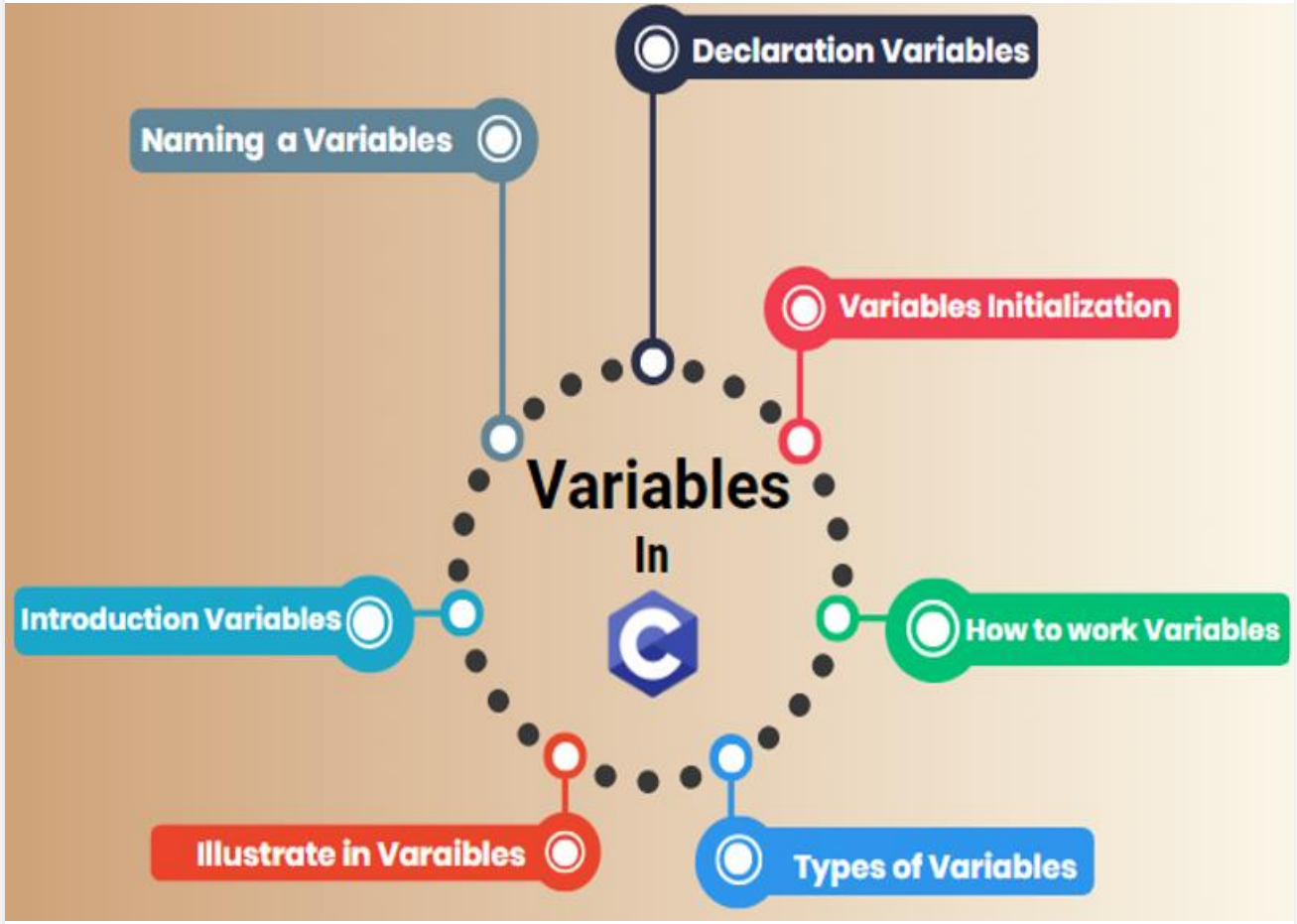
## Constants

Const int var = 5;

- Const — Keyword
- int — Data type
- var — Name of Student
- = 
- 5; — Initial Value




- Read-only variables whose values cannot be modified once they are declared in the C program.
- The **const** keyword is used to define the constants.
- We can only initialize the constant variable in C at the time of its declaration. Otherwise, it will store the garbage value.
- The constant variables are immutable after its definition,




# Variables

- Variables are used to store different forms of data like int, float, char, double, etc.
- It acts as a memory card where it saves all the data and used it during program execution

# Naming of a variable

Must not start with the number

Blank space between variables is not allowed

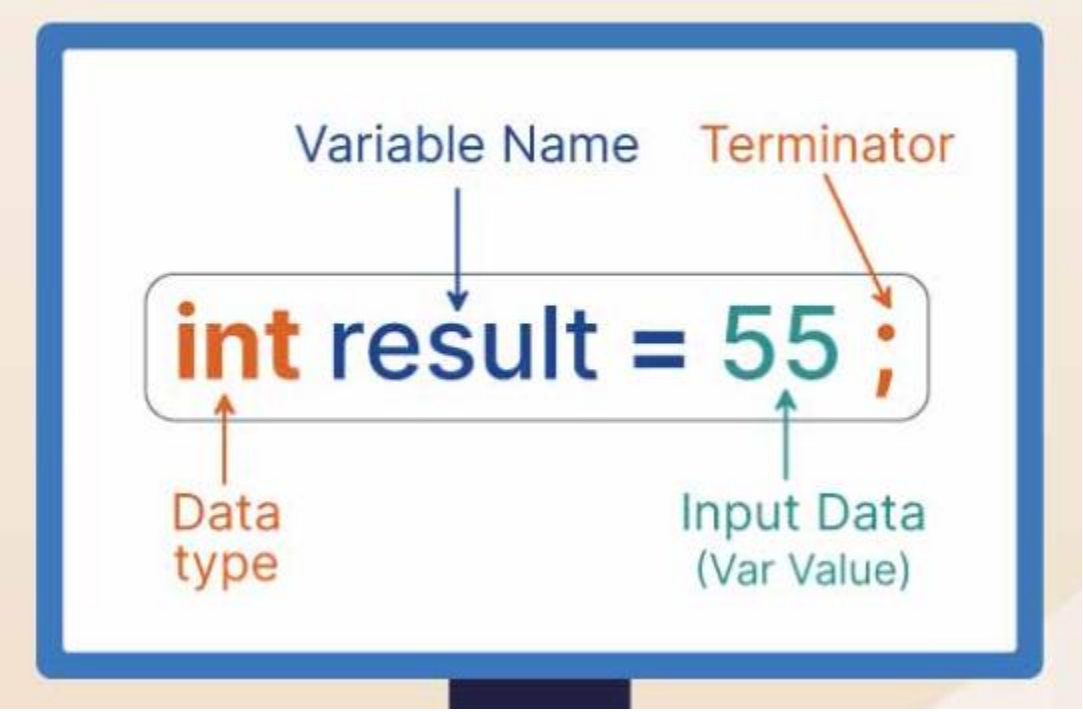
Keywords are not allowed to define as a variable

As C is a case sensitive language, upper and lower cases are considered as a different variable.

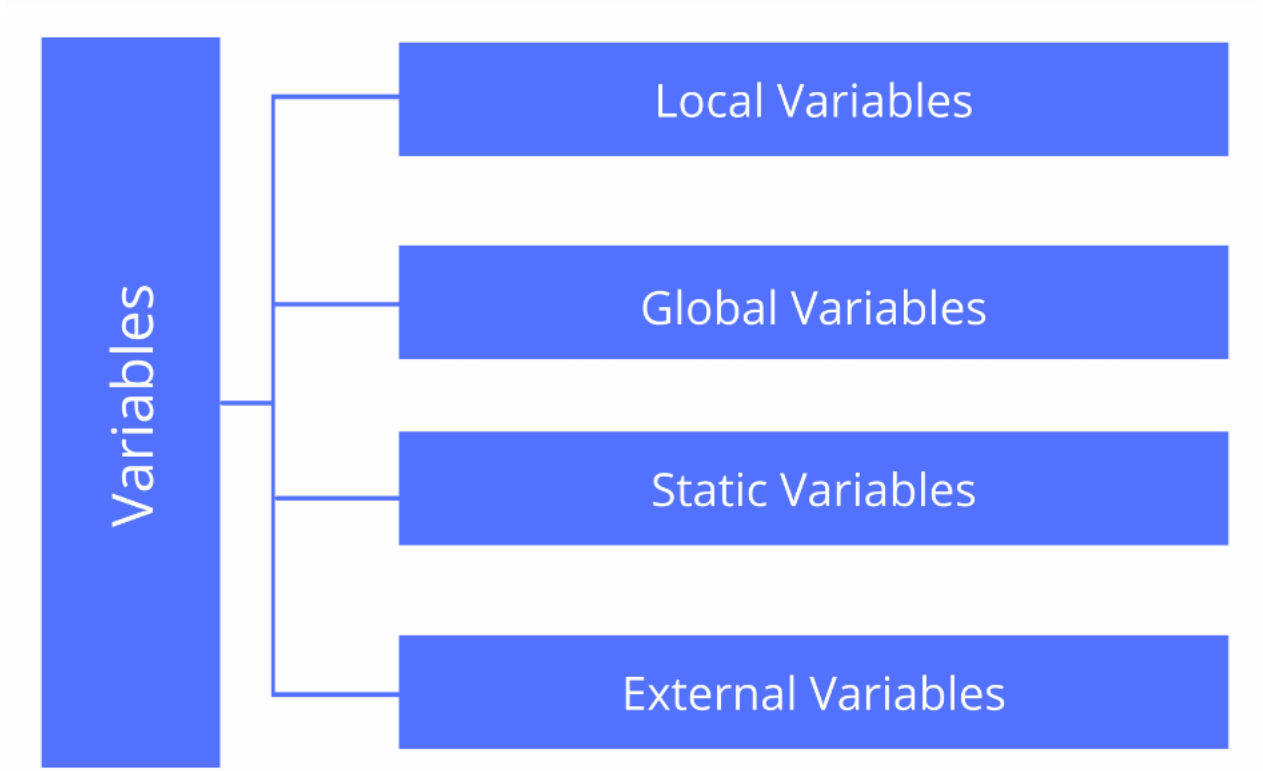
Variable names can be a combination of string, digits, and special characters like underscores (_).

# Variable declaration and initialization

- After variables are declared, the space for those variables has been assigned and it is used for the program.



# Types of variables



# Types of Variable

- **Local variable:** Variables declared inside the functions and only local functions can change the value of variables.

- **Global variable:** Variables are declared outside the functions and any functions can change the value of variables.

```
int main()
{
int m =10; //local variable
}
```

```
int n = 6; //global variable
int main()
{
int m =10; //local variable
}
```

# Types of Variable

- **Static variable:** Declared with the static keyword

```
int main()
{
int m =10; //local variable
static n = 6; //static variable
}
```

- **External variable:** Declared using the extern keyword which can be used in multiple C source files.

```
extern m =10; //external variable
```

# Identifier Vs Variable

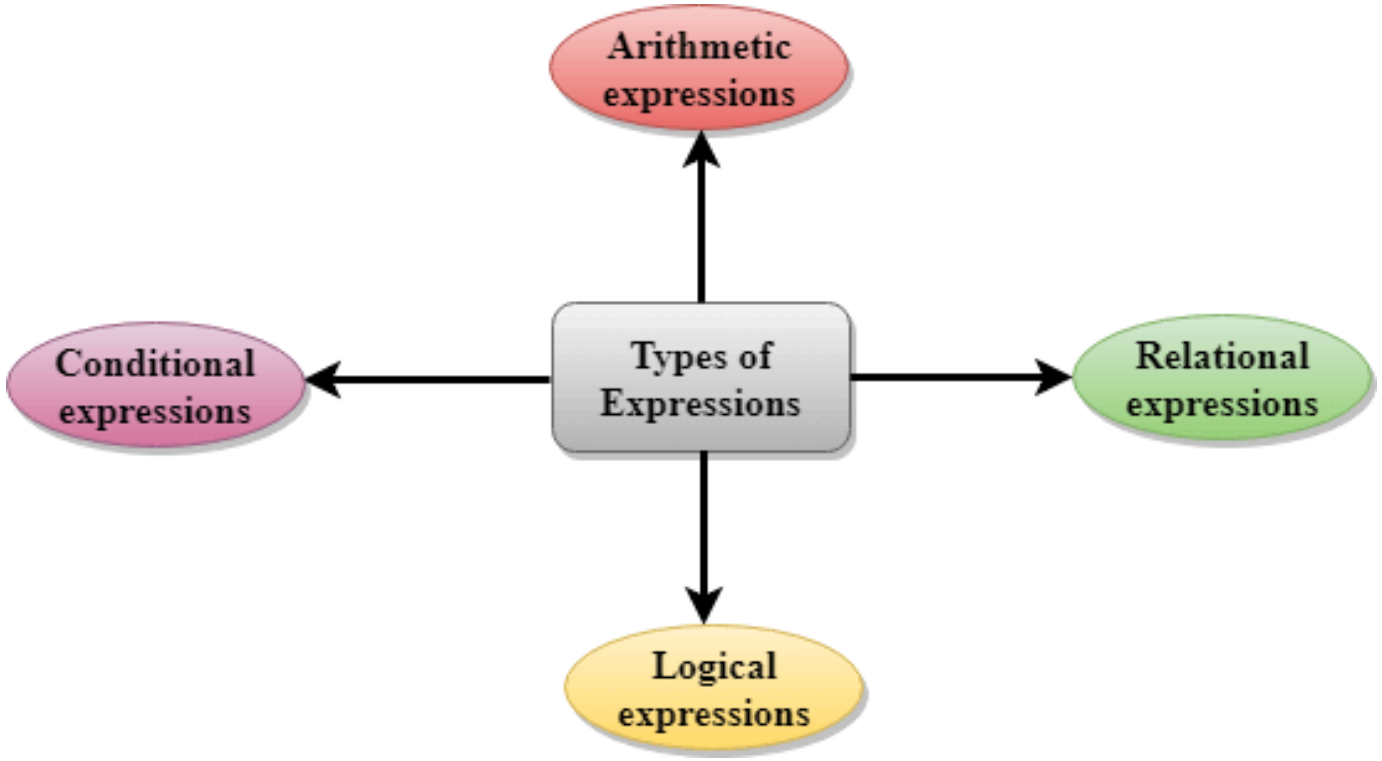
An identifier is a **name used to identify** a variable, function, array, or any other user-defined item in a program.

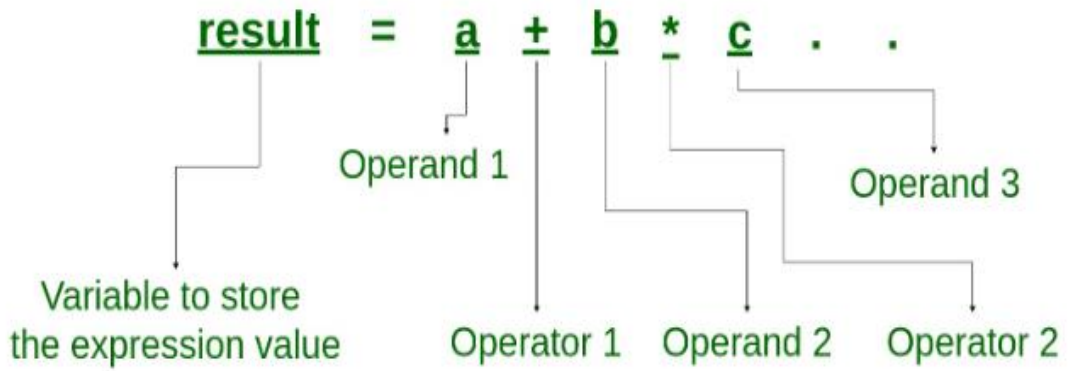
Whereas a variable is a **storage location identified by an identifier** that holds data which can be modified during the program's execution.

All variables in C are identifiers, but not all identifiers are variables.

# Expressions

An expression is a combination of operators, constants, variables and functions






| Expression | Interpretation | Value |
|---|---|---|
| a < b | True | 1 |
| (a + b) >= c | True | 1 |
| (b + c) > (a + 5) | False | 0 |
| c != 3 | False | 0 |
| b == 2 | True | 1 |

| Logical Operators | | |
|---|---|---|
| For all examples below consider a = 10 and b = 5 | | |
| **Operator** | **Description** | **Example** |
| && | Logical AND | (a>b) && (b==5) gives true |
| \|\| | Logical OR | (a>b) \|\| (b==2) gives true |
| ! | Logical NOT | !(b==5) gives false |

```
int a = 10, b = 5, c;

c = (a>b) ? 20 : 12;
```

# Upcoming Slides

- Operators
- Logical Expressions
- Conditional Statements
- Number System