Introduction to Computing and Programming

Introduction to Programming, Identifiers and Constants, Number System

Content

- Quick Recap
- Structure of C language
- Keywords
- Delimiters
- Variables

Recap

- What is Computer?
- **Functions of Computer:**
  - Input,
  - Processing
  - Output
  - Storage
- **Components of Computer:**
  - Hardware
  - Software
  - Users
- **Types of Computers**: Microcomputer, Minicomputer, Personal Computer, Supercomputer, Laptop, Tablets
- **What is Programming & Its importance**
- **Types of Programming Language**
- **History of C**

Why programming is important for all of us?

Automating tasks: D*ata processing, simulation, and optimization*, which helps programmer to save time and focus on more complex tasks.

Problem-solving: Provide ability to solve complex problems using algorithms that would be difficult to solve using traditional methods.

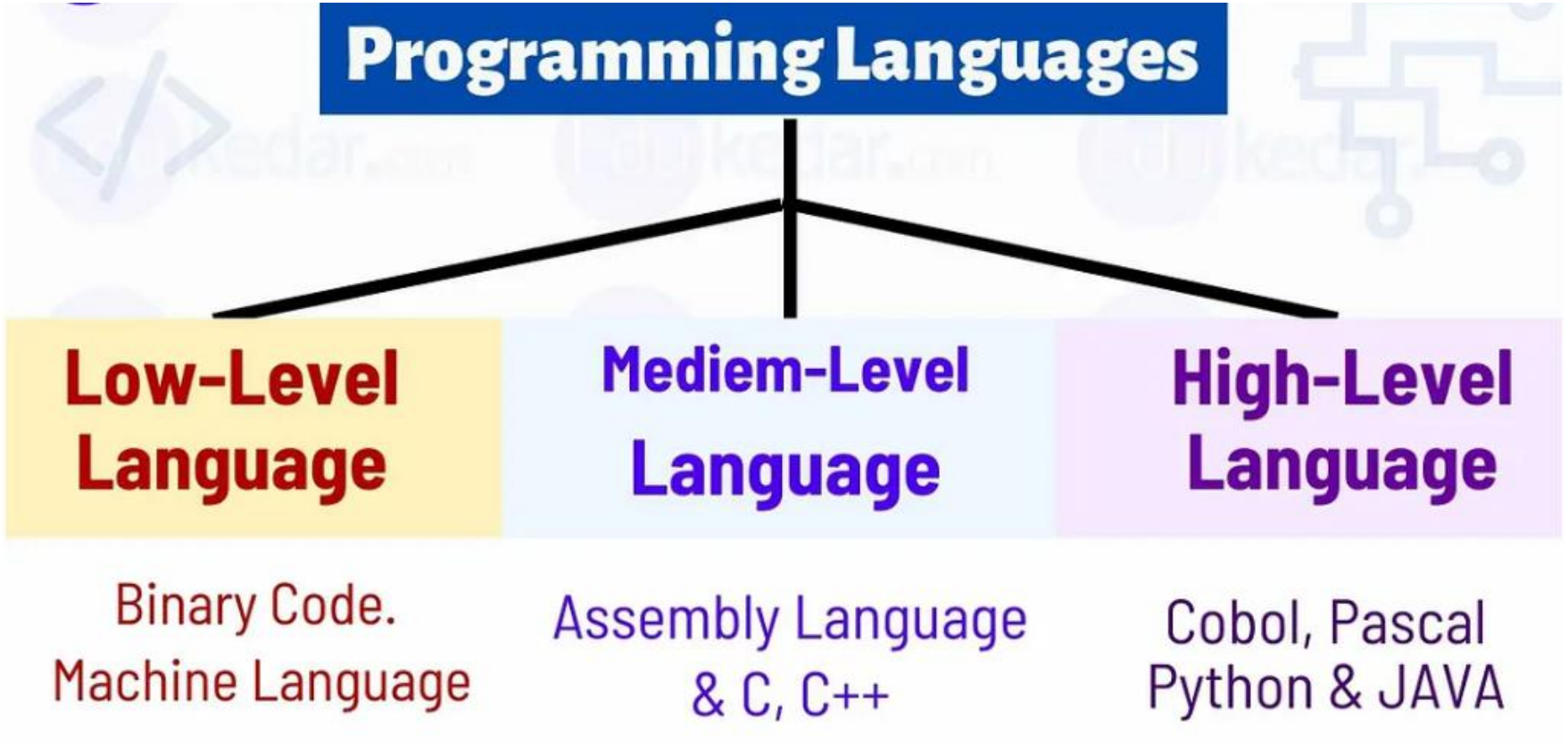
Collaboration: Allows engineers to work collaboratively on projects which saves time and ensures that the code is error-free.

Job opportunities: Knowledge of programming languages can increase an engineering student's job opportunities.

Innovation: Allow engineers to develop new products and technologies.

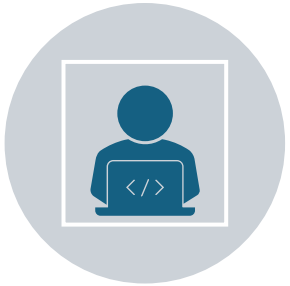
Computer Language - A Way to Talk to a Computer!

Types of Programming Language



Introduction to C

C is a general purpose and structured programming language.



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

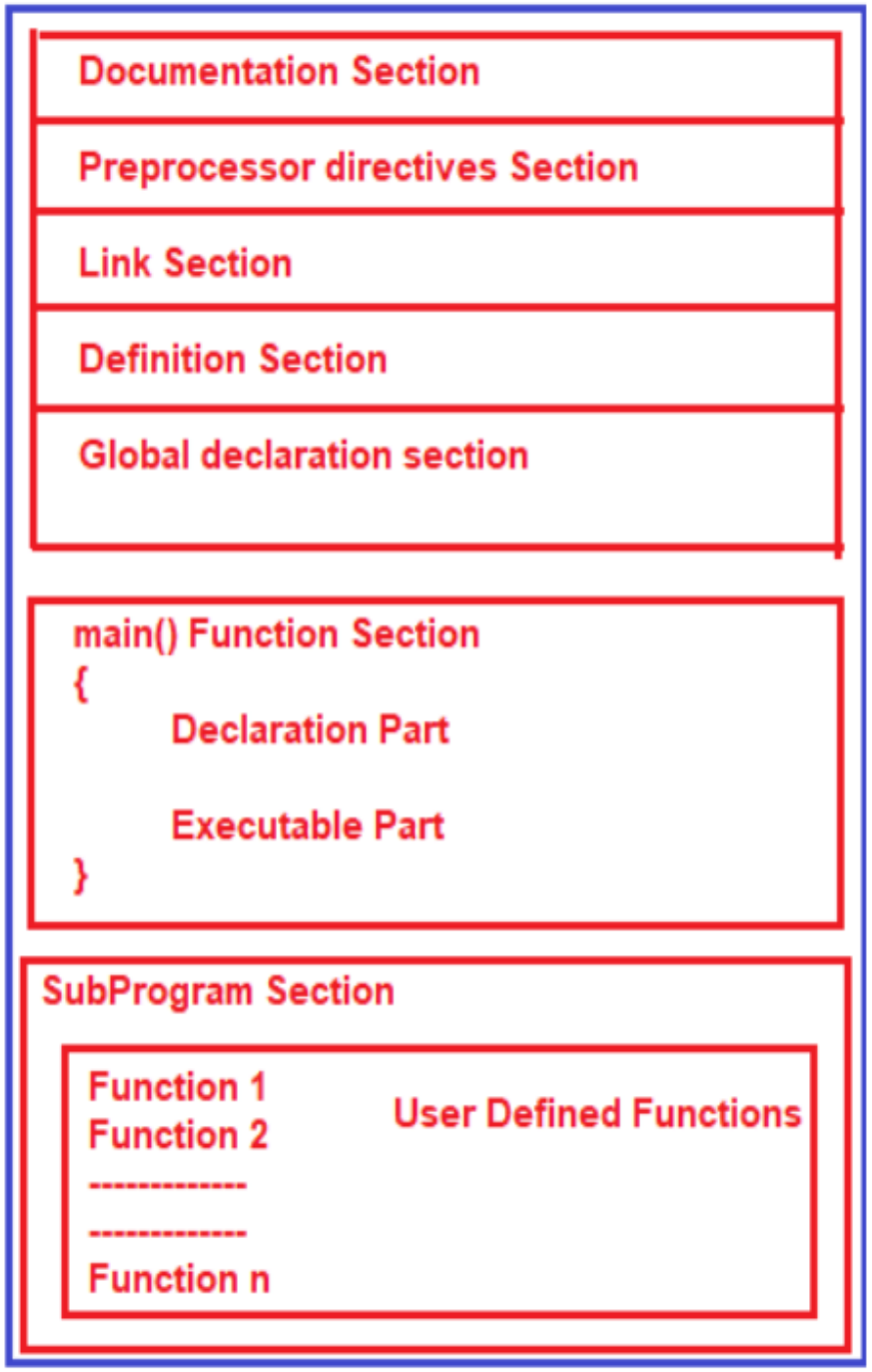
It can be used to write very concise source program due to the availability of extensive libraries.

It is highly portable.

History of C Language

- Developed in 1970's by Dennis Ritchie at AT & T Bell Laboratories.
- It became popular in mid 1980's with the availability of compilers for various platforms.
- Some standardization has been made for C implementation
  - ANSI
  - GNU Compiler Collection **(GCC)**
  - C11 → C17/18 → **C23 (latest)**

Language	Year	Developed By
Algol	1960	International Group
BCPL	1967	Martin Richard
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K & R C	1978	Kernighan & Dennis Ritchie
ANSI C	1989	ANSI Committee
ANSI/ISO C	1990	ISO Committee
C99	1999	Standardization Committee



Structure of C Program

Example

```c
#include <stdio.h>

int main () {

      int a;
      int b;
      /* for loop execution */
      for( a = 1; a < 6; a++ )
      {
            /* for loop execution */
            for ( b = 1; b <= a; b++ )
            {
                  printf("%d",b);
             }
             printf("\n");
      }

      return 0;
}
```

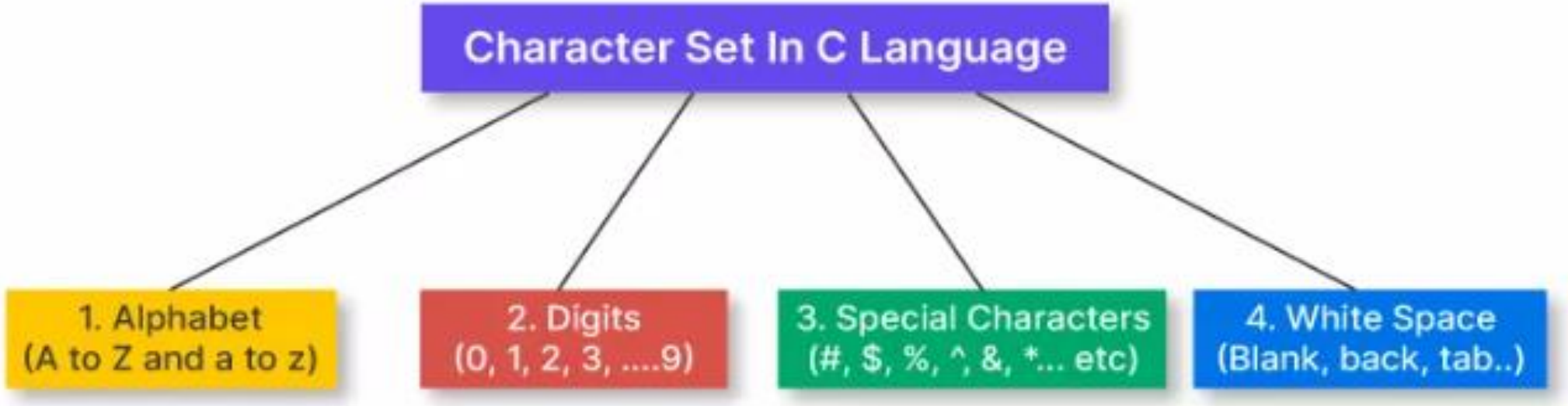
OUTPUT →

```
1
12
123
1234
12345
```

Elements of C

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

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

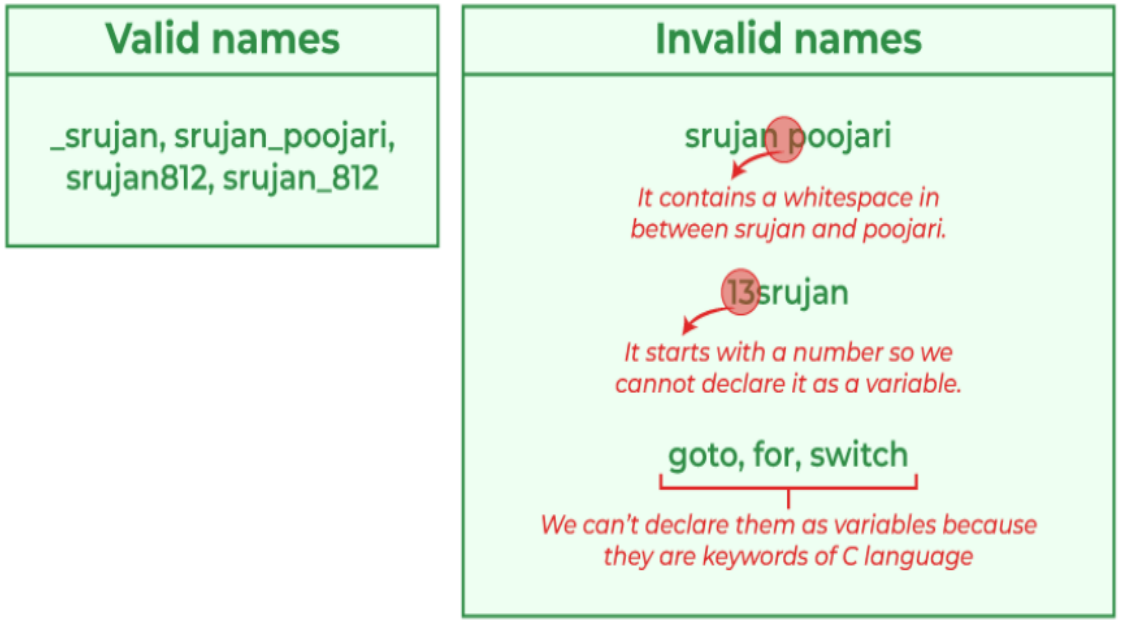
Delimiters

:	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

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

How to Declare Constants

const int var; ✗

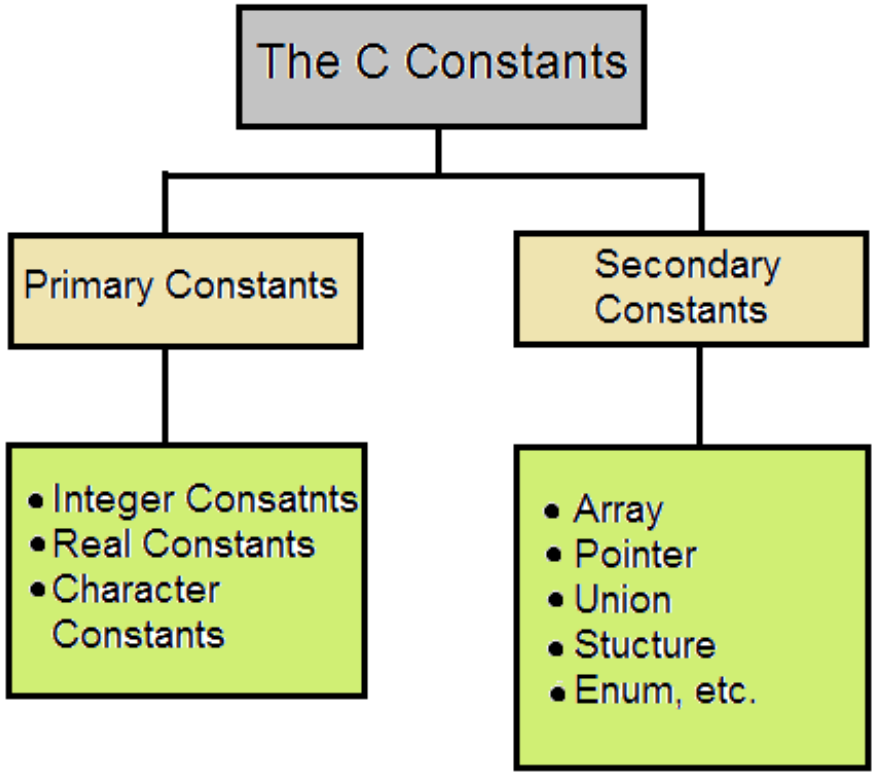
const int var;
var=5 ✗

Const int var = 5; ✓

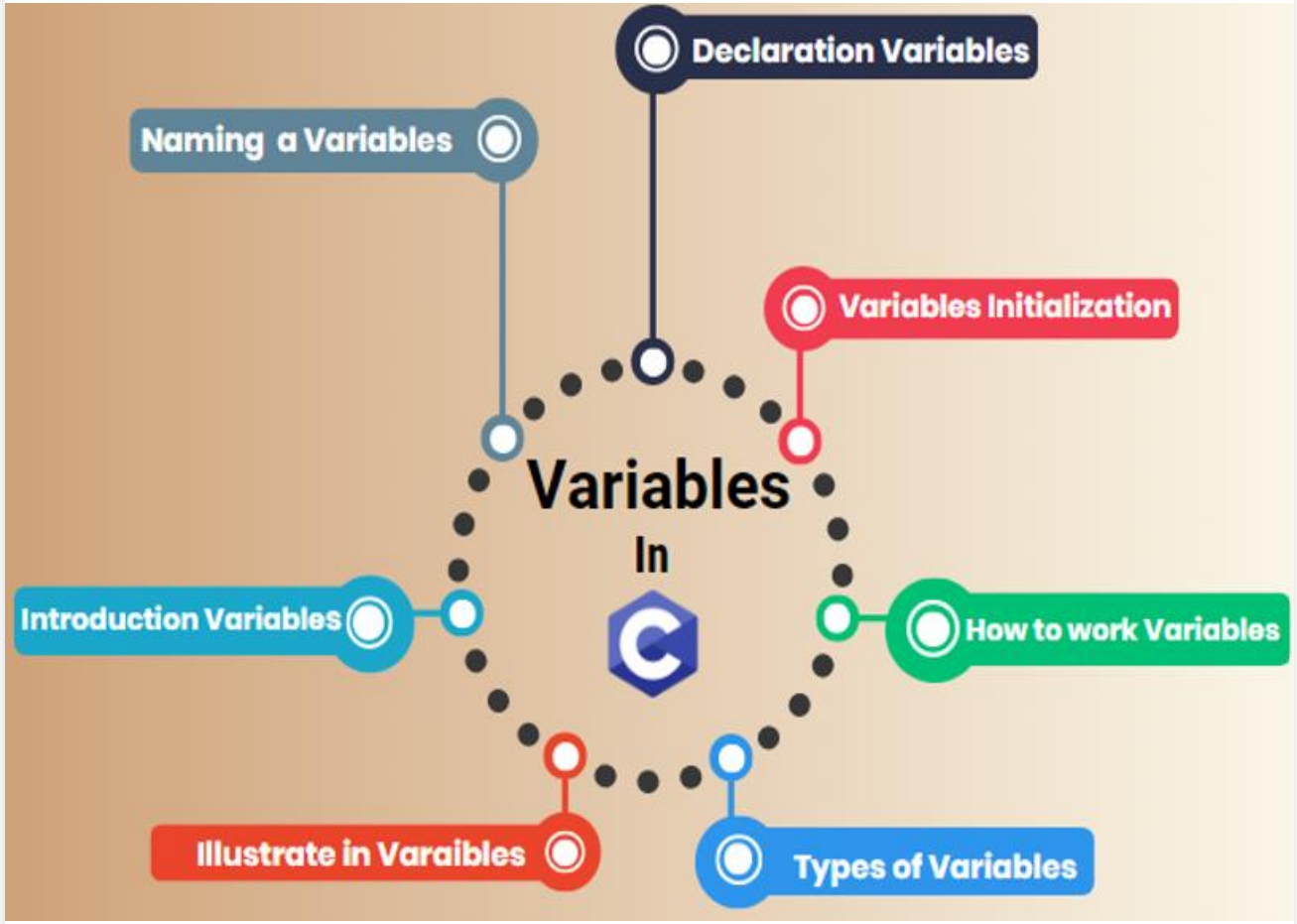
Constants

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

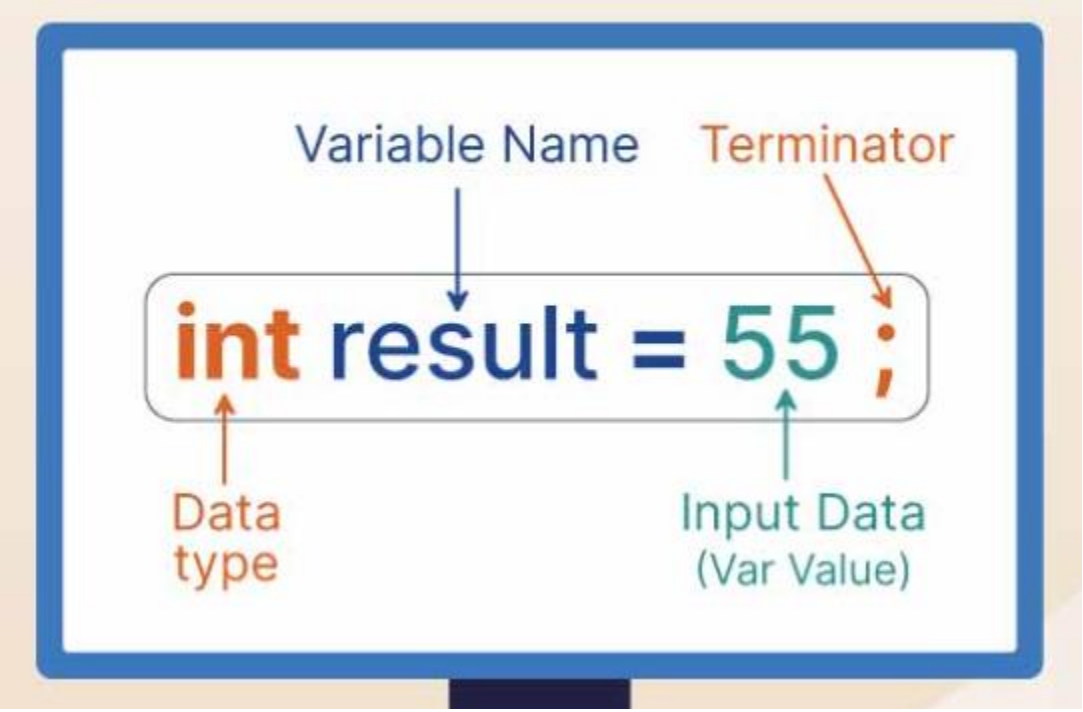
Keywords are not allowed to define as a variable

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

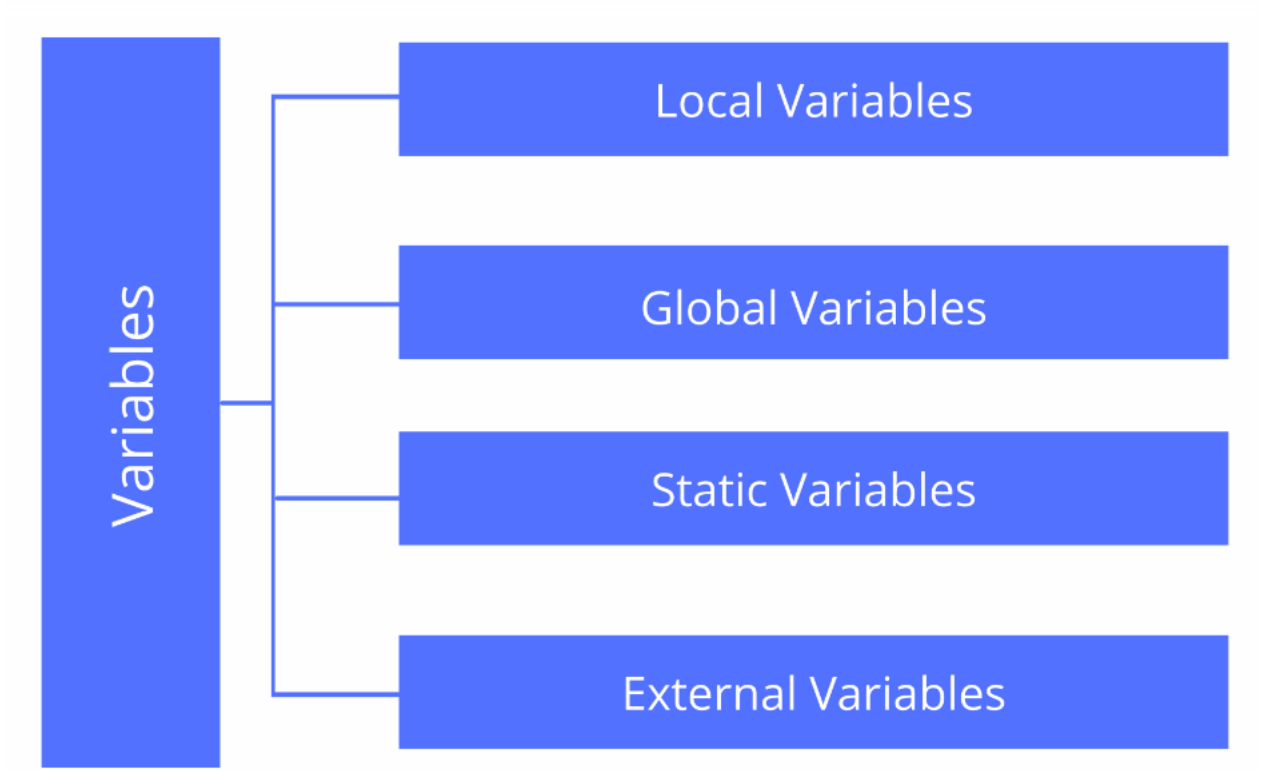
Variable names can be a combination of string, digits, and special characters like underscores (_).

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
```

Upcoming Slides

- Conditional Statements
- Expressions
- Comments
- Operators
- Number System