


# Keyword, Variable and Data Type

## Unit 2

### INTRODUCTION

C tokens, Identifiers and Keywords are the basics and are part of the syntax in a C program. All are explained in this unit with definition and simple example programs. Also, you will learn about variables, rules for naming a variable, constants and different type of constants in C programming. Through this unit you will learn about data types which are very important in programming. C data types are defined as the data storage format that a variable can store a data to perform a specific operation. Size of variable, constant and array are determined by data types. C language has some predefined set of data types to handle various kinds of data that we use in our program.

<p style="text-align: center;">Timeframe</p> <div style="text-align: center;">  </div> <p style="text-align: center;">How long ?</p>	<p>We expect that this unit will take maximum 8 hours to complete.</p>
---	--

<b>Unit Structure</b>	<ul style="list-style-type: none"> <li>Lesson- 1 : Keyword and Identifier</li> <li>Lesson- 2 : Variable</li> <li>Lesson- 3 : Constant</li> <li>Lesson- 4 : Data type</li> </ul>
-----------------------	---

**Lesson -1****Keyword and Identifier****Learning Outcomes****Outcomes**

Upon completion of this lesson you will be able to

- Define token.
- Define identifier.
- Define keyword.

	<b>Keywords</b>	<b>Identifier, Token</b>
---	-----------------	--------------------------

**TOKENS IN C**

A C program consists of various tokens and a token is either keywords, an identifiers, a constants, a strings literal, operator or a symbols.

**KEYWORDS**

Keywords are pre-defined words in a C compiler. Each keyword is meant to perform a specific function. Different programming languages have different set of reserved keywords, but there is one important and common rule in all the programming languages that these reserved words cannot be used as constants or variables or any other identifier names. So we should keep in mind that; while giving a name to your variable, constants or any other identifier you should not use any reserved keyword for that programming language. The following list shows the reserved words in C.

auto	else	long	switch	break	enum	register	typedef	
case	extern	return	union	char	float	short	unsigned	
const	for	signed	void	continue	goto	sizeof	volatile	
default	if	static	while	do	int	struct	_Packed	double



Note it!

---

*Keywords cannot be used as constants or variables or any other identifier names.*

---

**IDENTIFIER**

In C any name is called identifier. This name can be variable name, function name, enum constant name, micro constant name, goto label name, any other data type name like structure, union, enum names or typedef name. In other words in C, identifier is a name used to identify a variable, function, or any other user defined item.

**Rules for constructing identifiers**

- First character must be an alphabet or underscore and subsequent characters must be either letter, digits, or underscore.

- C does not allow punctuation characters such as @, \$, and % within identifiers.
- Only 31 characters are significant.
- Can not be the same as C keywords and should not have the same name as functions that are in C library.
- C is a case-sensitive programming language. Thus, Name, names, NAME are three different identifiers in C.
- Must not contain white space

Here are some examples of acceptable identifiers:

mohd zara abc move\_name a\_123  
 myname50 \_temp j a23b9 retVal



1. Let's do simple exercise. In this exercise we will match valid and invalid identifier.

**Match the column**

valid identifier	(a)_no
	(b) 34_
	(c) char
invalid identifier	(d) _char
	(e)num ber
	(f) @no

2. Determine which of the following are valid identifiers. If invalid, explain why.

- |                     |                     |
|---------------------|---------------------|
| a) record1          | b) 1record          |
| c) file_3           | d) return           |
| e) \$tax            | f) name             |
| g) name and address | h) name_and_address |
| i) name-and-address | j) 123-45-6789.     |

## Lesson-2 Variable


### Learning Outcomes



### Outcomes

Upon completion of this lesson you will be able to

- Define variable.
- Declare variables and assign values to variables.
- Write programs using variables.

	<b>Keywords</b>	<b>Variable, Declaration</b>
---	-----------------	------------------------------

### VARIABLE

Computer programs usually work with different types of data and need a way to store the values being used. These values can be numbers or characters. C has two ways of storing number values such as variables and constants with many options for each. A variable is a data storage location that has a value that can change during program execution. So finally we can say that an identifier whose value is allowed to change during the execution of a program is called a variable. Every variable must be individually declared (i.e., defined) before it can appear in a statement.

In other word we can say variable is container of data. In real world you have used various type containers for specific purpose. For example you have used match box to store match sticks, file cabinet to store files etc. In the same way variables of different data type is used to store different types of data. As for example integer variables are used to store integers char variables is used to store characters etc. Before you get to variables, you need to know a little about the operation of your computer's memory. Hope all of us already know that computer uses random access memory (RAM) to store information while it's operating. Each computer has a certain amount of RAM installed. RAM is located in integrated circuits, or chips, inside your computer. RAM is volatile, which means that it is erased and replaced with new information as often as needed. Being volatile also means that RAM "remembers" only while the computer is turned on and loses its information when you turn the computer off. The amount of RAM in a system is usually specified in kilobytes (KB) or megabytes (MB), or gigabytes (GB) and so on One kilobyte of memory consists of 1,024 bytes. Thus, a system with 640KB of memory actually has  $640 * 1,024$ , or 65,536, bytes of RAM. One megabyte is 1,024 kilobytes. A machine with 4MB of RAM would have 4,096KB or 4,194,304 bytes of RAM.

The RAM in computer is organized sequentially, one byte following another. Each byte of memory has a unique address by which it is identified an address that also distinguishes it from all other bytes in memory. Now that you understand a little about the nuts and bolts of memory storage, you can get back to C programming and how C uses memory to store information.



In programming, a variable is a container (storage area) to hold data. To indicate the storage area, each variable should be given a unique name (identifier). Variable name is just the representation of a memory location. Every variable must be individually

Study skills

declared (i.e., defined) before it can appear in a statement and value of the variable can change during program execution.

For example:

```
int number = 15;
```

Here, *number* is a variable of integer type. The variable is assigned value 15.

### PROPERTIES OF VARIABLE IN C:

Every variable in c have three most fundamental attributes. They are:

1. Name
2. Value
3. Address

### VARIABLE DECLARATIONS

C programming language requires a variable creation, i.e., declaration before its usage in your program. You cannot use a variable name in your program without declaring it. The purposes of a variable declaration-

- tells the compiler what the variable name is.
- specifies what type of data the variable will hold.

Where variables to be declared:

- Inside function (local variables)
- In the definition of function parameters (Formal parameters)
- Outside of all functions (global variables)

A variable declaration has the following form:

General syntax:

```
type variable_name;
```

Example:

```
int a
float temp;
char flag, ;
```

You can declare multiple variables of the same type on one line by separating the variable names with commas:

```
int count, number, start; /* three integer variables */
float percent, total; /* two float variables
```



Note it!

---

***If your program attempts to use a variable that hasn't been declared, the compiler generates an error message.***

---

### STORING VALUES IN VARIABLES

You have seen how to declare variables in the previous section. Now, let's store some values in those variables:

```
int a;
int b;
a = 10;
b = 20;
```

Here we are storing 10 in variable a and 20 in variable b. Almost all the programming languages have similar way of storing values in variable where we keep variable name in the left side of an equal sign = and value we want to store in the variable, we keep that value in the right side. When above statement is executed, the memory location named a will hold 10 and memory location b will hold 20.

Some more examples:

```
int a, b, sum;
a = 50;
b = 60;
sum = a + b;
```

We can also assign value at the time of declare as follows

```
int a = 50, b = 60, sum;
sum = a + b;
```

### CATEGORY OF C VARIABLE

On the basis of how many data a variable will store, we can categorize the all c variable in three groups.

1. Variables which can store only one data at time. Example: integer variables, char variables, pointer variables etc.
2. Variables which can store more than one data of similar type at a time. Example: array variables
3. Variables, which can store more than one value of dissimilar type at a time. Example: structure or union variables.

Not two variables in c can have same name with same visibility. For example:

```
#include<stdio.h>
int main(){
    auto int a=5; //Visibility is within main block
    static int a=10; //Visibility is within main block
    /* Two variables of same name */
    printf("%d",a);
    return 0;
}
```

### Output: compilation error

But it is possible that two variable with same name but different visibility. In this case variable name can access only that variable which is more local. In c there is not any way to access global variable if any local variable is present of same name. For example:

(a)

```
#include<stdio.h>
int a=50; //Visibility is whole the program
int main(){
    int a=10; //Visibility within main block
    printf("%d",a);
    return 0;
}
```

Output: 10

(b)

```
#include<stdio.h>
int main(){
    int a=10; //Visibility within main block.
    {
        a=a+5; //Accessing outer local variable a.
        int a=20; //Visibility within inner block.
        a=a+10; //Accessing inner local variable a.
```

```
printf(“%d”,a);//Accessing inner local variable a.
}
printf(“%d”,a); //Accessing outer local variable a.
return 0;
}
```

Output: 30 15

### RULES FOR NAMING VARIABLES

To use variables in your C programs, you must know how to create variable names. In C, variable names must adhere to the following rules:

- The name can contain letters, digits, and the underscore character (\_).
- The first character of the name must be a letter. The underscore is also a legal first character, but its use is not recommended.
- Case matters (that is, upper- and lowercase letters). Thus, the names count and Count refer to two different variables.
- C keywords can't be used as variable names. A keyword is a word that is part of the C language

The following list contains some examples of legal and illegal C variable names:

Variable	Name	Legality
Percent	Percent	Legal
y2x5__	fg7h	Legal
annual_	profit	Legal
_1990_tax	Legal	but not advised
savings#account	Illegal:	Contains the illegal character #
double	Illegal:	Is a C keyword
9winter	Illegal:	First character is a digit



Study skills

Let's consider a simple example. In this example we take two integers, add them and display the answer on the screen. The code of the program is written below.

```
#include <iostream.h>
main()
{
int x;
int y;
int z;
x = 5;
y = 10;
z = x + y;
}
```

The first three lines declare three variables x, y and z. These three declarations can also be written on one line. C provides us the comma separator (.). The above three lines can be written in a single line as below

```
int x, y, z;
```

- As we know that semicolon (;) indicates the end of the statement. So we can write many statements on a single line. In this way we can also write the above declarations in the following form

```
int x; int y; int z;
```

- For good programming practice, write a single statement on a single line.
- We have declared all the variables in a single line by using comma separator (.). This is a short method to declare a number of variables of the same data type.

## Lesson-3 Constant

### Learning Outcomes



### Outcomes

Upon completion of this lesson you will be able to

- Define constant.
- Declare constant.
- Apply escape sequence.

	<b>Keywords</b>	<b>Constants, Escape Sequence</b>
--	-----------------	-----------------------------------

### CONSTANT

Like a variable, a constant is a data storage location used by your program. As its name implies, the value stored in a constant can't be changed (fixed) during program execution. Constants (i.e., fixed values) are also called literals. A constant is a value or an identifier whose value cannot be altered in a program.

For example: 1, 2.5, "C programming", etc.

As mentioned, an identifier also can be defined as a constant.

```
const double PI = 3.14
```

### INTEGER CONSTANT

An integer constant is a numeric constant (associated with number) without any fractional or exponential part. There are three types of integer constants in C programming:

- decimal constant(base 10)
- octal constant(base 8)
- hexadecimal constant(base 16)

For example:

Decimal constants: 0, -9, 22 etc

Octal constants: 021, 077, 033 etc

Hexadecimal constants: 0x7f, 0x2a, 0x521 etc

In C programming, octal constant starts with a 0 and hexadecimal constant starts with a 0x.

### FLOATING-POINT CONSTANT

A floating point constant is a numeric constant that has either a fractional form or an exponent form.

For example:

-2.0

0.0000234

-0.22E-5

### CHARACTER CONSTANT

A character constant is a constant which uses single quotation around characters. For example: 'a', 'l', 'm', 'F'

### STRING CONSTANT

String constants are the constants which are enclosed in a pair of double-quote marks. For example:

```
"Bangladesh" //string constant
```

```
"" //null string constant
```

```
" " //string constant of six white space
```



```
"x"           //string constant having single character.
"Bangladesh Open University\n" //prints string with newline
```

### ESCAPE SEQUENCES

There are some characters which we use very frequently but they are invisible in your program and these characters are spaces, tabs (\t), new lines (\n). These characters are called **whitespaces**. These three important whitespace characters are common in all the programming languages and they remain invisible in your text document. In order to use these characters, escape sequence is used.

Escape Sequences	
Escape Sequences	Character
\b	Backspace
\f	Form feed
\n	Newline
\r	Return
\t	Horizontal tab
\v	Vertical tab
\\	Backslash
\'	Single quotation mark
\"	Double quotation mark
\?	Question mark
\0	Null character



Activity

1. Determine which of the following are valid character constants?

- ``a``
- ``${``
- ``\n``
- ``/n``
- ````
- ``\a``
- ``T``
- ``\0``
- ``xyz``
- ``\052``

2. Determine which of the following are valid string constants?

- ``8:15 P.M.``
- ```Red, White and Blue```
- ```Name:```
- ```Chap. 3 (Cont\'d)```
- ```1.3e-12```
- ```NEW YORK, NY 10020```
- ```The professor said, "Please don't sleep in class"```

## Lesson-4 Data Type


### Learning Outcomes



### Outcomes

Upon completion of this lesson you will be able to

- Define data type.
- Describe the data type in C.
- Understand the declaration of data type.

	<b>Keywords</b>	<b>integer, float, double.</b>
---	-----------------	--------------------------------

### INTRODUCING C DATA TYPE

Every programming language deals with some data. For example to print any message it requires character or string type of data. To solve any mathematic expression it requires integral as well as real number (floating type) of data. C is very rich in data type. Data types specify how we enter data into our programs and what type of data we enter. C language has some predefined set of data types to handle various kinds of data that we use in our program. These data types have different storage capacities. Data types are used to define a variable before to use in a program. Size of variable, constant and array are determined by data types. There are four data types in C language.

### C DATA TYPES

Types	Data Types
Basic data types	int, char, float, double
Enumeration data type	enum
Derived data type	pointer, array, structure, union
Void data type	void

### INTEGER DATA TYPE

- Integer data type allows a variable to store numeric values.
- “int” keyword is used to refer integer data type.
- The storage size of int data type is 2 or 4 or 8 byte.
- It varies depend upon the processor in the CPU that we use. If we are using 16 bitprocessor, 2 byte (16 bit) of memory will be allocated for int data type. Like wise, 4 byte (32 bit) of memory for 32 bit processor and 8 byte (64 bit) of memory for 64 bit processor is allocated for int datatype.
- int (2 byte) can store values from -32,768 to +32,767
- int (4 byte) can store values from -2,147,483,648 to +2,147,483,647.
- If you want to use the integer value that crosses the above limit, you can go for “long int” and “long long int” for which the limits are very high.



Note it!

*We can't store decimal values using int data type. If we use int data type to store decimal values, decimal values will be truncated and we will get only whole number. In this case, float data type can be used to store decimal values in a variable.*

#### FLOATING POINT DATA TYPE:

Floating point data type consists of 2 types. They are,

1. float
2. double

#### FLOAT

- Float data type allows a variable to store decimal values.
- Storage size of float data type is 4. This also varies depend upon the processor in the CPU as "int" data type.
- We can use up-to 6 digits after decimal using float data type.
- For example, 10.456789 can be stored in a variable using float data type.

#### DOUBLE

- Double data type is also same as float data type which allows up-to 10 digits after decimal.
- The range for double datatype is from  $1E-37$  to  $1E+37$

#### CHARACTER DATA TYPE

So far we have been looking on data types to store numbers, In programming we do need to store characters like a,b,c etc. For storing the character data C language provides char data type. By using char data type we can store characters in variables. While assigning a character value to a char type variable single quotes are used around the character as 'a'.

- Character data type allows a variable to store only one character.
- Storage size of character data type is 1. We can store only one character using character data type.
- "char" keyword is used to refer character data type.
- For example, 'A' can be stored using char datatype. You can't store more than one character using char data type.

Let's take a look into different data types that the C language provides us to deal with whole numbers, real numbers and character data.

#### **int - data type**

**int** is used to define integer numbers.

```
#include <stdio.h>
int main()
{
    int number;
    printf("Enter an integer: ");
    scanf("%d",&number);
    printf("Entered Number is = %d",number);
    return 0;
}
```

#### **Output**

```
Enter an integer: 4
Entered Number is = 4
```

#### **float - data type**

**float** is used to define floating point numbers.

```
#include <stdio.h>
int main()
{
    float f;
    printf("Enter a number: ");
    scanf("%f",&f);
    printf("Value = %f", f);
    return 0;
}
```

**Output**

```
Enter a number: 23.45
Value = 23.450000
```

**double - data type**

**double** is used to define BIG floating point numbers. It reserves twice the storage for the number.

```
{
    double atoms;
    atoms = 2500000;
}
```

**char - data type**

```
#include <stdio.h>
int main()
{
    char chr;
    printf("Enter a character: ");
    scanf("%c",&chr);
    printf("You entered %c.",chr);
    return 0;
}
```

**Output**

```
Enter a character: g
You entered g.
```

```
{
    char letter;
    letter = 'x';
}
```


**DERIVED DATA TYPE IN C LANGUAGE**

Array, pointer, structure and union are called derived data type in C language. You will learn about these from the next units.

**Size of data types in c**

DATA TYPE	SIZE (IN BYTE)
int	2
short int	2
long int	4
float	4
double	8
long double	10
char	1

enum	2
------	---

<p><b>Summary</b></p>  <p>Summary</p>	
<p><b>In this lesson</b></p> <ul style="list-style-type: none"> <li>▪ We have understood about C keywords and data types.</li> <li>▪ Also understood C constants and variables.</li> <li>▪ Also understood that how variables are declared in program.</li> </ul>	

**Assessment**



Assessment

**Fill in the blanks**

1. A character constant which uses .....around characters.
2. Variable is a .....to hold data.

**Multiple Choice Questions (MCQ)**

1. Which of the following are tokens in C?
  - a) Keywords
  - b) Variables
  - c) Constants
  - d) All of the above
2. Which of the following is not a keyword
  - a) void
  - b) int
  - c) main
  - d) for

**Exercises**

1. Describe the keywords and identifiers in C.
2. What is variable? Discuss how to declare variables and assign values to variables with an example.
3. What is constant?
4. What is the difference between a constant and a variable?
5. What is a string constant? How do string constants differ from character constants? Do string constants represent numerical values?
6. Write simple programs using constants and variables
7. What is data type? Describe the data types in C.?