Andhra Pradesh State Skill Development Corporation

Programming in C

Varibles and Data Types

Variables and Data types

Variable:

A variable is nothing but a name given to a storage area that our programs can manipulate. Each variable in C has a specific type, which determines the size and layout of the variable's memory; the range of values that can be stored within that memory; and the set of operations that can be applied to the variable.

The name of a variable can be composed of letters, digits, and the underscore character. It must begin with either a letter or an underscore. Upper and lowercase letters are distinct because C is case-sensitive. Based on the basic types explained in previous chapter, there will be the following basic variable types:

Type	Description
char	Typically a single octet byte. This is an integer type.
int	The most natural size of integer for the machine.
float	A single-precision floating point value.
double	A double-precision floating point value.
void	Represents the absence of type.

C programming language also allows us to define various other types of variables, which we will cover in subsequent chapters like Enumeration, Pointer, Array, Structure, Union, etc. For this chapter, let us study only basic variable types.

Definition of Variable in C:

A variable definition means to tell the compiler where and how much to create the storage for the variable. A variable definition specifies a data type and contains a list of one or more variables of that type as follows:

```
type  var_list
```

Here, type must be a valid C data type including char, w_char, int, float, double, bool or any user defined object, etc., and var_list may consist of one or more identifier names separated by commas. Some valid declarations are shown here:

```
int i, j, k;
char c, ch;
float f, salary;
double d;
```

The line int i, j, k; both declares and defines the variables i, j and k; which instructs the compiler to create variables named i, j and k of type int.

Variables can be initialized assignedaninitialvalue in their declaration. The initializer consists of an equal sign followed by a constant expression as follows:

```
type variable_name = value;
```

Examples :

```
extern int d = 3, f = 5;          // declaration of d and f.
int d = 3, f = 5;                 // definition and initializing d and
f. byte z = 22;                   // definition and initializes
z. char x = 'x';                  // the variable x has the value 'x'.
```

For definition without an initializer: variables with static storage duration are implicitly initialized with NULL allbyteshavethevalue0; the initial value of all other variables is undefined.

Variable Declaration:

A variable declaration provides assurance to the compiler that there is one variable existing with the given type and name so that compiler proceeds for further compilation without needing complete detail about the variable. A variable declaration has its meaning at the time of compilation only, compiler needs actual variable declaration at the time of linking of the program.

Rules for defining variables:

- A variable can have alphabets, digits, and underscore.
- A variable name can start with the alphabet, and underscore only. It can't start with a digit.
- No whitespace is allowed within the variable name.
- A variable name must not be any reserved word or keyword, e.g. int, float, etc.

Types of Variables in C:

There are many types of variables in c:

1. local variable
2. global variable
3. static variable
4. automatic variable
5. external variable

Local Variable:

A variable that is declared inside the function or block is called a local variable. It must be declared at the start of the block.

```
void function1(){
int x=10;//local variable
}
```

You must have to initialize the local variable before it is used.

Global Variable:

A variable that is declared outside the function or block is called a global variable. Any function can change the value of the global variable. It is available to all the functions.
It must be declared at the start of the block.

```
int value=20;//global variable
void function1(){
int x=10;//local variable
}
```

Static Variable:

A variable that is declared with the static keyword is called static variable. It retains its value between multiple function calls.

```
void function1(){
int x=10;//local variable
static int y=10;//static variable
x=x+1;
y=y+1;
printf("%d,%d",x,y);
}
```

If you call this function many times, the local variable will print the same value for each function call, e.g, 11,11,11 and so on. But the static variable will print the incremented value in each function call, e.g. 11, 12, 13 and so on.

Automatic Variable:

All variables in C that are declared inside the block, are automatic variables by default. We can explicitly declare an automatic variable using auto keyword.

```
void main(){
int x=10;//local variable (also automatic)
auto int y=20;//automatic variable
}
```

External Variable:

We can share a variable in multiple C source files by using an external variable. To declare an external variable, you need to use **extern keyword**.

```
extern int d = 3, f = 5;        // declaration of d and f.
int d = 3, f = 5;               // definition and initializing d and f.
byte z = 22;                    // definition and initializes z.
char x = 'x';                   // the variable x has the value 'x'.
```

Keywords:

Keywords are predefined, reserved words in C language and each of which is associated with specific features. These words help us to use the functionality of C language. They have special meaning to the compilers. Don't use keywords as a variable name.

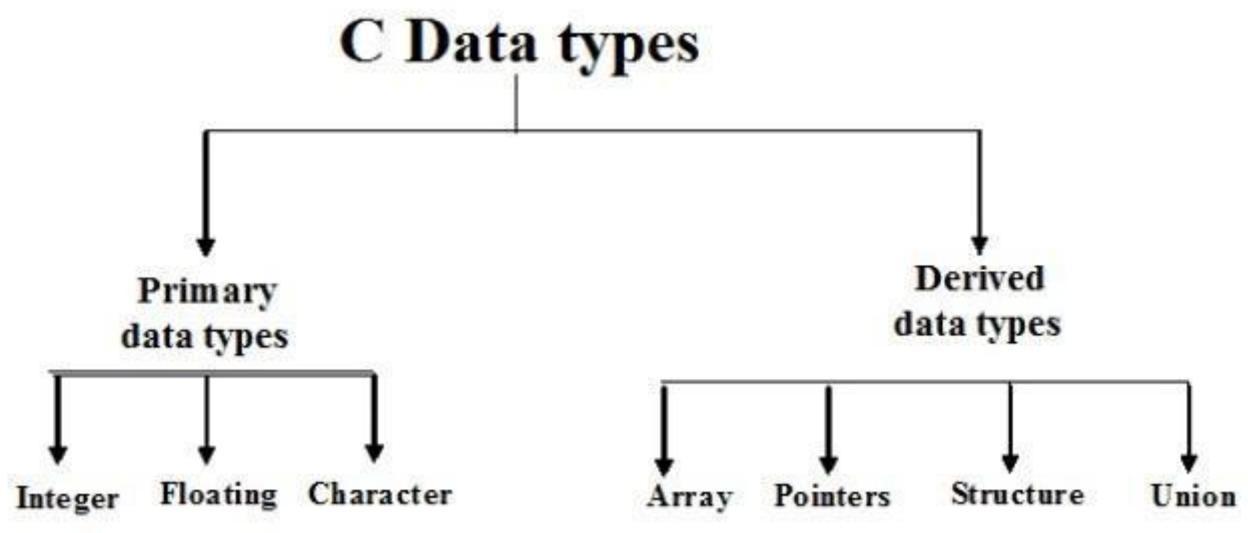
auto	**double**	**int**	**struct**
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Keywords are the words whose meaning has already been explained to the C compiler and their meanings cannot be changed. **Keywords** serve as basic building blocks for **program** statements. **Keywords** can be **used** only for their intended purpose. **Keywords** cannot be **used** as user-defined variables.

Data types in C language :

- C data types are defined as the data storage format that a variable can store a data to perform a specific operation.
- 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.

C has different data types for different types of data and can be broadly classified as:



Primary Data Types:

Integer Data Types:

Integers are whole numbers with a range of values, range of values are machine dependent. Generally an integer occupies 2 bytes memory space and its value range is limited to -32768 to +32767 (that is, -2 15 to +215 -1). A signed integer uses one bit for storing sign and the rest 15 bits for numbers.

To control the range of numbers and storage space, C has three classes of integer storage namely short int, int and long int. All three data types have signed and unsigned forms. A short int requires half the amount of storage than a normal integer. Unlike signed integer, unsigned integers are always positive and use all the bits for the magnitude of the number. Therefore, the range of an unsigned integer will be from 0 to 65535. The long integers are used to declare a longer range of values and it occupies 4 bytes of storage space.

Syntax:

```
int variable_name;

  int num1;
  short int num2;
  long int num3;
```

example: 5, 6, 100, 2500
Integer Data Type Memory Allocation:

Short int	int	long int
1 byte	2 bytes	4 bytes

Floating Point Data Types:

The float data type is used to store fractional numbers (real numbers) with 6 digits of precision. Floating point numbers are denoted by the keyword float. When the accuracy of the floating point number is insufficient, we can use the double to define the number. The double is the same as float but with longer precision and takes double space (8 bytes) than float. To extend the precision further we can use a long double which occupies 10 bytes of memory space.

Syntax:

```
    float variable;
     float num1;
     double num2;
     long double num3;
```

Example: 9.125, 3.1254

Floating Point Data Type Memory Allocation:

float	double	long double
4 bytes	8 bytes	10 bytes

Character Data Type:

Character type variables can hold a single character and are declared by using the keyword char. As there are signed and unsigned int (either short or long), in the same way there are signed and unsigned chars; both occupy 1 byte each, but have different ranges. Unsigned characters have values between 0 and 255, signed characters have values from –128 to 127.

Syntax:

```
         char variable;
    char ch = ‘a’;
```

Example: a, b, g, S, j.

Void Type:

The void type has no values therefore we cannot declare it as variable as we did in case of integer and float. The void data type is usually used with function to specify its type.

Skill AP Learn Anytime Anywhere Andhra Pradesh State Skill Development Corporation