Introduction to C

Tushar B. Kute,
http://tusharkute.com

What is C Programming?

- The C Language is developed by Dennis Ritchie for creating system applications that directly interact with the hardware devices such as drivers, kernels, etc.
- C programming is considered as the base for other programming languages, that is why it is known as mother language.

What is C Programming?

- C language is considered as the mother language of all the modern programming languages because most of the compilers, JVMs, Kernels, etc. are written in C language, and most of the programming languages follow C syntax, for example, C++, Java, C#, etc.
- It provides the core concepts like the array, strings, functions, file handling, etc. that are being used in many languages like C++, Java, C#, etc.

What is C Programming?

- A system programming language is used to create system software.
- C language is a system programming language because it can be used to do low-level programming (for example driver and kernel).
- It is generally used to create hardware devices, OS, drivers, kernels, etc. For example, Linux kernel is written in C.
- It can't be used for internet programming like Java, .Net, PHP, etc.

What is C Programming?

- A procedure is known as a function, method, routine, subroutine, etc. A procedural language specifies a series of steps for the program to solve the problem.
- A procedural language breaks the program into functions, data structures, etc.
- C is a procedural language. In C, variables and function prototypes must be declared before being used.

What is C Programming?

- A structured programming language is a subset of the procedural language.
- Structure means to break a program into parts or blocks so that it may be easy to understand.
- In the C language, we break the program into parts using functions.
- It makes the program easier to understand and modify.

What is C Programming?

- C is considered as a middle-level language because it supports the feature of both low-level and high-level languages.
- C language program is converted into assembly code, it supports pointer arithmetic (low-level), but it is machine independent (a feature of high-level).

  A Low-level language is specific to one machine, i.e., machine dependent. It is machine dependent, fast to run. But it is not easy to understand.
- A High-Level language is not specific to one machine, i.e., machine independent. It is easy to understand.

First C Program

```
#include <stdio.h>
        int main( ) {
        printf("Hello World\n");
        return 0;
}
```

Printf and scanf

- The printf() function is used for output. It prints the given statement to the console.
- The syntax of printf() function is given below:

```
printf("format string",argument_list);
```

- The format string can be %d (integer), %c (character), %s (string), %f (float) etc.
- The scanf() function is used for input. It reads the input data from the console.

```
scanf("format string",argument_list);
```

Examples

- Program to print cube of given number.
- Program to print sum of 2 numbers

Variables

- A variable is a name of the memory location. It is used to store data. Its value can be changed, and it can be reused many times.
- It is a way to represent memory location through symbol so that it can be easily identified.
- Let's see the syntax to declare a variable:

```
type variable_list;
```

Variables

- The example of declaring the variable is given below:

```
int a;
float b;
char c;
```

- Here, a, b, c are variables. The int, float, char are the data types.

Variables

- We can also provide values while declaring the variables as given below:

```
int a=10,b=20;
//declaring 2 variable of integer type
float f=20.8;
char c='A';
```

Variables : Rules

- 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.

Variables : Rules

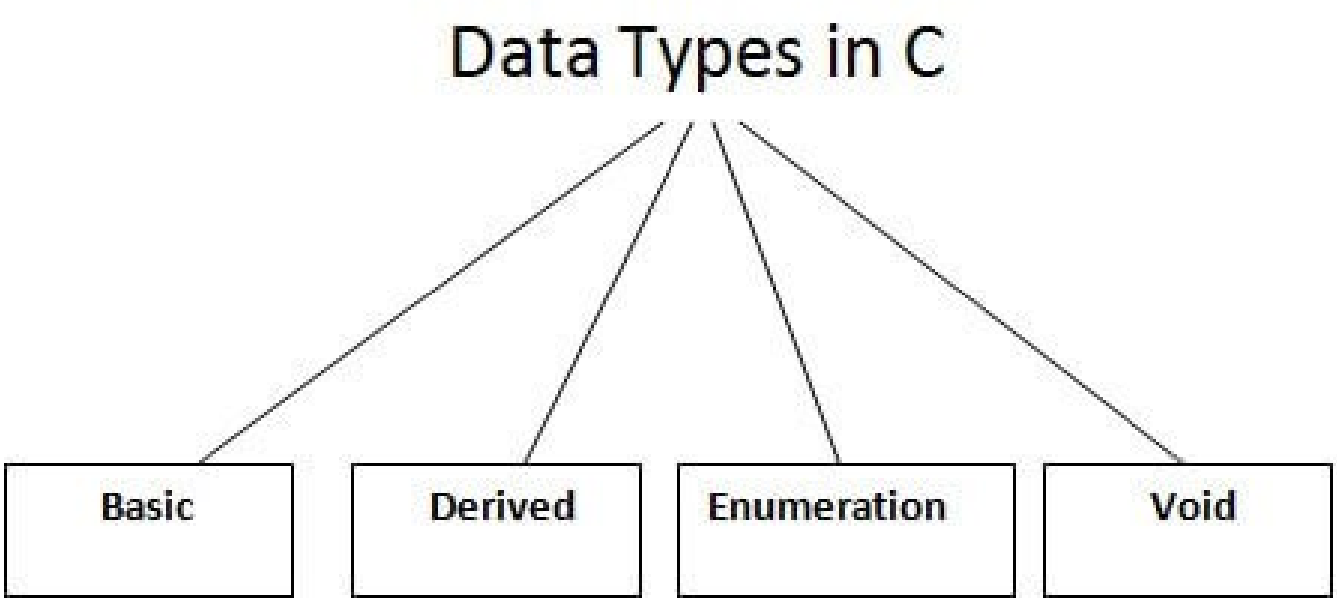
- Valid variable names:

```
int a;
int _ab;
int a30;
```

- Invalid variable names:

```
int 2;
int a b;
int long;
```

Data Types



Data Types

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

Data Types

Data Types	Memory Size	Range
char	1 byte	–128 to 127
signed char	1 byte	–128 to 127
unsigned char	1 byte	0 to 255
short	2 byte	–32,768 to 32,767
signed short	2 byte	–32,768 to 32,767
unsigned short	2 byte	0 to 65,535
int	2 byte	–32,768 to 32,767
signed int	2 byte	–32,768 to 32,767
unsigned int	2 byte	0 to 65,535

Data Types

short int	2 byte	–32,768 to 32,767
signed short int	2 byte	–32,768 to 32,767
unsigned short int	2 byte	0 to 65,535
long int	4 byte	-2,147,483,648 to 2,147,483,647
signed long int	4 byte	-2,147,483,648 to 2,147,483,647
unsigned long int	4 byte	0 to 4,294,967,295
float	4 byte	
double	8 byte	
long double	10 byte	

Keywords

- A keyword is a reserved word. You cannot use it as a variable name, constant name, etc. There are only 32 reserved words (keywords) in the C language.
- A list of 32 keywords in the c language is given below:

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

Operators

- An operator is simply a symbol that is used to perform operations. There can be many types of operations like arithmetic, logical, bitwise, etc.
- There are following types of operators to perform different types of operations in C language.

  Arithmetic Operators

  Relational Operators

  Shift Operators

  Logical Operators

  Bitwise Operators

  Ternary or Conditional Operators

  Assignment Operator

  Misc Operator

Operators

Category	Operator	Associativity
Postfix	() [] -> . ++ - -	Left to right
Unary	+ - ! ~ ++ - - (type)* & sizeof	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Shift	<< >>	Left to right
Relational	< <= > >=	Left to right
Equality	== !=	Left to right
Bitwise AND	&	Left to right
Bitwise XOR	^	Left to right
Bitwise OR	|	Left to right
Logical AND	&&	Left to right
Logical OR	||	Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %=>>= <<= &= ^= |=	Right to left
Comma	,	Left to right

Comments

- Comments in C language are used to provide information about lines of code.
- It is widely used for documenting code. There are 2 types of comments in the C language.

  Single Line Comments

  Multi-Line Comments

Comments

- Single line comments are represented by double slash \\.
- Let's see an example of a single line comment in C.

```
#include<stdio.h>
int main(){
   //printing information
   printf("Hello C");
return 0;
}
```

Comments

- Let's see an example of a multi-Line comment in C.

```
#include<stdio.h>
int main(){
   /*printing information
     Multi-Line Comment*/
   printf("Hello C");
return 0;
}
```

Format Specifiers

- %d or %i It is used to print the signed integer value where signed integer means that the variable can hold both positive and negative values.
- %u It is used to print the unsigned integer value where the unsigned integer means that the variable can hold only positive value.
- %o It is used to print the octal unsigned integer where octal integer value always starts with a 0 value.

Format Specifiers

- %x It is used to print the hexadecimal unsigned integer where the hexadecimal integer value always starts with a 0x value. In this, alphabetical characters are printed in small letters such as a, b, c, etc.
- %X It is used to print the hexadecimal unsigned integer, but %X prints the alphabetical characters in uppercase such as A, B, C, etc.
- %f It is used for printing the decimal floating-point values. By default, it prints the 6 values after '.'.

Format Specifiers

- %e/%E It is used for scientific notation. It is also known as Mantissa or Exponent.
- %g It is used to print the decimal floating-point values, and it uses the fixed precision, i.e., the value after the decimal in input would be exactly the same as the value in the output.
- %p It is used to print the address in a hexadecimal form.
- %c It is used to print the unsigned character.
- %s It is used to print the strings.
- %ld It is used to print the long-signed integer value.

Format Specifiers

```
int main()
{
  int b=6;
  int c=8;
  printf("Value of b is:%d", b);
  printf("\nValue of c is:%d",c);

    return 0;
}
```

If statement

- The if-else statement in C is used to perform the operations based on some specific condition.
- The operations specified in if block are executed if and only if the given condition is true.
- There are the following variants of if statement in C language.

  If statement

  If-else statement

  If else-if ladder

  Nested if

If statement

- The if statement is used to check some given condition and perform some operations depending upon the correctness of that condition.
- It is mostly used in the scenario where we need to perform the different operations for the different conditions. The syntax of the if statement is given below.

```
if(expression){
    //code to be executed
}
```

Example:

```
#include<stdio.h>
int main(){
    int number=0;
    printf("Enter a number:");
    scanf("%d",&number);
    if(number%2==0){
        printf("%d is even number",number);
    }
        return 0;
}
```

if-else

```
if(expression){
    //code to be executed if condition is true
}else{
    //code to be executed if condition is false
}
```

Loops

- While loop
- For loop
- Do-while loop
- Break
- Continue
- goto

While Loops

- While loop is also known as a pre-tested loop. In general, a while loop allows a part of the code to be executed multiple times depending upon a given boolean condition.
- It can be viewed as a repeating if statement. The while loop is mostly used in the case where the number of iterations is not known in advance.
- Syntax of while loop:
- The syntax of while loop in c language is given below:

```
while(condition){
    //code to be executed
}
```

Do-While Loop

- The do while loop is a post tested loop. Using the do-while loop, we can repeat the execution of several parts of the statements.
- The do-while loop is mainly used in the case where we need to execute the loop at least once. The do-while loop is mostly used in menu-driven programs where the termination condition depends upon the end user.
- The syntax of the C language do-while loop is given below:

```
do{
    //code to be executed
}while(condition);
```

for Loop

- The for loop in C language is used to iterate the statements or a part of the program several times.
- It is frequently used to traverse the data structures like the array and linked list.
- The syntax of for loop in c language is given below:

```
for(Expression 1; Expression 2; Expression 3)   {
    //code to be executed
}
```

break

- The break is a keyword in C which is used to bring the program control out of the loop.
- The break statement is used inside loops or switch statement.
- The break statement breaks the loop one by one, i.e., in the case of nested loops, it breaks the inner loop first and then proceeds to outer loops.

continue

- The continue statement in C language is used to bring the program control to the beginning of the loop.
- The continue statement skips some lines of code inside the loop and continues with the next iteration.
- It is mainly used for a condition so that we can skip some code for a particular condition.
- Syntax:

```
//loop statements
continue;
//some lines of the code which is to be skipped
```

continue

- The continue statement in C language is used to bring the program control to the beginning of the loop. The continue statement skips some lines of code inside the loop and continues with the next iteration. It is mainly used for a condition so that we can skip some code for a particular condition.
- Syntax:
-
- //loop statements
- continue;
- //some lines of the code which is to be skipped

Array

- An array is defined as the collection of similar type of data items stored at contiguous memory locations.
- Arrays are the derived data type in C programming language which can store the primitive type of data such as int, char, double, float, etc. It also has the capability to store the collection of derived data types, such as pointers, structure, etc.
- The array is the simplest data structure where each data element can be randomly accessed by using its index number.

Array

- We can declare an array in the c language in the following way.

```
data_type array_name[array_size];
```

- Now, let us see the example to declare the array.

```
int marks[5];
```

- Here, int is the data_type, marks are the array_name, and 5 is the array_size.

Thank you

@mitu_skillologies

@mITuSkillologies

@mitu_group

@mitu-skillologies

@MITUSkillologies

kaggle

@mituskillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

@mituskillologies

contact@mitu.co.in

tushar@tusharkute.com