**C Language**

**What is C Language?**

C is a programming language developed at AT & T’s Bell Laboratories of USA in 1972 by Dennis Ritchie.

Any programming Language can be divided in to two categories.

Problem oriented (High level language)

Machine oriented (Low level language)

But C is considered as a Middle level Language.

C is modular, portable, reusable

**Structured language**

It has the ability to divide and hide all the information and instruction.

Code can be partitioned in C using functions or code block.

C is a well-structured language compare to other.

**General purpose language**

Make it ideal language for system programming.

It can also be used for business and scientiﬁc application.

ANSI established a standard for c in 1983.

The ability of c is to manipulate bits, byte and addresses.

It is adopted in later 1990.

**Portability**

Portability is the ability to port or use the software written.

One computer C program can be reused.

By modiﬁcation or no modiﬁcation.

**Code Re-usability & Ability to customize and extend**

A programmer can easily create his own function

It can can be used repeatedly in different application

C program basically collection of function

The function are supported by 'c' library

Function can be added to 'c' library continuously

**Limited Number of Key Word**

There are only 32 keywords in 'C'

27 keywords are given by Ritchie

5 is added by ANSI

The strength of 'C' is lies in its in-built function

UNIX system provides as large number of C function

Some function are used in operation.

Other are for specialized in their application

**C program structure**

Pre-processor directives

Global declarations

Main ()

{

Local variable deceleration

Statement sequences

Function invoking

}

**C Keywords**

Keywords are the words whose meaning has already been explained to the C compiler. There are only 32 keywords available in C. The keywords are also called ‘Reserved words’.

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

**Rules for Writing, Compiling and Executing the C program**

C is case sensitive means variable named "COUNTER" is different from a variable named "counter".

All keywords are lowercased.

Keywords cannot be used for any other purpose (like variable names).

Every C statement must end with a; Thus; acts as a statement terminator.

First character must be an alphabet or underscore, no special symbol other than an underscore, no commas or blank spaces are allowed with in a variable, constant or keyword.

Blank spaces may be inserted between two words to improve the readability of the statement. However, no blank spaces are allowed within a variable, constant or keyword.

Variable must be declared before it is used in the program.

File should be have the extension .c

Program need to be compiled before execution

**Escape Sequences**

An escape sequence in C language is a sequence of characters that doesn't represent itself when used inside string literal or character. It is composed of two or more characters starting with backslash \. For example: \n represents new line.

\a - Bell

\n - New line

\r - Carriage return

\b - Backspace

\f - Formfeed

\t - Horizontal tab

\" - Quotation mark

\v - Vertical tab

\' - Apostrophe

\\ - Backslash

\? - Question mark

\0 - Null

**Place Holders**

Long int - %ld

**Data Types**

**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 bit processor, 2 byte (16 bit) of memory will be allocated for int data type.

Likewise, 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.

**CHARACTER DATA TYPE:**

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.

**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.

**C Data types / Storage Size Range**

Char / 1 –127 to 127

Int / 2 –32,767 to 32,767

ﬂoat / 4 1E–37 to 1E+37 with six digits of precision

Double / 8 1E–37 to 1E+37 with ten digits of precision

Long double / 10 1E–37 to 1E+37 with ten digits of precision

Long int / 4 –2,147,483,647 to 2,147,483,647

Short int / 2 –32,767 to 32,767

Unsigned short int / 2 0 to 65,535

Signed short int / 2 –32,767 to 32,767

Long long int / 8 – (2power (63) –1) to 2(power) 63 –1

Signed long int / 4 –2,147,483,647 to 2,147,483,647

Unsigned long int / 4 0 to 4,294,967,295

Unsigned long long int / 8 2(power) 64 –1

**Variables**

**Arithmetic Operators**

Arithmetic operators are used to perform mathematical calculations like addition, subtraction, multiplication, division and modulus in C programs

Arithmetic Operators Example

+ (Addition) A+B

– (Subtraction) A-B

\* (multiplication) A\*B

/ (Division) A/B

% (Modulus) A%B

**Relational Operators**

Relational operators are used to ﬁnd the relation between two variables. i.e. to compare the values of two variables in a C program

Operators Example/Description

> x > y (x is greater than y)

< x < y (x is less than y)

>= x >= y (x is greater than or equal to y)

<= x <= y (x is less than or equal to y)

== x == y (x is equal to y)

!= x! = y (x is not equal to y)

**Logical Operators**

These operators are used to perform logical operations on the given expressions. There are 3 logical operators in C language. They are, logical AND (&&), logical OR (||) and logical NOT (!).

Operators Example/Description

&& (logical AND) (x>5) &&(y<5) It returns true when both conditions are true

|| (logical OR) (x>=10) ||(y>=10) It returns true when at-least one of the condition is true

! (Logical NOT)

! ((x>5) &&(y<5)) It reverses the state of the operand “((x>5) && (y<5))” If “((x>5) && (y<5))” is true, logical NOT operator makes it false

**Unary Operators**

**Print and Input Statements**

Printf

Scanf

Getch()

Getche()

Getchar()

Gets()

**String Copy and Compare**

Strcpy()

Strcmp()

**Conditional Statements**

If()

If () else

If () else if

Condition Operator

A = 10 ? B = 20:B=30

**Jump Statements**

**Locate Statement**

Gotoxy(x,y)

Gotoxy sets the cursor position on the screen.

X: Column (X-Axis) 1-80

Y: Row (Y-Axis) 1-25

If Specified anything else then it will set the cursor on 1,1

e.g

gotoxy(20,12)

Sets the cursor at Row Number 12, Column Number 20

**LOOP Statement**

Repetition of statements (Process) is called Loop.

for(expr1;expr2;exp3)

{

Body of our program needs to be repeat.

}

Expr1 = Initial Value (Starting)

Expr2 = Ending Value (Expression)

Expr3 = Increment/Decrement (STEP)

e.g

int a;

for(a=1;a=10;a++)

{

Body of my program will repeat 10 times until value of a will be 10.

}

Assignment:

1. Draw a Box
2. Input a Name, and scroll the name on Row 12 from Column 1 to 80.
3. Input a Name and scroll it horizontal and vertically in all the screen

While (expression)

{

Body of our program needs to be repeat.

}

Do while ()

Do

{

Body of our program needs to be repeat.

} while(expression)

DO – While

Body must be executed one time in any case.

Assignment:

Write a program of CALCULATOR of Add, Subtract, Multiply and Divide

Program will take two Inputs.

1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit

Assignment:

Write a program to generate Multiple Tables.

* Input Start Table Number: 2
* Input Ending Table Number: 10
* Input Multiple: 12

After generation of each Table, Pause the Screen with the message “Press any key to continue”

It will generate tables from 2 to 10 with the maximum of 12 Multiples.

**Case Control Statement**

1. Switch (Same like Nested IF)
2. Break (Loop, Swtich)
3. Continue (Jump to loop)
4. Goto (Jump to any Label☺

Switch Example:

Switch(myvar)

{

Case <Condition>:

‘Body of True Condition

Case <Condition>:

Body of True Condition

Default:

Body of False Condition if all of the above Case condition are false.

}

**CONST Qualifier**

It is used to declare a variable whose value cannot be change.

Can be used to declare a variable with FIXED Value.

Example: Value of PI = 3.142

const float pi=3.143;

**Default Value**

int a;

float b;

char c;

|  |  |  |
| --- | --- | --- |
| Int a (2 Bytes) | Float b (4 Bytes) | Char c (1 Byte) |
| 010101001010 |  | 010101010 |
|  |
|  |  |
|  |

In Memory, Computers stores the values as BINARY CODE. Best practice is to define the default value at the time of variable declaration.

Int a =0;

Float b = 0;

Char c = 0;

**ARRAY**

Array is a collection of items (data types like int, float, char etc.)

Two types of Array.

1. Single Dimension (Only Rows)
2. Multi Dimension (Rows and Columns)

**Single Dimension**

Character Array.

Char ch[10];

|  |  |
| --- | --- |
| Index Position | Ch Array Element Values |
| 0 |  |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |

Assignment:

Create Character Array variable of size 25.

1. Write a program to get input of Character Array Variable, and Calculate Length of the String.
2. Write a program to get input of character array variable and another input of integer input of number of character, then print Left of n Characters from the input.
3. Same as Program 2, Print Right of n Characters.
4. Write a program which will take character array input, starting position, number of characters, then print number of n characters between start and end position.

LEN, LEFT, RIGHT, MID

Assignment:

1. String Reverse

Nabeel -- > leeban

1. Change to Upper Case  
   nabeel (NABEEL)
2. Change to Lower Case  
   NABEEL (nabeel)

**Numeric ARRAY**

* Int
* float
* double

int data type has two bytes

example

int a=10;

C Compiller will allocate 2 bytes (16 bits) for integer variable a in memory.

|  |  |
| --- | --- |
| Byte 1 | Byte 2 |
| 10 (00000000 00001010) | |

char name[20];

int a[5]={1,3,5,7,9;

create an integer array of 5 elements. Each Integer has 2 Bytes, means 5 elements of integer will take 2 x 5 = 10 Bytes

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Element 0 | | 1 | | 2 | | 3 | | 4 | |
| Value 1 | | 3 | | 5 | | 7 | | 9 | |
| Byte 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

Assignement:

Take 10 elements of Integer Array and find out the following:

1. Average
2. Minimum Value
3. Maximum Value

Array Sort Logic – Bubble Sort

Bubble sort in C to arrange numbers in ascending order, you can modify it for descending order and can also sort strings. The bubble sort algorithm isn't efficient as its average-case complexity is O(n2) and worst-case complexity is O(n2). There are many fast sorting algorithms like Quicksort, heap-sort, and others. Sorting simplifies problem-solving in computer programming.

* Compare 1st Element Value to 2nd Element Value. If 1st Element is greater than 2nd element then SWAP Element 1 with Element 2.

#include <stdio.h>

#include <conio.h>

void main(void)

{

int num[10];

int a,b,c;

int swap;

clrscr();

for(a=0;a<=9;a++)

{

printf("enter number %d : ",a+1);

scanf("%d",&num[a]);

}

for(a=0;a<=8;a++)

{

for(b=0;b<=8-a;b++)

{

if(num[b] > num[b+1])

{

swap = num[b];

num[b] = num[b+1];

num[b+1]=swap;

}

}

}

for(a=0;a<=9;a++)

{

printf("\n%d",num[a]);

}

getch();

}

Bubble Sort Algorithm for 10 elements

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 30 |  | loop master | Child loop |  |  |  |  |  |  |  |  | Processing |
| 20 |  | 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 25 |  | 1 | 0 |  |  |  |  |  |  | 7 |  | 8 |
| 30 |  | 2 | 0 |  |  |  |  |  | 6 |  |  | 7 |
| 35 |  | 3 | 0 |  |  |  |  | 5 |  |  |  | 6 |
| 50 |  | 4 | 0 |  |  |  | 4 |  |  |  |  | 5 |
| 65 |  | 5 | 0 |  |  | 3 |  |  |  |  |  | 4 |
| 85 |  | 6 | 0 |  | 2 |  |  |  |  |  |  | 3 |
| 90 |  | 7 | 0 | 1 |  |  |  |  |  |  |  | 2 |
| 98 |  | 8 | 0 |  |  |  |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  |  |  |  |  | 45 |

Program to Sort Array element in Descending Order

1. Program to Copy Array into another Array. Print Both Arrays
2. Program to Move Array Values into another Array. Print Both Arrays
3. Find single Value from the Array
   1. Program will take 10 numbers input into an array.
   2. Input number to find
   3. Find the given input number and print the number and element index (Position)
4. Find All Values from the Array. Same as Program 3 but it will find and print all matching.

All Program for Character Array.

1. Program to take 10 Characters and print.
2. Program to take 10 characters and Sort
3. Program to take 10 characters and Sort Descending
4. Program to Copy Array
5. Program to Move Array
6. Program to Find singe Character in Array
7. Program to Find Multiple Character Values in Array

Character Array[] = One STRING

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | O | H | A | I | L |  |  |  |  |

N$=”Sohail” (String in GWBasic, but in Computer it is Character Array N[6])

In Access S hort Text, means String (but in computer it is Character Array StudentName[25])

**TEXT / BACKGROUND COLOR (FOREGROUND/BACKGROUND COLOR)**

0-15 (Color Number)

0-7 (Dark Color)

8-15 – Same as 0-7 Colors but in Sharpness.

textcolor(colornumber);

textbackground(colornumber);

cprintf(“”);

0=BLACK, /\* dark colors \*/

1-BLUE,

2-GREEN,

3-CYAN,

4-RED,

5-MAGENTA,

6-BROWN,

7-LIGHTGRAY,

8-DARKGRAY, /\* light colors \*/

9-LIGHTBLUE,

10-LIGHTGREEN,

11-LIGHTCYAN,

12-LIGHTRED,

13-LIGHTMAGENTA,

14-YELLOW,

15-WHITE

;

128-BLINK

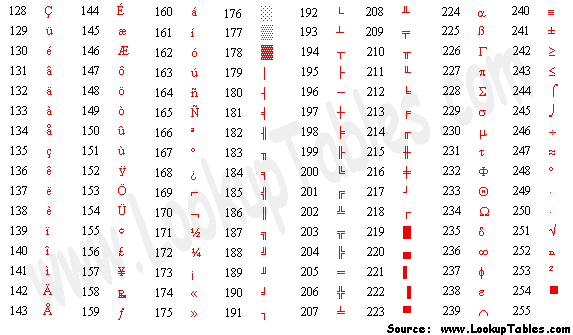
For Blinking

Add 128 with any color number.

Textcolor(2+128);

Green with blinking

**ASCII Code Table**

**ASCII Special Codes**

Keyboard layout contains Two Types of Keys:

1. ASCII Keys:   
   ASCII key has ONE BYTE Storage. (Alphabets (Small/Caps), Numerical Numbers, Special Characters like (+,-$# etc), Tab Key, Back Space,
2. Extended Keys:  
   Extended Keys has TWO Bytes Storage.  
   4 Arrow Keys(Left, Down, Up, Arrow), Insert, Home, Page UP, Page Down, Delete, End, Page Up, Function Keys(F1-F12) ALT+Keys,

User Types ‘A’ – Keyboard will send one Byte to Computer (INPUT) because ‘A’ is one Byte character.

What our program will do?

Getch() will receive one byte input from Keyboard buffer, and will store into variable CH. After that PRINTF() will print the Character ‘A’ and it’s ASCII Code in a Decimal (Integer). Our program will compare the CH Value with 27, which is False, in this case loop will start again.

User Hit ‘RIGHT ARROW’ – Keyboard will send Two Bytes to computer (INPUT) because ‘RIGHT ARROW’ is an extended Key and it has 2 bytes of storage.

What our program will do?

GETCH() will receive one byte from the Keyboard Buffer out of Two Bytes. Means ONE BYTE still exists in Keyboard Buffer.

After that PRINTF() will print the First Byte what has been received by GETCH. In case of Extended Keys 1st Byte contains NULL character which has an ASCII Code 0.

Our program will compare the CH value with Code 27, which is false, in this case loop will start again.

Again GETCH() will receive one byte from the Keyboard but this time Keyboard buffer already has One Byte so GETCH() will not wait for the user to provide INPUT, it will automatically receive ONE BYTE from the Keyboard Buffer and will continue.

Up-0,72

Down-0,80

Right-0,77

Left-0,75

Assignment:

Design a Menu of Calculator. Use Box, Shadow and Colors.

Calculator Menu

ALT-A Addition

ALT-S Subtraction

ALT-M Multiplication

ALM-D Division

End Exit

**FUNCTIONS**

Function is a ready-made / predefined program. We can re-use it any time whenever required.

In C We do have two types of Functions

1. C Library Functions (Built in Functions available in C Language by Turbo C)
   1. String Functions
   2. Math Functions
   3. Memory Functions
   4. Files Operation Functions
   5. Etc.
2. User Defined Functions (These functions designed by the user).

Each Function has four different types

1. Function which will not take any parameter and will not return any value. For Example: clrscr();
2. Function which will take parameter, but will not return any value. For Example: SCANF(), PRINTF(“SOHAIL”), GETS(myvar)
3. Function which will not take any parameter, but can return value. For Example: GETCH(), GETCHE(), GETCHAR()
4. Function which will take parameter, and return value. L = LEN(“SOHAIL”), STRCMP()

Function Declaration

<Return Value Type> FUNCTION NAME ([<Parameter List>])

1. Function with No Return value and No Parameter
   1. void FunctionName(void)
2. Function with No Return Value but with Parameter
   1. Void FunctionName(int a)
   2. Void FunctionName(char a);
   3. Void FunctionName(int a[10]);
   4. Void FunctionName (char ch[20]);
   5. Void FunctionName(int a, int b)
   6. Void FunctionName(int a, char b, int c[10], char d[20])
3. Function with No parameter but has return value
   1. Int FunctionName() This function will return integer value
   2. Char FunctionName() Return Single Character
   3. Int[10] FunctionName() Return integer element with 10 elements
   4. Char[10] FunctionName() Return Character Array with 10 Elements
4. Function with Parameter and Return Value