Cheat sheet

What is C?

C is a procedural and general-purpose programming language that combines the features of both high-level and low-level languages, resulting in faster execution and smoother application development. Hence, C earned its patent as the first middle-level programming language.
C was born in 1972 at Bell Laboratories. A remarkably simple and highly readable programming language resulted in groundbreaking advancements in the IT industry.

Syntax

Every C program execution begins from the main function

```
#include <stdio.h>

int main() // Every C program execution begins
{
   printf("Hello World");

   return 0;
}
```

Data types

Data type	Size in bytes	Gormat specifier
char	1	%c
signed char	1	%c
unsigned char	1	%c
short int or int	2	%hd, %d
unsigned int	2	%u
long int	4	%ld
unsigned long int	4	%lu
Float	4	%f
Double	8	%lf
long double	10	%Lf

Variables

Syntax

<data type> <variable-name = value>;

Valid variable names	Invalid variable names
int sum;	int 30a;
float _ab;	float x y;
int b20;	int goto;

Constant

Syntax

Const <data type> variable-name = value;

value is constant throughout the program run-time. Constant values are unchangeable.

Operators

Unary Operators

Unary Operators	Description	Example (x=5, y=2)
-	Unary minus	y=-y
++	Increment	x=++y / x=y++
--	Decrement	x=--y/x= y--
!	Logical not	!(x<=y)

Binary operator

Arithmetic Operators	Description	Example (x=5, y=2)
+	Addition	x + y
-	Subtraction	x – y
*	Multiplication	x * y
/	Division	x / y
%	Modular	x % y

Relational Operators	Description	Example (x=5, y=2)
>	Greater than	x > y
<	Less than	x < y
<=	Less than or equal to	x <= y
>=	Greater than or equal to	x >= y
==	Equal to	x == y
!=	Not equal to	x !=y

Logical Operators	Description	Example (x=5, y=2, z=3)
&&	Logical AND	x > y && x < z
||	Logical OR	x < y || x > z
!	Logical NOT	!(x < y)

Bitwise Operators	Description	Example (x=5, y=2)
&	AND	x & y
|	OR	x | y
^	XOR	x ^ y
~	Complement	~ x
>>	Right shift	x >> 1
<<	Left shift	x << 1

Decision-making statements

Simple if statement

```
if (condition)
{
  Statement1;
}
```

If-else statement

```
if (condition)

  Statement1;

else

Statement 2;
```

Else-if ladder statements

```
if (condition1)
{
   statement1;
}
else if (condition2)
{
  statement2;
}
else if(condition3)
{
  statement3;
}
—----------------
—----------------
else if (condition n-1)
{
   statement n-1;
}
else
{
  statement n;
}
```

Loop Control Statements in C

For loop

Syntax:

```
for (expression1; expression2; expression3)
{
   statement 1;
   statement 2;
   —-------------
   —-------------
   statement n;
}
```

While loop

Syntax:

```
while (condition)
{
   statement 1;
   statement 2;
   —-------------
   —-------------
   statement n;
}
```

Do-while loop

Syntax:

```
do
{
   statement 1;
   statement 2;
   —-------------
   —-------------
}
while (condition);
Statement n;
```

Arrays

Syntax

```
datatype array_name[array_size];

int i, a[5]={10,20,30,40,50};
for(i=0;i<n;i++)
{
    printf("%d",a[i]);
}
```

Strings

Syntax

Ways to initialize a string in c is:

```
char str[6]={'h','e','l','l','o','\0'};
char str[]={'h','e','l','l','o','\0'};
char str[]="hello world";
char str[5]="name";
```

Full Stack Developer
With a Job Guarantee

CLICK THE LINK TO ENROLL NOW

https://bit.ly/3OpPEHL

simplilearn

Pointers

Syntax

```
<datatype> *pointer_variable;

int *p;
int *arr[15];
char *a;
```

Functions

A function or method is a block of statement which is invoked when a function is called.

```
void myFunction()
{
printf("Executed");
}

int main()
{
  myFunction(); // call the function
  return 0;
}
```

Structures

Syntax

```
struct structure_name
{
  int num;
  char gender;
};
```

File Handling

The declaration is

FILE *prt1,*ptr2; (FILE asterisk and followed by filepointer_name)

Palindrome using Recursion function

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>

bool Rec_palindrome(char str[], int f, int l)
{

    if (f == l)
    return true;


    if (str[f] != str[l])
    return false;


    if (f < l + 1)
    return Rec_palindrome(str, f + 1, l - 1);

    return true;
}

bool palindrome(char str[])
{
int n = strlen(str);

if (n == 0)
return true;

return Rec_palindrome(str, 0, n - 1);
}

int main()
{
    char str[] = "malayalam";

    if (palindrome(str))
    printf("The given string is palindrome");
    else
    printf("The given string is not palin-
drome");

    return 0;
}
```

Pyramid star pattern program

```
#include <stdio.h>
int main()
 {
       int i, j, rows = 5;
         for (int i = 0; i < rows; i++) {

        for (int j = rows - i; j > 1; j--) { //Loop for blank
space
            printf(" "); //Print Space
          }

          for (int j = 0; j <= i; j++) { //loop for star
              printf("* "); //prints * and blank space
          }
          printf("\n"); //New line
 }
}
```