* Constants:
→ Fined entity whose value doesnot change during program execution.

↳ Can also be defined as the value i.e. to be stored in a particular memory location. For eg: $x = 5$:
Here 5 is a constant

↳ There are basically 2 types of constants.
① Primary Constant
ⓐ Integer Constant:
↳ These are the whole numbers without any decimal part
↳ Rules for constructing integer constant:
① Must have at least one digit
② Must not have a decimal point
③ Can be either +ive or negative.
④ The default sign is considered to be +ive
⑤ No commas & blank spaces are considered.
⑥ Range for integer is ~~-326~~ -32767 to 32767
Eg: 2, 1346, etc.

ⓑ Real constant:
→ These are the constants that are used to represent the continuously changing values such as wt., temperature & so on.
↳ Real constants can be expressed in either fractional form or exponential form.
↳ Range $-3.4e^{38}$ to $3.4e38$

* Rules for constructing real constants, in fractional form:

→ Must have at least one digit & decimal point.
→ Can be either negative or positive.
↳ +ive is default sign.
↳ No commas or blankspaces are allowed.
Eg: 3.14, ~~-6.67E27~~ -6.67 etc.

Rules for constructing Real constants ~~into~~ in Real form.
↳ Represented as mantissa e exponent or mantissa E exponent
- Inhere, mantissa is a part before exponent & exponent is a power
- Both mantissa & exponent must have at least one digit. They can be either positive or negative. But the default sign is considered to be positive.
Eg. ~~-6~~ 6.67E-11

↳ No commas or blankspace are allowed.

③ Character constants:-

→ Are the constants that represents alphabets, digits & special symbols.

* Rules for constructing character constants:

① Represented inside the single quotation

② Range of character constant is zero to 255.

Eg: 'a', 'M', '#', '1' etc.

(2) Rules Secondary Constants

Rules:

These are the constants that are derived from primary constants it includes. Array, string, functions, structure, union, pointer & so on

* Keywords
→ Words that have been already defined in C-compiler.
↳ So, they are also known as predefined or inbuilt or reserved words.

↳ There are total 32 keywords in C. That are

① int ② float ③ char ④ switch ⑤ for ⑥ while.

* Identifiers:
→ Unique names given to the various entities, in C such as variables, Arrays, functions, structure, pointer, union & so on

↳ They are given by the programmer while writing the program code.
Eg. int a; → Here a is a variable whose identity is its name i.e 'a'. which is also known as identifier.

* Rules for naming identifier:
- Must be contextual & meaningfull.
- Must start with either underscore or alphabet.
- No special symbols can be used within identifiers.
- No commas or whitespaces are alloweed.
- Keywords cannot be used as an identifier.

* Tokens in C:-
→ Also known as building blocks of C as they consists of the small units that are required to develop a fully executable program code.
- They include
  1. keywords
  2. Identifiers
  3. Constants
  4. strings:
     - They are the collection of characters & they are represented by double quotation Eg: "Hello world"

5. Operators:-
   mathematical * symbols that perform different operations & represent comparison along computation.

⑥ ⑥ Special Symbols:

→ All the symbols except digits, alphabets & blank spaces are considered as special symbol.

Eg: @ , #.etc

* Variables:

→ Entities whose value can be changed during program execution.

↳ Can also be defined as the memory location in which constants are stored.

Eg int a;

Here a is a variable in which integer constant 5 is stored.

Rules for naming variables:-

→ Names must be contextual & meaningfull.

↳ Must begin with either alphabet or underscore

Variable name	validity
-num	valid ✓
num	valid ✓
1_num	invalid ✗
num1	valid
&	invalid

↳ No special ~~names~~ symbols can be used within a name of variable.

↳ Keywords cannot be used as variable name.

↳ C is highly case sensitive language
Eg. name & Name are referred as differend variables.

↳ No commas & blank are allowed.
Eg: num,1 is invalid
but num1 is valid

→ Max length of variable names can ~~be not be more~~ 32 characters.

↳ Variable must be declared before using it.

Variable declaration:-

↳ Process of allocating a memory for storing a particular type of ~~fune~~ constant.

↳ Also known as memory location.

Syntax.

```
Data type varname ;
```

Type of data

name
→ user choice.

Data Types in C.

→ It indicates the type & nature of data to be stored in particular memory location.

Basically there are 2 categories of data types.

① Primary Data Types

↳ Includes:

ⓐ Integer data type

↳ It indicates the whole numbers without any decimal part

↳ keyword is 'int'.

↳ Format specifier: '%d'.

↳ It occupies two to four bytes of memory which is compiler dependent

↳ Range:- (-32767 to 32767).

Eg int num; , int n = 3;

ⓑ Float Data Type:

↳ Indicates numbers with decimal.

↳ keyword: 'float'

↳ Format specifier: '%f'

↳ Memory occupy: four bytes

↳ Range: ($-3.4e^{-38}$ to $3.4e38$)

↳ It gives sim digits of precision.

Eg: float PI = 3.14; float salary;

ⓒ Double

↳ Also represents decimal numbers but this data type gives 14 digits of precision.

keyword: 'double'

format specifier: '%lf'

memory occupy: 8 bytes

↳ Range: (-1.7e308 to 1.7e308)
Eg: float r = 0.00093721;

↳ When the range of double is not sufficient then we use long double data type. which occupies 10 bytes of memory & gives 18 digit of precesion.
↳ formad specifier: '%Lf'

③ character
↳ Indicates alphabets, digits, & special symbols
↳ Keyword: 'char'
↳ Format specifier: '%c'
↳ Memory occupy:- Only one byte.
Eg. char d = 'c';

④ Void
↳ Data type that doesnot hold any value.
↳ Specially used for functions that doesnot return any value.
↳ keyword: 'void'.

② Secondary Data Types
→ Also known as derived data types as are derived from primary data types.
→ It includes: Array, function, pointer, structure, union & so on.

* C program to i/p two double numbers & display their sum.

```
/*
 * program to take two numbers & display their
   sum.
 * author :
 *  07/01/2079
 */
 #include<stdio.h>
 #include<conio.h>

    Void main() {  double x, y, sum;
    printf(" Enter one number:");
    scanf("%lf", &x);

    printf (" Enter second number:");
    scanf("%lf", &y);

    sum = x+y;
    printf(" The sum of %lf & %lf is %lf", x, y, sum);

    getch();
    }
```

* Simple I/O functions:

→ There are basically two functions for I/P & O/P in C. They are:

① printf()

→ It is a simple O/P function.

Syntax:
printf ("string / format specifier/escape sequencer");

Eg:
printf("value of a = %d. \n value of b = %d", a, b);

② scanf()
↳ Simple i/p function that takes the i/p from the user.
Syntax:-
scanf("format specifiers", &var1, &var2, --);

Eg: scanf ("%d %d", &pi, &num2);

Inlays of variable ~~declaration~~:- Initialization.

Variable initialization is the process of storing a value in a particular location.
It can be done in two ways in C:

① Compile time initialization:-
↳ If the value of the variable is given by the programmer while writing the program it is known as compile time initialization

Q. Program to initialize & number & display it.

```
#include <stdio.h>
void main() {
    int a = 5;    // compile time initialization
```

```
printf("value of a is %d", a);
getch();
}
```

② Run time initialization:.

→ If the value of a variable is given by the user during execution or runtime it is known as run-time initialization

Q. program to i/p a number & display it using runtime initialization.

```
#include <stdio.h>
#include <conio.h>
void main(){
int n;
printf("Enter a number: ");
scanf("%d",&n);

printf("value you entered is %d", n);
getch();
}
```

Q WAP to input the time in seconds & convert it into seconds, minutes & hours ?

```
#include<stdio.h>
#include<conio.h>
void main() {
```

```
int second, minute, hour, remsecond;

printf("Enter seconds: ");
scanf("%d", &second);

hour = (second / 3600) %10;
remsecond = second % 3600;

minute = remsecond / 60;
second = remsecond % 60;

printf("Hours = %d, In It minutes = %d,
        In It seconds = %d", hour, minute, second);

getch();
}
```