Constants or Literals and Variables

A constant is a value or an identifier whose value cannot be altered in a program.

For example: 1, 2.5, "C programming is easy", ‘apple’ etc.

We can define constants in a C program in the following ways.

1. **By “const” keyword**
2. **By “#define” preprocessor directive**

Syntax1: **const type constant_name**;
Eg 1: **const double PI = 3.14** //variable *PI* is a constant ,3.14 cannot be changed

Eg-1

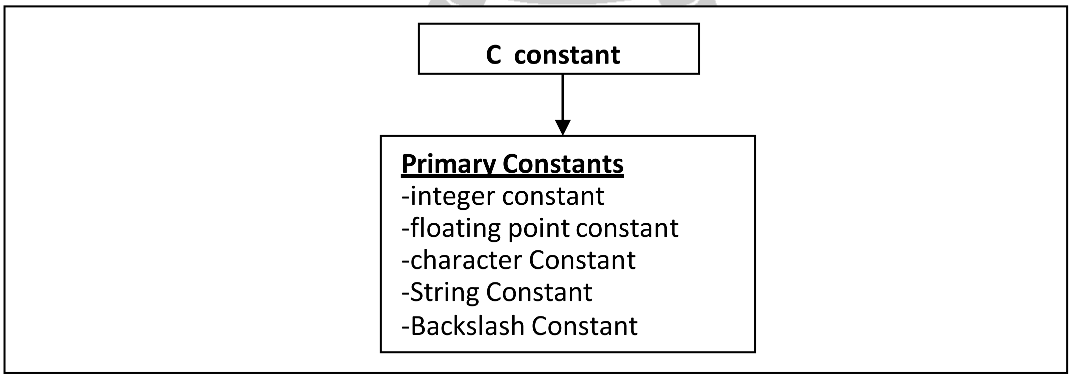
```
#include<stdio.h>
main()
{
  const int SIDE = 10;
  int area;
  area = SIDE*SIDE;
  printf("The area of the square %d is: %d sq. units" , SIDE, area);
}
```

Output

```
The area of the square 10 is: 100 sq. Units
```

Syntax 2: #define variable value
Eg-2 #define PI 3.14

Constants can be classified into broad categories



1. Integer constants

An integer constant is a numeric constant (associated with number) without any fractional or exponential part. There are three types of integer constants in C programming:

- **decimal constant(base 10)**
- **octal constant(base 8)**
- **hexadecimal constant(base 16)**

For example:

Decimal constants: 1,0, -9, 22 etc
Octal constants: 021, 077, 033 etc
Hexadecimal constants: 0x7f, 0x2a, 0x521 etc

In C programming, octal constant starts with a 0 and hexadecimal constant starts with a 0x.

55 /*int constant */
55l /*unsigned int constant*/
55 ul /*unsigned long constant*/

Rules for defining integer constants:

- An integer constant must have at least one digit.
- It must not have a decimal point.
- It can either be positive or negative.
- No commas or blanks are allowed within an integer constant.
- If no sign precedes an integer constant, it is assumed to be positive.
- The allowable range for integer constants is -32768 to 32767.

2. Floating-point constants

A floating point constant is a numeric constant that has either a fractional form or an exponent form(decimal point). For example:

-2.0
0.0000234
-0.22E-5

6.333 –correct

633E-4L-correct

633E---illegal..incomplete exponent
633f—illegal..no decimal or exponent
.e633—illegal..missing integer

Rules for defining floating point(real) constants:

- A real constant must have at least one digit
- It must have a decimal point
- The mantissa part and exponential part should be separated by a letter e/E
- The mantissa part must have a positive or negative sign. The default sign of mantissa part is positive.
- No commas or blanks are allowed within a real constant.

3. Character constants

A character constant is a constant which uses single quotation around characters.

For example:

'a'
'6',
'=',
'F'

Rules for defining character constants

- A character constant is a single alphabet, a single digit or a single special symbol enclosed within single quotes.
- The maximum length of a character constant is 1 character.

4. String constants

String constants are the constants which are enclosed in a pair of double-quote marks. For example:

```
"good"          //string constant
""              //null string constant
"      "        //string constant of six white space
"x"             //string constant having single character.
```

String constants are enclosed within double quotes.

5. Backslash Character Constants in C:

- There are some characters which have special meaning in C language.
- They should be preceded by backslash symbol.(\)

Backslash character	Meaning
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Horizontal tab
\”	Double quote
\’	Single quote
\\	Backslash
\v	Vertical tab
\a	Alert or bell
\?	Question mark
\N	Octal constant (N is an octal constant)
\XN	Hexadecimal constant (N – hex.dcml cnst)

Example Program Using Const Keyword

```
#include<stdio.h>
int main( )
{
int const BASE ,HEIGHT;
float area;
char NEWLINE=’\n’;
area=0.5*BASE*HEIGHT
printf(“The Area of Triangle is:”);
printf(“%c”,NEWLINE);
printf(“%f”,area);
return 0;
}
```

```
OUTPUT
10 20
The Area of Triangle is:

100
```

Using # Define

```
#include<stdio.h>
#define BASE 100
#define HEIGHT 100
#define NEWLINE ‘\n’
int main( )
{
float area;
area=0.5*BASE*HEIGHT
printf(“The Area of Triangle is:”);
printf(“%c”,NEWLINE);
printf(“%f”,area);
return 0;
```

```
OUTPUT
10 20
The Area of Triangle is:

100
```

Example program using const keyword in C:

```
#include <stdio.h>
void main()
{
const int height = 100; /*int constant*/
const float number = 3.14; /*Real constant*/
const char letter = 'A'; /*char constant*/
const char letter_sequence[10] = "ABC"; /*string constant*/
const char backslash_char = '\?'; /*special char cnst*/
printf("value of height :%d \n", height );
printf("value of number : %f \n", number );
printf("value of letter : %c \n", letter );
printf("value of letter_sequence : %s \n", letter_sequence);
printf("value of backslash_char : %c \n", backslash_char);
}
```

Output:

```
value of height : 100
value of number : 3.140000
value of letter : A
value of letter_sequence : ABC
value of backslash_char : ?
```

2. Example program using #define preprocessor directive in C:

```
#include <stdio.h>
#define height 100
#define number 3.14
#define letter 'A'
#define letter_sequence "ABC"
#define backslash_char '\?'
void main()
{
printf("value of height : %d \n", height );
printf("value of number : %f \n", number );
printf("value of letter : %c \n", letter );
printf("value of letter_sequence : %s \n",letter_sequence);
printf("value of backslash_char : %c \n",backslash_char);
}
```

Output:

```
value of height : 100
value of number : 3.140000
value of letter : A
value of letter_sequence : ABC
value of backslash_char : ?
```

Difference between variable and constants

The **difference between variables** and **constants** is that **variables** can change their value at any time but **constants** can never change their value.

Variable

```
int a =10;
a++;
printf("%d",a);
-----------------------
o/p:= 11
```

Constant variable

```
const int a =10;
a++;
printf("%d",a);
---------------------
o/p:= 10
```

VARIABLES

Variable is the name of memory location which holds the data. Unlike constant, variables are changeable, value of a variable can be changed during execution of a program. A programmer must chose a meaningful variable name.

Variables are used for holding data values so that they can be utilized for various computations in a program.A variable must be declaed and then used for coputation work in program./A variable is an identifier used for storing and holding some data(value).

All variables have three important attributes.

1.A ***data type: Like*** int, double, float. Once defined,the type of a C variable cannot be changed.
2.A ***name*** of the variable.
3.A ***value*** that can be changed by assigning a new value to the variable. The kind of values a variable can assume depends on its type.
Eg : for variable int salary,it can only take integer values can only take integer values like 65000 and not 6500.0

Rules For Constructing Variables

1. A variable name can be a combination of alphabets, numbers and special character underscore(_).
2. The first character in the variable name must be an alphabet.
3. No commas or blank spaces are available are allowed within a variable name.
4. No special symbol other than an underscore is allowed.
5.Upper and Lower case names are treated as different, as C is case sensitive, so it is suggested to keep the variable names in lower case.

Declaring and Initializing a variable:=

→Declaration of a variable must be done before it is used for any computation in the program.
→Declaration tells the compiler what the variable name is.
→ Declaration tells what type of data the variable will hold.
→Until the variable is not defined/or/declared compiler will not allocate memory space to the variables.
→ A variable can also be declared outside main() function.
→A variable can also be declared in other program and declared using extern keyword.

```
int yearly_salary;
float monthly_salary;
int a;
double x;
int ECE1111;
```

Initializing a variable:=
Initializing a variable means to provide a value to variable

```
int yearly salary=5,00,000
float monthly_salary= 41666.66
```

Difference between identifier and variable

Identifier	**Variable**
Indentifier is the name given to a variable,function etc.	While variable is used to name a memory location which stores data
An identifier can be a variable ,but not all identifiers are variables	All variables names are identifiers

Example : void average()
{
}
	Example: int average

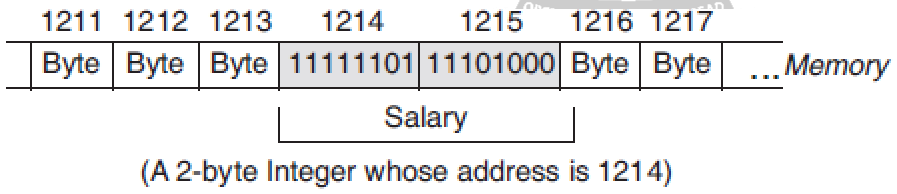
Variables are a way of reserving memory to hold some data and assign names to them so thatwe don't have to remember the numbers like REG46735 or memory address like FFFFoxFF and instead we can use the memory location by simply referring to the variable.

Every variable is mapped to a unique

memory address.And variable will be

having a data type associated

```
int salary = 65000;
```



[[[[[note A computer memory is made up of registers and cells. It accesses data in a collection of bits, typically 8 bits, 16 bit, 32 bit or 64 bit. A computer memory holds information inthe form of binary digits 0 and 1 (bits).**]]]]]**