Parul® University

Unit 1

303105104 - Computational Thinking for Structured Design-1

I Semester

C basics

- C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system.
- C is the most widely used computer language.
- It keeps fluctuating at number one scale of popularity along with Java programming language, which is also equally popular and most widely used among modern software programmers.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Facts about C

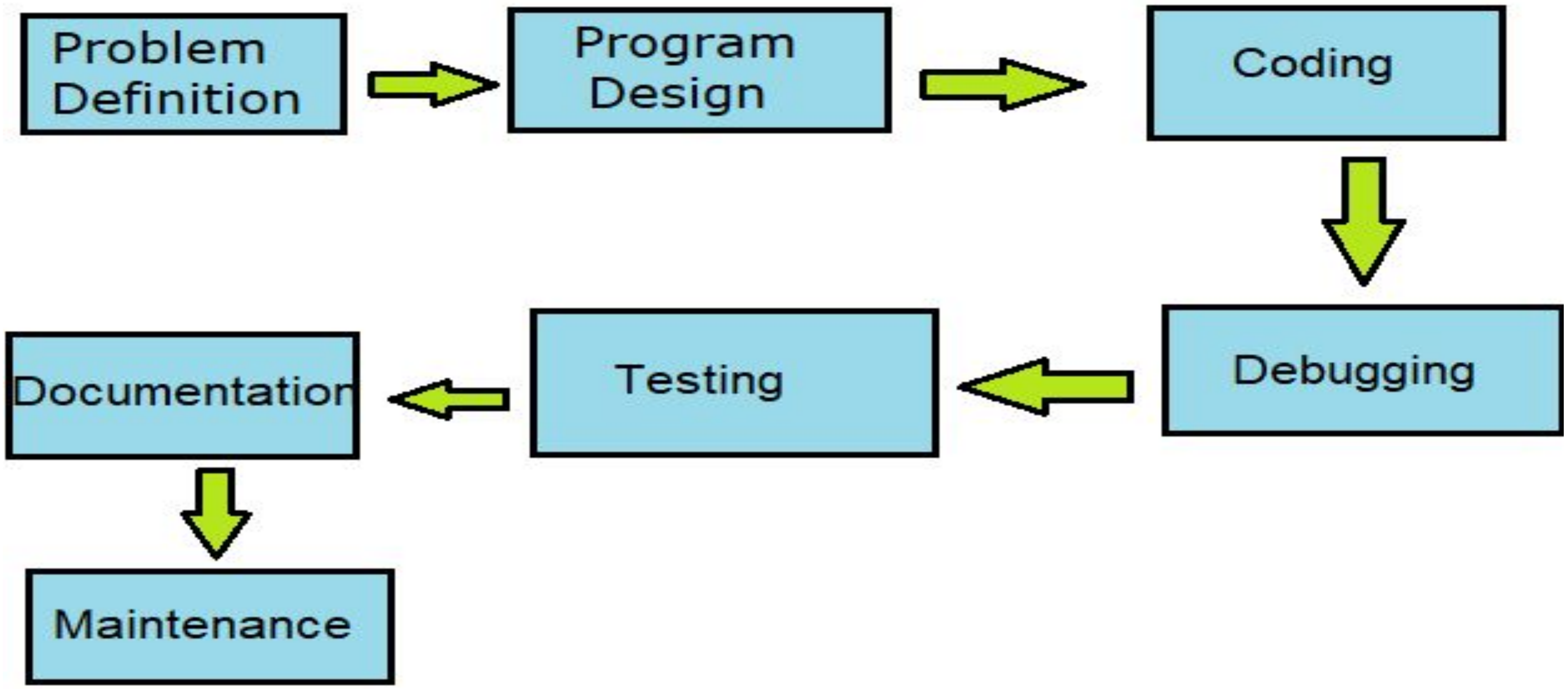
- C was invented to write an operating system called UNIX.
- C is a successor of B language which was introduced around the early 1970s.
- The language was formalized in 1988 by the American National Standard Institute (ANSI).
- The UNIX OS was totally written in C.
- Today C is the most widely used and popular System Programming Language.
- Most of the state-of-the-art software have been implemented using C.

History of C language

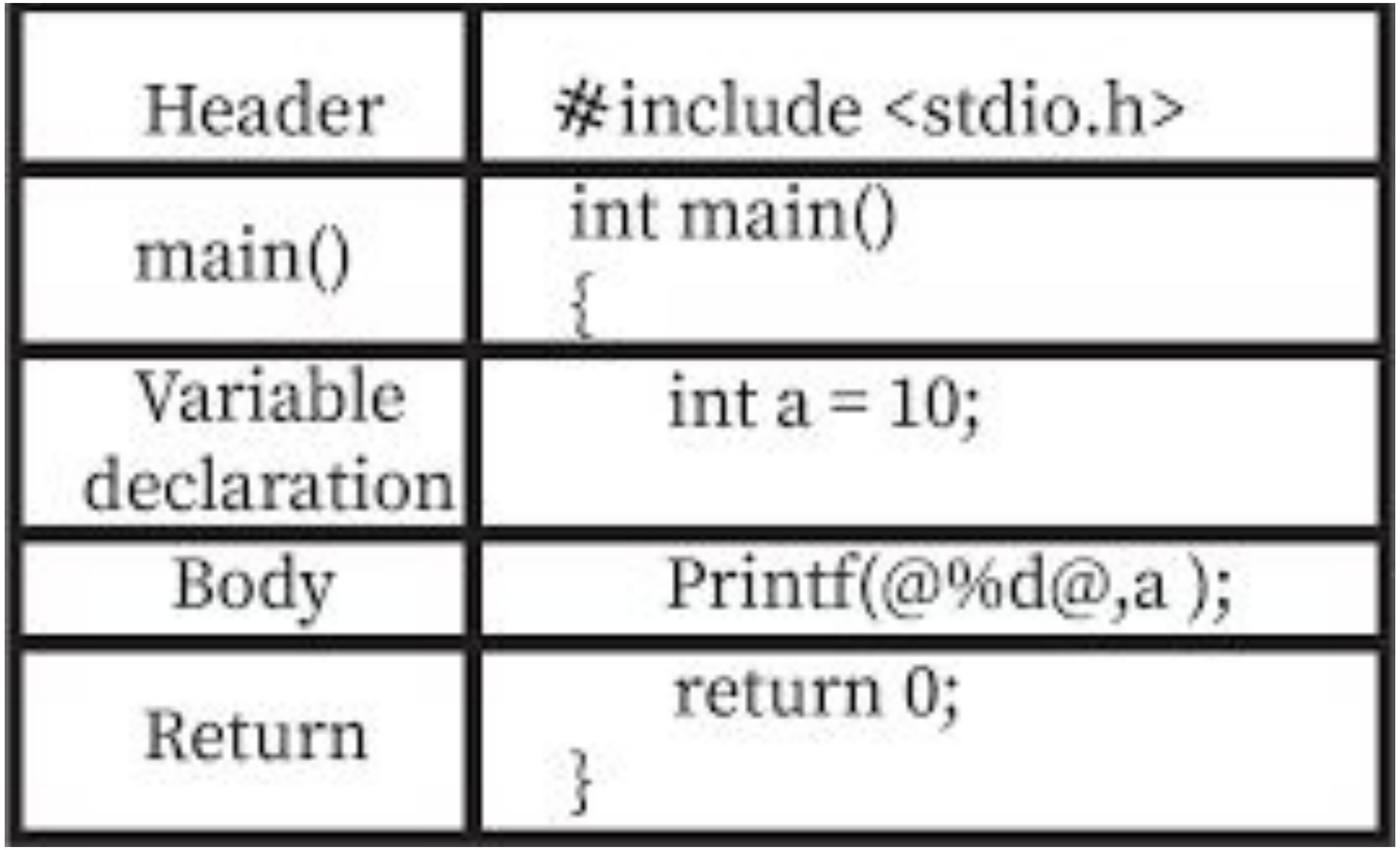
Let's see the programming languages that were developed before C language.

Language	Year	Developed By
Algol	1960	International Group
BCPL	1967	Martin Richard
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K & R C	1978	Kernighan & Dennis Ritchie
ANSI C	1989	ANSI Committee (American National Standards Institute)
ANSI/ISO C	1990	ISO Committee
C99	1999	Standardization Committee

Program Development Steps



Structure of C program



Header	#include <stdio.h>
main()	int main() {
Variable declaration	int a = 10;
Body	Printf(@%d@,a);

Return	return 0;
	}

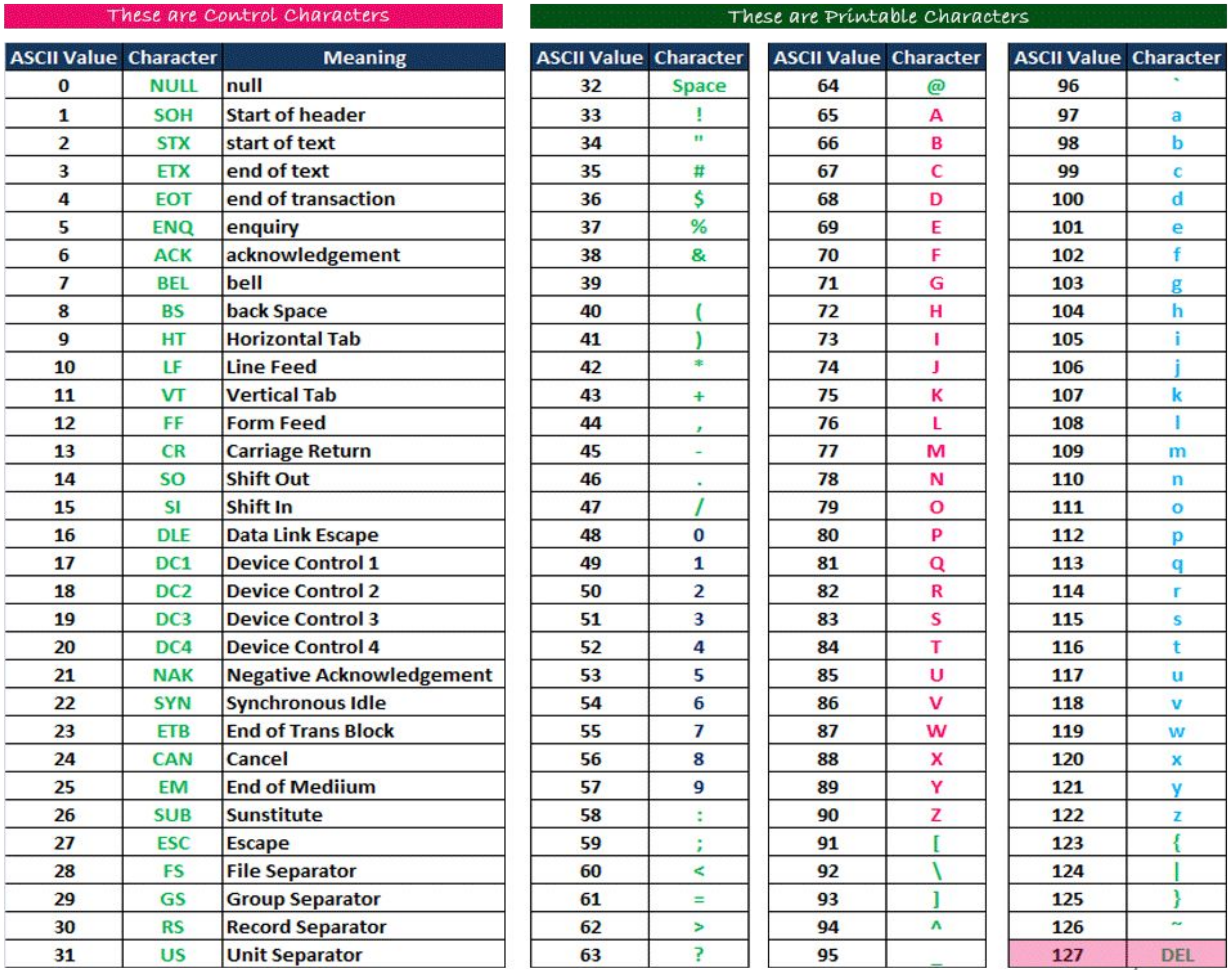
C Character Set

- As every language contains a set of characters used to construct words, statements, etc., C language also has a set of characters which include alphabets, digits, and special symbols.
- C language supports a total of 256 characters.
- C language character set contains the following set of characters...
    - Alphabets
    - Digits
    - Special Symbols

C Character Set

- Alphabets: lower case letters - a to z, UPPER CASE LETTERS - A to Z
- Digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Special Symbols: ~ @ # $ % ^ & * ( ) _ - + = { } [ ] ; : ' " / ? . > , < \ | tab newline space NULL bell backspace verticaltab etc.,

Commonly used characters in C with their ASCII values



These are Control Characters			These are Printable Characters					
ASCII Value	Character	Meaning	ASCII Value	Character	ASCII Value	Character	ASCII Value	Character
0	NULL	null	32	Space	64	@	96	`
1	SOH	Start of header	33	!	65	A	97	a
2	STX	start of text	34	"	66	B	98	b
3	ETX	end of text	35	#	67	C	99	c
4	EOT	end of transaction	36	$	68	D	100	d
5	ENQ	enquiry	37	%	69	E	101	e
6	ACK	acknowledgement	38	&	70	F	102	f
7	BEL	bell	39		71	G	103	g
8	BS	back Space	40	(	72	H	104	h
9	HT	Horizontal Tab	41	)	73	I	105	i
10	LF	Line Feed	42	*	74	J	106	j
11	VT	Vertical Tab	43	+	75	K	107	k
12	FF	Form Feed	44	,	76	L	108	l
13	CR	Carriage Return	45	-	77	M	109	m
14	SO	Shift Out	46	.	78	N	110	n
15	SI	Shift In	47	/	79	O	111	o
16	DLE	Data Link Escape	48	0	80	P	112	p
17	DC1	Device Control 1	49	1	81	Q	113	q
18	DC2	Device Control 2	50	2	82	R	114	r
19	DC3	Device Control 3	51	3	83	S	115	s
20	DC4	Device Control 4	52	4	84	T	116	t
21	NAK	Negative Acknowledgement	53	5	85	U	117	u
22	SYN	Synchronous Idle	54	6	86	V	118	v
23	ETB	End of Trans Block	55	7	87	W	119	w
24	CAN	Cancel	56	8	88	X	120	x
25	EM	End of Mediium	57	9	89	Y	121	y
26	SUB	Sunstitute	58	:	90	Z	122	z
27	ESC	Escape	59	;	91	[	123	{
28	FS	File Separator	60	<	92	\	124	|
29	GS	Group Separator	61	=	93	]	125	}
30	RS	Record Separator	62	>	94	^	126	~
31	US	Unit Separator	63	?	95	_	127	DEL

Keywords

- Keywords are predefined, reserved words used in programming that have special meanings to the compiler. Keywords are part of the syntax and they cannot be used as an identifier.
- **For example:**

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

Identifiers

- Identifier refers to name given to entities such as variables, functions, structures etc.
- Identifiers must be unique. They are created to give a unique name to an entity to identify it during the execution of the program.
- For example:

```
int money;
double accountBalance;
```

- Here, **money** and **accountBalance** are identifiers.

Data types

- **A data type, in programming, is a classification that specifies which type of value a variable has and what type of mathematical, relational or logical operations can be applied to it without causing an error.**

DATA TYPE	USED FOR	RANGE	SIZE (IN BYTES)	EXAMPLE
Integer	Whole numbers (%d)	32,768 to 32,767	2	7, 12, 999
Float (floating point)	Number with a decimal point (%f)		4	3.15, 9.06, 00.13
Character	Encoding text numerically (%c)	-128 to 127	1	97 (in ASCII, 97 is a lower case 'a')

Constants in C

- A constant is a value or variable that can't be changed in the program, for example: 10, 20, 'a', 3.4, "c programming" etc.

Constant	Example
Decimal Constant	10, 20, 450 etc.
Real or Floating-point Constant	10.3, 20.2, 450.6 etc.
Octal Constant	021, 033, 046 etc.
Hexadecimal Constant	0x2a, 0x7b, 0xaa etc.
Character Constant	'a', 'b', 'x' etc.
String Constant	"c", "c program", "c in javatpoint" etc.

```
#include<stdio.h>
void main(){
const float PI=3.14;
printf("The value of PI is: %f",PI);
}
```

```
#include<stdio.h>
#define PI 3.14;
void main(){
printf("The value of PI is: %f",PI);
}
```

Variables in C

- 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:
  - int a;
  - float b;
  - char c;

C Expressions

- An expression is a combination of constants and variables interconnected by one or more operators. An expression consists of one or more operands and one or more operators.
- num1 + num2 // variables num1 and num2 are operands and + is the operator used.

C Statements

- A program is a collection of statements.
- A statement is an instruction given to the computer to perform an action.
- There are three different types of statements in C:
    - Expression Statements
    - Compound Statements
    - Control Statements

C Statements

- An expression statement or simple statement consists of an expression followed by a semicolon (;).
- Given below are few examples of expression statements:

```
a = 100;
b = 20;
c = a / b;
```

- A compound statement also called a block, consists of several individual statements enclosed within a pair of braces { }.
- Given below are few examples of compound statements:

```
{
   a = 3;
   b = 10;
   c = a + b;
}
```

- A single statement or a block of statements can be executed depending upon a condition using control statements like if, if-else, etc. We shall learn more about control statements in later sections.
- Given below is an example of if control statement:

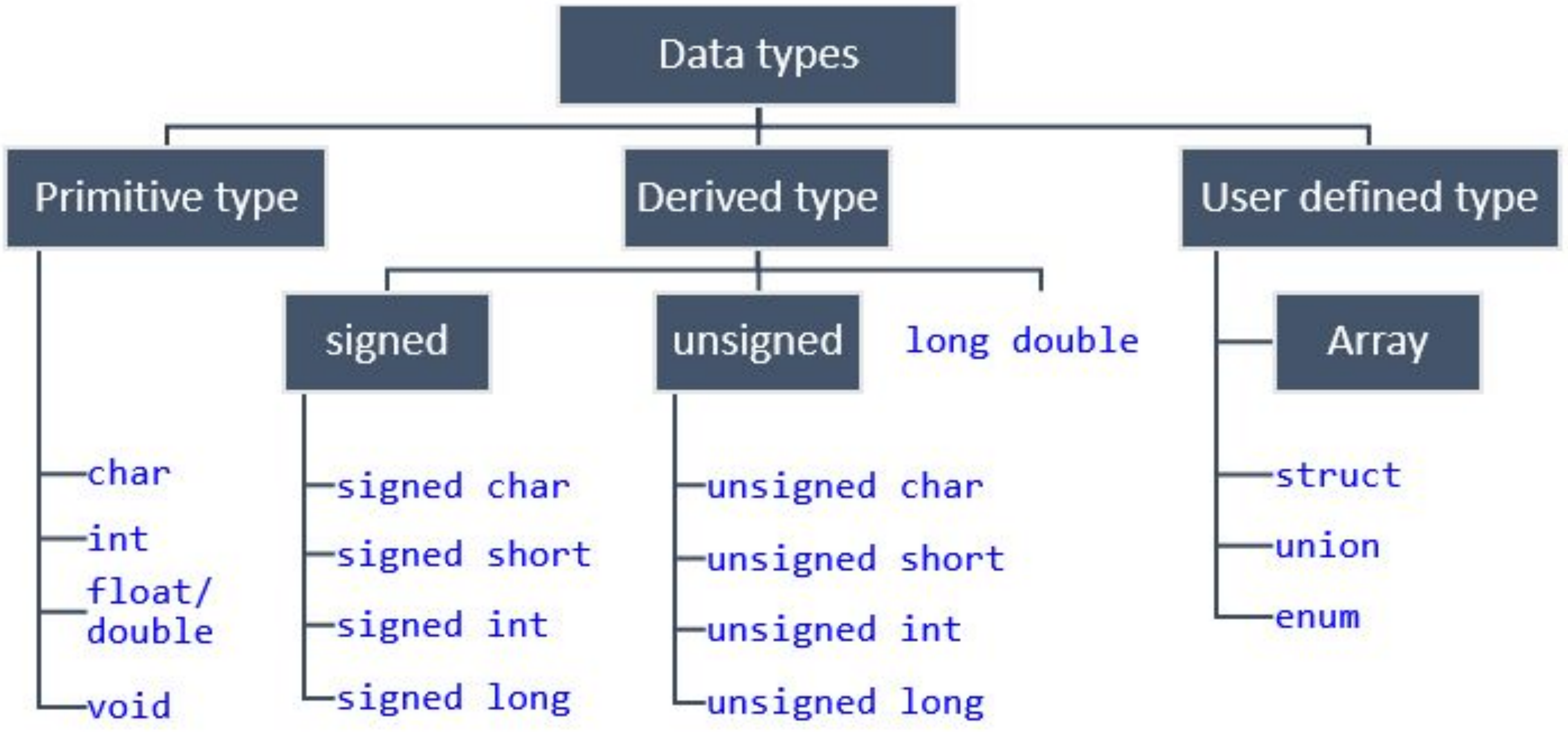
```
a = 10;
if ( a > 5) {
   b = a + 10;
}
```

C 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 (Miscellaneous Operator: &/?/!/$ etc.)

The precedence and associativity of C operators is given below:

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


Data types
Primitive type
Derived type
User defined type
char
int
float/ double
void
signed
signed char
signed short
signed int
signed long
unsigned
unsigned char
unsigned short
unsigned int
unsigned long
long double
Array
struct
union
enum

Extended Data Type

Type name	Description
bigint	signed integer with a size of 8 bytes
ubigint	unsigned integer with a size of 8 bytes
int4	signed integer with a size of 4 bytes
uint4	unsigned integer with a size of 4 bytes
int2	signed integer with a size of 2 bytes
uint2	unsigned integer with a size of 2 bytes
int1	signed integer with a size of 1 byte
uint1	unsigned integer with a size of 1 byte
mint	signed machine-dependent C int
muint	unsigned machine-dependent C int
mlong	signed machine-dependent C long
mulong	unsigned machine-dependent C long

dec_t	DECIMAL data type structure
dtime_t	DATETIME data type structure
intrvl_t	INTERVAL data type structure
loc_t	TEXT / BYTE locator structure

Program-1

- Installation C IDE, Basic Structure of C program.Format Specifiers, Escape Character. Run time input/Output Programs.

Program-2

A. Write a c program to calculate Area of Rectangle, Perimeter of a Rectangle and Diagonal of a Rectangle.
B. Write a c program to calculate Area of square,Perimeter of a square and Diagonal of a square.
C. Write a c program to calculate total area ofCylinder and volume of a cylinder.

Conditional Statements

- Conditional statements help you to make a decision based on certain conditions. These conditions are specified by a set of conditional statements having boolean expressions which are evaluated to a boolean value of true or false.
- There are the following types of conditional statements in C.
    - If statement
    - If-Else statement
    - Nested If-else statement
    - If-Else If ladder
    - Switch statement

If statement

- The single if statement in C language is used to execute the code if a condition is true. It is also called a one-way selection statement. When we use the if condition, we pass the argument and if the argument will be satisfied then the respective code will be executed otherwise nothing can happen.

```
#include<stdio.h>
#include<conio.h>
void main()
{
       int num=0;
       printf("enter the number");
       scanf("%d",&num);
       if(n%2==0)
       {
              printf("%d number in
     even",num);
       }
       getch();
}
```

If-else statement

- The if-else statement in C language is used to execute the code if the condition is true or false. It is also called a two-way selection statement.

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int num=0;
  printf("enter the number");
  scanf("%d",&num);
  if(n%2==0)
  {
      printf("%d number in even", num);
  }
  else
  {
      printf("%d number in odd",num);
  }
  getch();
}
```

Nested If-else statement

- The nested if...else statement is used when a program requires more than one test expression. It is also called a multi-way selection statement. When a series of the decision are involved in a statement, we use the if-else statement in nested form.
- Nested if-else statements can be useful when we can have multiple sources of expression and the values and based on the specific value, we need to check nested conditions.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
        int a,b,c;
        clrscr();
        printf("Please Enter 3 number");
        scanf("%d%d%d",&a,&b,&c);
```

Step-1

```
if(a>b)
        {
                if(a>c)
                {
                printf("a is greatest");
                }
                else
                {
                printf("c is greatest");
                }
        }
```

Step-2

```
else
        {
                if(b>c)
                {
                printf("b is greatest");
                }
                else
                {
                printf("c is greatest");
                }
        }
        getch();
}
```

Step-3

If..else If ladder

- The if-else-if statement is used to execute one code from multiple conditions. It is also called a multipath decision statement. It is a chain of if..else statements in which each if statement is associated with an else if statement and the last would be an else statement.

```
#include<stdio.h>
#include<conio.h>
void main( )
{
            int a;
            printf("enter a number");
            scanf("%d",&a);
            if( a%5==0 && a%8==0)
            {
                printf("divisible by both 5 and 8");
            }
            else if( a%8==0 )
            {
                printf("divisible by 8");
            }
            else if(a%5==0)
            {
                printf("divisible by 5");
            }
            else
            {
                printf("divisible by none");
            }
           getch();
}
```

Switch Statement

- switch statement acts as a substitute for a long if-else-if ladder that is used to test a list of cases. A switch statement contains one or more case labels that are tested against the switch expression. When the expression match to a case then the associated statements with that case would be executed.

- We have seen the way of using conditional statements such as if, if-else. if-else ladder, but the need for an additional way of dealing with conditional statements may seem unnecessary but based on the certain usage, switch case was defined to check for the single condition, and based on the multiple cases, code can be executed.

```
#include <stdio.h>
#include <conio.h>
void main()
{
        int num = 8;
       switch (num)
       {
           case 7:
              printf("Value is 7");
              break;
           case 8:
              printf("Value is 8");
              break;
           case 9:
              printf("Value is 9");
              break;
           default:
              printf("Out of range");
              break;
        }
        getch();
}
```

Program-3

A. The total distance travelled by vehicle in ‘t’ seconds is given by distance = **ut+1/2(at^2)** where ‘u’ and ‘a’ are the initial velocity (m/sec.) and acceleration (m/sec2). Write C program to find the distance travel   led at regular intervals of time given the values of ‘u’ and ‘a’. The program should provide the flexibility to the user to select his own time intervals and repeat the calculations for different values of ‘u’ and ‘a’.

B. Write a C program, which takes two integer operands and one operator from the user, performs the operation and then prints the result. (Consider the operators +, -, *, /, % and use Switch Statement).

Loops

- Loop is used to execute the block of code several times according to the condition given in the loop. It means it executes the same code multiple times so it saves code
- There are 3 types of loop –
    - while loop
    - do – while loop
    - for loop

while Loop –

- While loop execute the code until condition is false.

- **Syntax**

```
while(condition){
//code
}
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i = 20;
while( i <=20 ) {
printf ("%d " , i );
i++;
}
getch();
}
```

Output:

```
20
```

do – while loop

- It also executes the code until condition is false. In this at least once, code is executed whether condition is true or false but this is not the case with while. While loop is executed only when the condition is true.

- **Syntax**

```
do{
//code
}while(condition);
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i = 20;
do{
printf ("%d " , i );
i++;
}
while( i < =20 );
getch();
}
```

Output:

```
20
21
```

for Loop

- It also executes the code until condition is false. In this three parameters are given that is
  - Initialization
  - Condition
  - Increment/Decrement

- **Syntax**

```
for(initialization; condition; increment/decrement)
{
//code
}
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i;
for( i = 20; i < 25; i++) {
printf ("%d " , i);
}
getch();
}
```

Output:

```
20
21
22
23
24
```

Looping Related Programs

21. C program to print ODD numbers from 1 to N using while loop.
22. C program to print numbers from 1 to 10 using while loop.
23. C Program to find factorial of a number.
24. C Program to find sum of first N natural number, N must be taken by the user.
25. C program to print all prime numbers from 1 to N.
26. C program to print all even and odd numbers from 1 to N.
27. C program to print all Armstrong numbers from 1 to N.

Looping Program – Pattern Based

Program-28

```
*
* *
* * *
* * * *
* * * * *
```

Program-29

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Program-30

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

Program-31

```
* * * * *
* * * *
* * *
* *
*
```

Program-32

```
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
```

Program-33

```
* * * * * * * * *
  * * * * * * *
    * * * * *
      * * *
        *
```

Query?