Module-1

**History of C programming language and its features**

**1.1 Algorithm & Flowchart : Three construct of Algorithm and flowchart: Sequence, Decision (Selection) and Repetition**

**1.2 Character Set, Identifiers and keywords, Data types, Constants, Variables.**

**1.3 Operators-Arithmetic, Relational and logical, Assignment, Unary, Conditional, Bitwise, Comma, other operators. Expression, statements, Preprocessor, Structure of basic C program.**

**Q1.History and features of C.**

C programming language is a MUST for students and working

professionals to become a great Software Engineer specially when they

are working in Software Development Domain. Following are the key

advantages of learning 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

Features:

1) Simple

C is a simple language in the sense that it provides a structured approach (to break the problem

into parts), the rich set of library functions, data types, etc.

2) Machine Independent or Portable

Unlike assembly language, c programs can be executed on different machines with some machine

specific changes. Therefore, C is a machine independent language.

3) Mid-level programming language

Although, C is intended to do low-level programming. It is used to develop system applications

such as kernel, driver, etc. It also supports the features of a high-level language. That is why it is

known as mid-level language.

4) Structured programming language

C is a structured programming language in the sense that we can break the program into parts

using functions. So, it is easy to understand and modify. Functions also provide code reusability.

5) Rich Library

C provides a lot of inbuilt functions that make the development fast.

6) Memory Management

It supports the feature of dynamic memory allocation. In C language, we can free the allocated

memory at any time by calling the free() function.

7) Speed

The compilation and execution time of C language is fast since there are lesser inbuilt functions and

hence the lesser overhead.

8) Pointer

C provides the feature of pointers. We can directly interact with the memory by using the pointers.

We can use pointers for memory, structures, functions, array, etc.

9) Recursion

In C, we can call the function within the function. It provides code reusability for every function.

Recursion enables us to use the approach of backtracking.

10) Extensible

C language is extensible because it can easily adopt new features.

**Q2. Algorithm and flowchart. 3 constructs for algorithm and flowchart.**

Algorithm:

An algorithm is a procedure used for solving a problem.

It a sequence of finite steps to solve a particular problem.

Each step results in an action.

A well designed algorithm is guaranteed to terminate.

Flowchart:

A flowchart is a diagrammatic representation of algorithm to plan the solution to the

problem.

Constructed by using special geometrical symbols where each symbol represents an

activity. The activity would be input/output of data, computation/processing of data etc.

Three constructs of algorithm and flowchart:

1) Sequence-Each step or process in algorithm is executed in specified order .

2) Selection / decision - if .... then, if.....then.....else - In an algorithm the algorithm is either true or false , there is no state in between . The decision

can be stated as

3) Iteration / Repetition-Repeat ,While, if .. then.. go

Advantages of using flowchart:

1. Communication: They are good visual aid to understand the program.

2. Quicker grasps of relationships.

3. Effective analysis.

4. Synthesis

5. Proper program documentation

6. Efficient coding

7. Orderly debugging & testing of programs.

8. Efficient program maintenance.

Limitations of using flowchart:

1. Complex logic

2. Alterations & Modifications

3. Reproduction

4.Loss of Objective

**Q3. State and explain character set,** **Identifiers and keywords, Data types, Constants, Variables of C.**

Character set:

C language consist of some characters set, numbers and some special symbols. The character set of C consist of all the alphabets of English language.

C consist of

* Alphabets a to z, A to Z
* Numeric 0,1 to 9
* Special Symbols { , } , [ , ] , ? , + , - , \* , / , % , !, ; , and more

Tokens:

The words formed from the character set are building blocks of C and are sometimes known as tokens. Tokens represent the individual entity of language. The following different types of token are used in C

1. Identifiers 2)Keywords 3)Constants 4) Operators 5)Punctuation Symbols

Identifiers:

* A 'C' program consist of two types of elements , user defined and system defined.
* Identifier is nothing but a name given to these elements.
* An identifier is a word used by a programmer to name a variable , function, or label.
* Identifiers consist of letters and digits if any order, except that the first character must be letter.
* Both Upper and lowercase letters can be used

Keywords:

* Keywords are nothing but system defined identifiers.
* Keywords are reserved words of the language.
* They have specific meaning in the language and cannot be used by the programmer as variable or constant names.
* C is case sensitive, it means these must be used as it is.
* 32 Keywords in C Programming

Variable:

* A variable is nothing but a name given to a storage area that our programs can manipulate. Each variable in C has a specific type, which determines the size and layout of the variable's memory; the range of values that can be stored within that memory; and the set of operations that can be applied to the variable.
* The name of a variable can be composed of letters, digits, and the underscore character. It must begin with either a letter or an underscore. Upper and lowercase letters are distinct because C is case-sensitive.

Constants:

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

e.g. 1, 2.5,

* As mentioned, an identifier also can be defined as a constant.

e.g. const double PI = 3.14

* Here, PI is a constant. Basically what it means is that, PI and 3.14 is same for this program.

Integer constants

* A 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)

Floating-point constants

* A floating point constant is a numeric constant that has either a fractional form or an exponent form.For example: 2.0,0.0000234,-0.22E-5

Character constants

* A character constant is a constant which uses single quotation around characters. For example: 'a','l', 'm', 'F'

String constants

* String constants are the constants which are enclosed in a pair of double-quote marks. For
* example: "good" ,"x","Earth is round\n"

**Q4. What are escape sequences?**

* For example: \n is used for newline. The backslash ( \ ) causes "escape" from the normal way the characters are interpreted by the compiler. Escape Sequences Character

\b

Backspace

\f

Form feed

\n

Newline

\r

Return

\t

Horizontal tab

\v

Vertical tab

\\

Backslash

\'

Single quotation mark

\"

Double quotation mark

\?

Question mark

\0

Null character

**Q5. State and explain the operators in C.**

C programming has wide range of operators to perform various operations. For better

understanding of operators, these operators can be classified as:

* Arithmetic Operators
* Increment and Decrement Operators
* Assignment Operators
* Relational Operators
* Logical Operators
* Conditional Operators
* Bitwise Operators
* Special Operators

*Arithmetic Operator:*

+:addition or unary plus

-: subtraction or unary minus

\*: multiplication

/: division

%: remainder after division( modulo division)

*Increment and Decrement Operators:*

1. C programming has two operators increment ++ and decrement -- to

change the value of an operand (constant or variable) by 1.

2. Increment ++ increases the value by 1 whereas decrement -- decreases the

value by 1.

3. These two operators are unary operators, meaning they only operate on a

single operand.

eg. int a=10, b=100

++a = 11

--b = 99

*C Assignment Operators:*

An assignment operator is used for assigning a value to a variable.

Example:

* =

a = b

a = b

* +=

a += b

a = a+b

* -=

a -= b

a = a-b

* \*=

a \*= b

a = a\*b

* /=

a /= b

a = a/b

* %=

a %= b

a = a%b

*C Relational Operators:*

A relational operator checks the relationship between two operands. If the relation is true, it returns 1; if the relation is false, it returns value 0.

Relational operators are used in decision making and loops.

Example:

== Equal to

5 == 3 returns 0

> Greater than

5 > 3 returns 1

< Less than

5 < 3 returns 0

!= Not equal to

5 != 3 returns 1

>= Greater than or equal to

5 >= 3 returns 1

<= Less than or equal to

5 <= 3 return 0

*C Logical Operators:* &&, || and ! are the three logical operators.

expr1 && expr2 has a value 1 if expr1 and expr2 both are nonzero.

expr1 || expr2 has a value 1 if expr1 and expr2 one of them is nonzero.

!expr1 has a value 1 if expr1 is zero else 0.

Example

if ( marks >= 40 && attendance >= 75 ) grade = ‘P’

If ( marks < 40 || attendance < 75 ) grade = ‘N’

*C Bitwise Operators:* Bitwise Operators are used for manipulating data at the bit level, also called bit levelprogramming. Bitwise operates on one or more bit patterns or binary numerals at the

level of their individual bits. They are used in numerical computations to make the

calculation process faster.

&

Bitwise AND operator

|

Bitwise OR operator

^

Bitwise exclusive OR operator

~

Binary One’s Complement

Operator is a unary operator

<<

Left shift operator

>>

Right shift operator

*Conditional Operator (ternary):*

The first operand/expression is evaluated, and its value determines whether the second or third

operand/expression is evaluated:

If the value is true, the second operand/expression is evaluated.

If the value is false, the third operand/expression is evaluated.

The result is the value of the second or third operand/expression. The syntax is:

First operand ? second operand : third operand

e.g. a=10; b=15;

X=(a>b)?a : b;

*C Comma Operator:*

The comma operator is type of special operators in C which evaluates first operand and then

discards the result of the same, then the second operand is evaluated and result of same is

returned. Comma operator is a binary operator and has the least precedent of all C operators.

*C SizeOf Operator:*

* C provides a very useful operator sizeof, for calculating the size of an data item or type.
* It takes a single operand that may be a type name (e.g.,int) or an expression (e.g.,100) and returns the size of a specified entity in bytes.
* The outcome is totally machine dependent.

**Q6. What is operator precedence in C?**

* Operator precedence: a rule used to clarify unambiguously which operations (operator

and operands) should be performed first in the given (mathematical) expression.

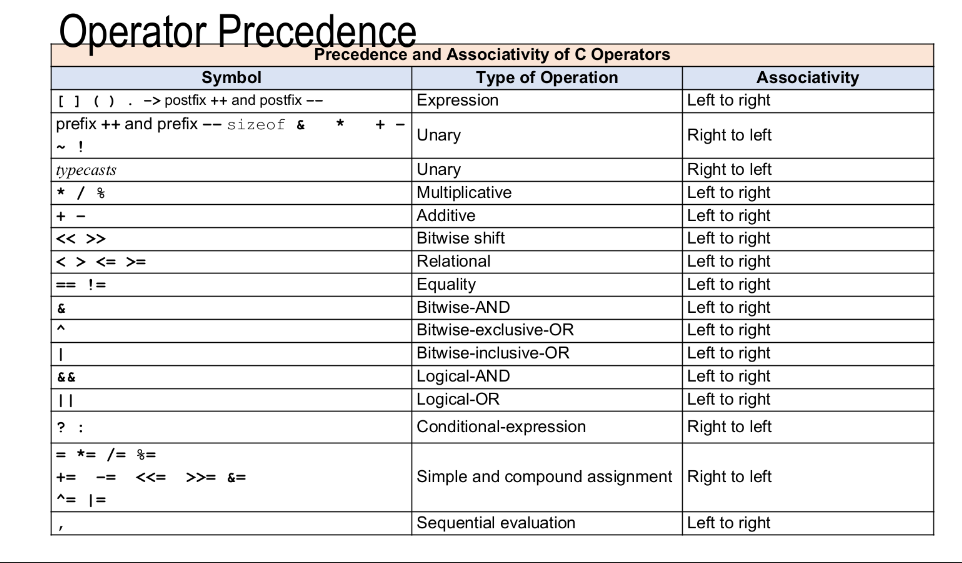
* Use precedence levels that conform to the order commonly used in mathematics.
* However, parentheses take the highest precedence and operation performed from the

innermost to the outermost parentheses.

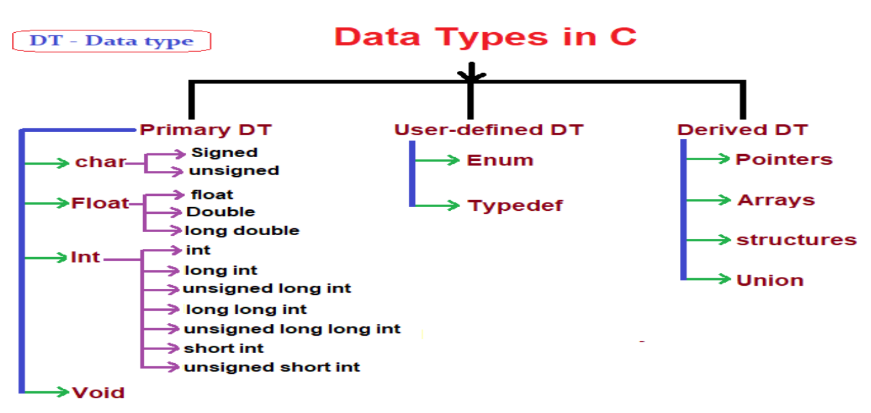
* Precedence and associativity of C operators affect the grouping and evaluation of

operands in expressions.

* Is meaningful only if other operators with higher or lower precedence are present.
* Expressions with higher-precedence operators are evaluated first.



**Q7. State and explain the data types in C.**



*User Defined Data Types - typedef*

* typedef declaration do not introduce new type but introduces new name or creating

synonym (or alias) for existing type.

* To construct shorter or more meaningful names for types already defined by the

language or for types that you have declared.

* A typedef declaration does not reserve storage.
* The name space for a typedef name is the same as other ordinary identifiers.
* Therefore, a program can have a typedef name and a local-scope identifier by the

same name.

* The syntax is,

typedef type-declaration the\_synonym

* You cannot use the typedef specifier inside a function definition.
* When declaring a local-scope identifier by the same name as a typedef, or when

declaring a member of a structure or union in the same scope or in an inner scope,

the type specifier must be specified.

* For example,

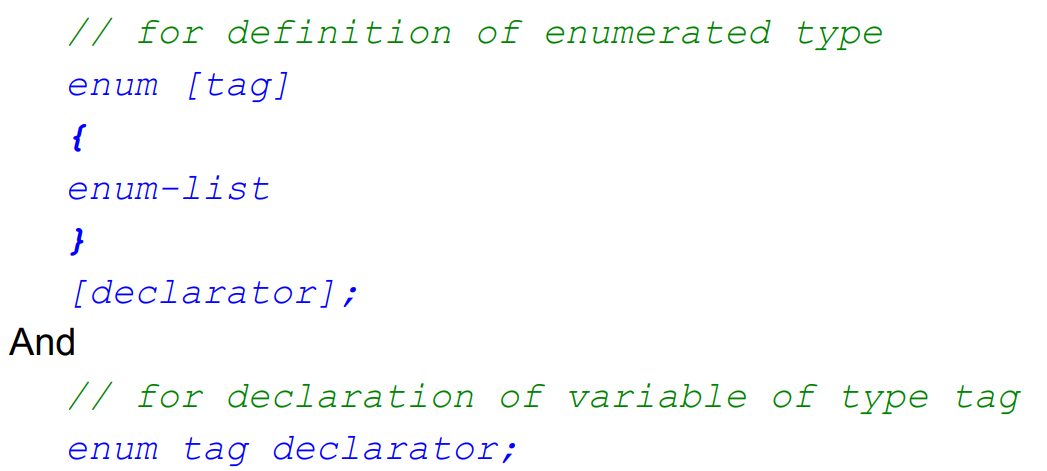
typedef float TestType;

User Defined Data Types – enum

• enum is another user-defined type consisting of a set of named constants called enumerators.

• Using a keyword enum, it is a set of integer constants represented by identifiers.

• The syntax is shown below



• The values in an enum start with 0, unless specified otherwise, and are incremented by 1. For example, the following enumeration,

• enum days {Mon, Tue, Wed, Thu, Fri, Sat, Sun};

• Creates a new data type, enum days, in which the identifiers are set automatically to the integers 0 to 6.

• To number the days 1 to 7, use the following enumeration

• enum days {Mon = 1, Tue, Wed, Thu, Fri, Sat, Sun};

• Or we can re-arrange the order

• enum days {Mon, Tue, Wed, Thu = 7, Fri, Sat, Sun};

• These enumeration constants are, in effect, symbolic constants whose values can be set automatically

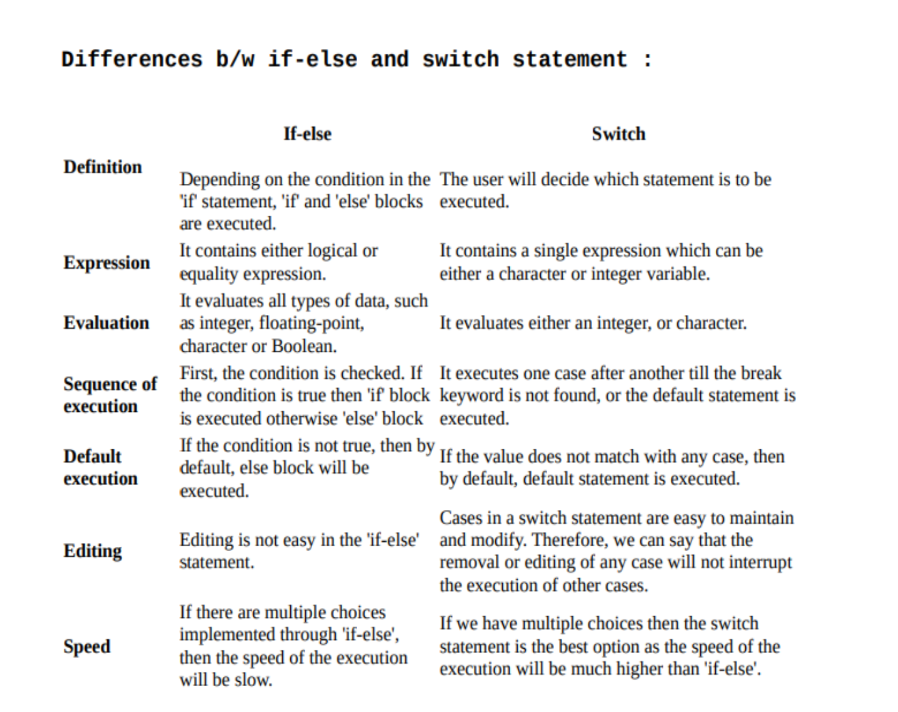
Module-2

**2.1 Decision making statements- if statement, if-else statement , ifelse-if ladder, nested if-else, switch statement**

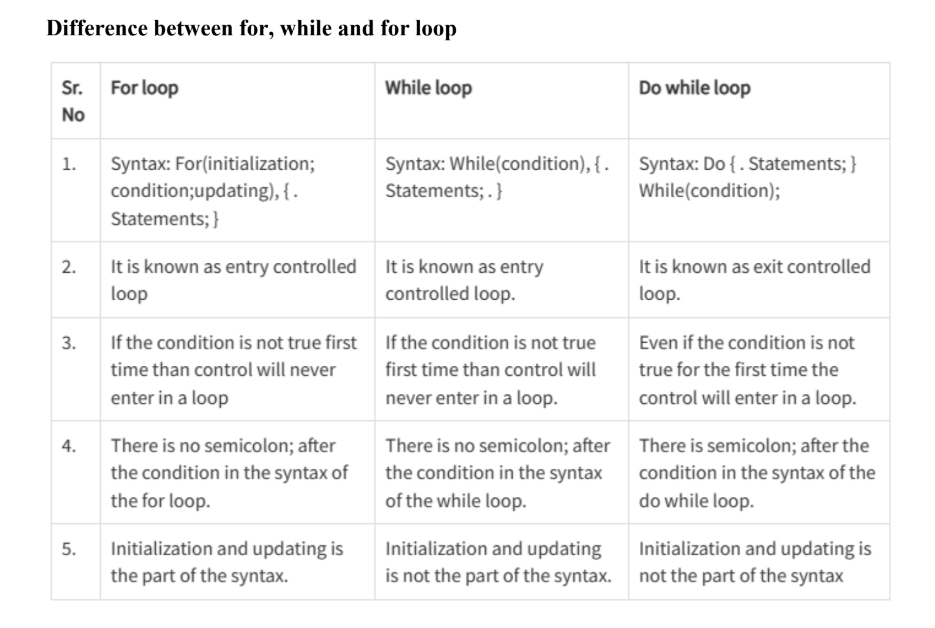
**2.2 Looping – while , do-while, for**

**2.3 Jump Statements- break, continue, goto, return, exit**

**Q1.Differentiate between if-else and switch statements.**

****

**Q2. Difference between for, while and do-while loops.**

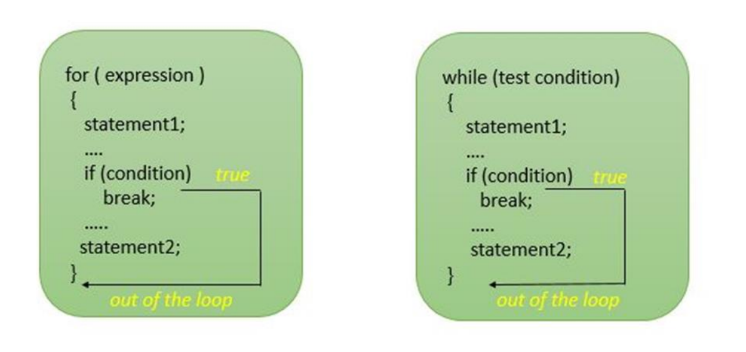
****

**Q3. Explain break statement.**

* Sometimes it becomes necessary to come out of the loop even before loop condition

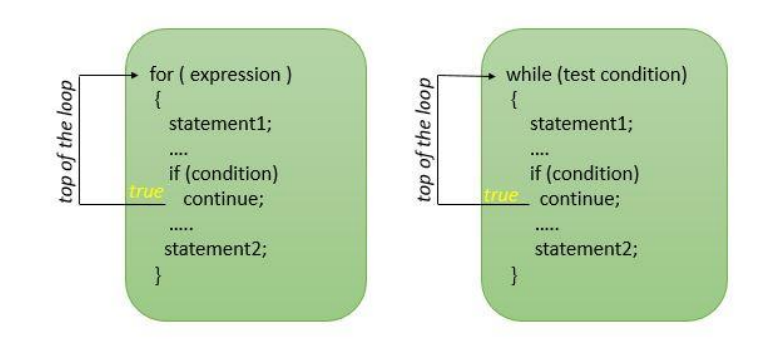
becomes false then break statement is used.

* Break statement is used inside loop and switch statements.
* It cause immediate exit from that loop in whichit appears and it is generally written with condition.
* It is written with the keyword as break.
* When break statement is encountered loop is terminated and control is transferred to the statement, immediately after loop or situation where we want to jump out of the loop instantly without waiting to get back to conditional state.
* When break is encountered inside any loop, control automatically passes to the first statement after the loop.
* The break statement is almost always used with if...else
* statement inside the loop.

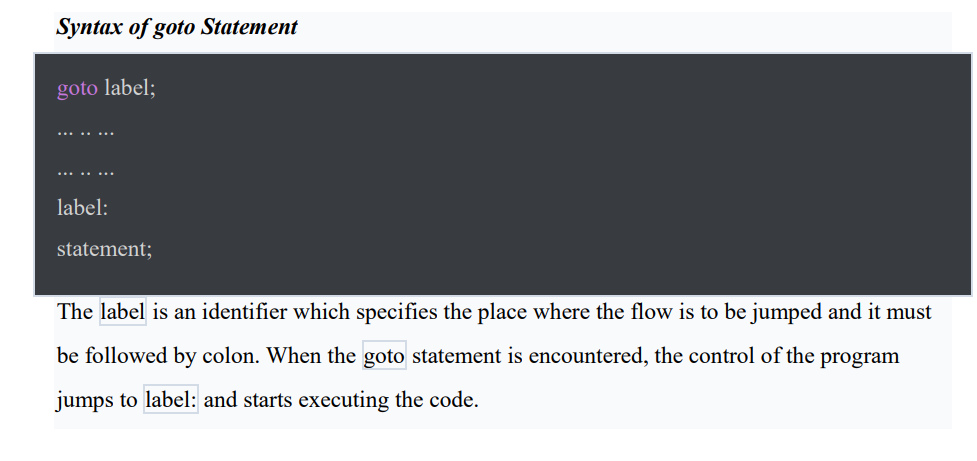


**Q4. Explain continue statement.**

* Continue statement is used for continuing next iteration of loop after skipping some statement of loop. When it encountered control automatically passes through the beginning of the loop. It is usually associated with the if statement. It isuseful when we want to continue the program without executing any part of the program. The difference between break and continue is, when the break encountered loop is terminated and it transfer to the next statement and when continue is encounter control come back to the beginning position.
* In a while and do while loop after continue statement control transfer to the test condition and then loop continue where as in, for loop after continue control transferred to the updating expression and condition is tested.

****

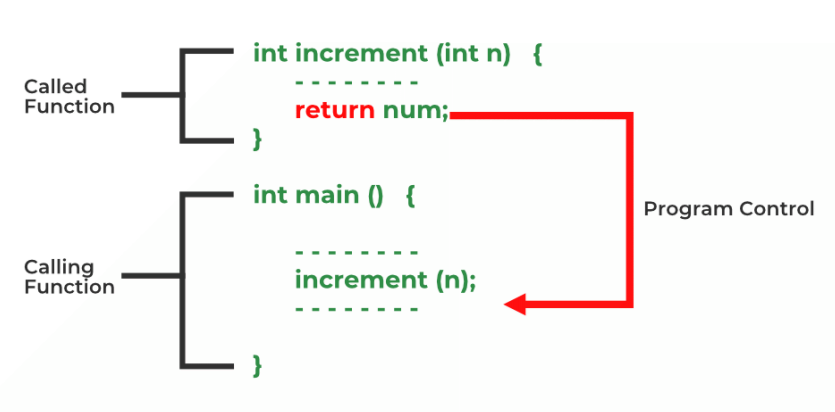
**Q5. Explain goto label statement.**

****

**Q6. Return statement.**

return statement (keyword return) : C return statement ends the execution of a function and returns the control to the function from where it was called. The return statement may or may not return a value depending upon the return type of the function.

In C, we can only return a single value from the function using the return statement and we have to declare the data\_type of the return value in the function definition/declaration

****

**Q7. Exit statement.**

C exit() function is a standard library function defined in the header file used to terminate C program execution immediately with an error code.

\* void exit(int status)

Exit Success is indicated by exit(0) statement which means successful termination of the program, i.e. program has been executed without any error or interrupt.

Exit Failure is indicated by exit(1) which means the abnormal termination of the program, i.e. some error or interrupt has occurred.

Module-3

**3.1 Introduction to Functions, declaring and defining function, calling function, passing arguments to a function, recursion and its application.**

**3.2 Library functions – getchar( ), putchar( ), gets( ), puts( ), Math function, Ctype functions**

**3.3 Storage classes in C-auto, extern, static, register.**

**Q1.Two types of functions.**

Functions can be divided into two categories :

▪ Predefined functions (standard functions)

Built-in functions provided by C that are used by programmers without having to write any code for them. i.e: printf( ), scanf( ), etc

▪ User-Defined functions

Functions that are written by the programmers themselves to carry out various individual tasks.

Q2. Explain function prototyping/declaration.

* A function prototype will tell the compiler that there exist a function with this name defined somewhere in the program and therefore it can be used even though the function has not yet been defined at that point.
* Function prototypes need to be written at the beginning of the program.
* If the function receives some arguments, the variable names for the arguments are not needed. Only the data type need to be stated.