Subject : PROGRAMMING IN C

Subject Code : 18K2CS03

Staff Name : R.Valarmathy

G.MuthamizSelvi

N.Anuradha

UNIT I: Overview of C- importance of C- Sample Programs- Basics Structure of C programs- Executing C programs- Constants, Variables and Data types: Keywords and Identifiers- Operators and Expressions

UNIT II: Managing Input and Output Operations: Decision Making and Branching: Introduction- Decision Making with if statement- Simple if statement- if …..Else statement- Nested if- Else if Ladder- Switch Statement- Ternary operator- Goto statement. Decision Making and Looping: While, do, for, Jumps in loops.

UNIT III: Arrays: Introduction- One dimensional Array- Two dimensional Arrays- Multidimensional--Arrays. Character Array and Strings: Introduction- Declaring and initializing String Variables- String Handling Functions.

Programming in C[18K2CS03]

UNIT 1

OVERVIEW OF C

- History of c
- Importance of c
- Sample program
- Structure of c program

History of ANSI C

YEAR	LANGUAGE	DEVELOPER
1960	ALGOL	International group
1967	BCPL	Martin Richards
1970	B	Ken Thompson
1972	Traditional c	Dennies Ritchie
1978	K&R C	Kerningam and Ritchie
1989	ANSI C	ANSI Committee
1990	ANSI C / ISO C	ISO Committee
1999	C99	Standardization committee

Importance of C

- C is a programming Language
- C is a Robust Language
- Programs written in c are efficient and fast
- C is highly portable.i.e.., software written for one computer and run on another computer
- An important features of c is its ability to extend itself.A c program is basically a collection of functions

Format of Simple C Program

```
                Function name <----------        main()
start of program<-------------------------        {
                                                  ……
Program statements<----------------------          ……
                                                   ……
    End of program<-----------------------          }
```

SAMPLE PROGRAM: Adding Two Numbers

```
main() {
int  number; float
amount;
number=100;
amount=30.75+75.35;
printf(“%d\n”,number);
printf(“%5.2f”,amount); }
```

Output

100

106.10

Basic Structure of C Programs

Documentation section
Link section
Definition section
Global Declaration section

Main() Function section
{
 	
}

Subprogram section   (User defined fuctions)

C Structure

- Documentation Section. It is the section in which you can give comments to make the program more interactive.
- Preprocessor directives Section. This section involves the use of header files that are to included necessarily
- Definition section. This section involves the variable definition and declaration in C.
- Global declaration Section. This section is used to define the global variables to be used in the programs.
- Function prototype declaration section.

CHAPTER 2

constants,variables and datatypes

- Character Set
- Tokens
- Constants
- Variables
- Datatypes

Character set

The C character set includes the upper case letter A to Z,the lower case letter a to z ,the decimal degits 0 to 9 and certain special charecters.

- Letters — a,b,c…..z
- Digits — 0,1,2,……9
- Special Characters — , . ; : ‘? ”!|/ \~_&$%^*-+< >{}[]()#
- White Spaces — blank space,horizontal tab, carriage return,new line,form feed

Tokens

Smallest individual units in a program are known as tokens.There are 6 types of tokens.

1.Keywords
2.Identifiers.
3.Constants
4.Strings
5.Special symbols
6.Operators

keywords

- Keywords are Predefined tokens in c.
- These are also called reserved words
- Keywords have special meaning to the compiler.
- These keywords can be used only for their intended action;They cannot be used for any other purpose.C has 32 Keywords.

C Keywords

Auto	**Double**	**int**	**struct**
Break	Else	long	Switch
Case	Enum	register	Typedef
Char	Extern	return	Union
Const	Float	short	Unsigned
Continue	For	signed	Void
Default	Goto	sizeof	Volatile
Do	If	static	while

Identifiers

- Identifiers are distinct names given to programs elements such as constants variables,etc…
- An identifiers is a sequence of letters,digits and the special character'_'(underscore).

1.It must start with either a letter or underscore.

2.No commas or blanks are allowed within a variable name.

3.Identifiers are case sensitive.

4.An identifiers can be of any length.

5.No special symbol can be used in a variable name.

Constants

- Constant is a literal,which remain unchanged during the execution of a program.
- Constant is a fixed value that cannot be altered during the execution of a program.
- Two types of constants.

1.Numeric constants

int,real

2.Character constants

single character,string

Integer constants

- An integer constant refer to a sequence of digits.
- It should not contain either a decimal point or exponent.
- Commas,blanks and non digit characters are not allowed in integer constants.
- The value of integer constant cannot exceed specifieds limits.The valid range is -32768 to +32767

Real constants

- Real values are often called floating point constant. There are two ways to represent a real constant decimal form and exponential form.
- In exponetial form of representation ,the real constant is represented in two parts.The part appearing before ‘e’ is called mantissa,where as the part following ‘e’ is called exponent.
  - The mantissa part and the exponential part should be separated by a letter e
  - The mantissa part may have a positive or negative sign. – Default sign of mantissa part is positive.
  - Exponent must have atleast one digit, which must be a positive or negative integer.default sign is positive.
  - Range of real constants expressed in exponential form is -3.4e38 to3.4e38

Character constant

- A character constant is a single alphabet,a single digit or a single special symbol enclosed within single inverted commas.Both the inverted commas point to the left.'A' is valid character constant whereas A is not.
- The maximum length of a character constant can be 1 character.

  Note: every charcter has its ASCII value.That means every charcter is interchage with integer constant. ex.'A' value is 65 and 'a' value is 97.

String constants

- A string constant is a sequence of characters enclosed in double quotes.The character may be letters, numbers,blank space or special characters.
- Single string constant "A" is not equalent to the single character constant 'A'.
- Each string constant must end with a special character'\0'.This character is called null character and used to terminate the string.The compiler automatically places a '\0' null character at the end of every string constant.

Escape sequance

- Some non-printing characters and some other characters such as double quote(“),single quote(‘),Question mark(?),and back slash(\),require an escape sequence.

A list of commanly used back slash character constant is given below:

Escape sequanc	Meaning	ASCII value	Escape sequenc	Meaning	ASCII value
\a	Bell	7	\r	Carriage return	13
\b	Back space	8	\”	Double quote	34
\t	Tab	9	\’	Single quote	39
\n	New line	10	\?	Question mark	63
\v	Vertical tab	11	\\	back slash	92
\f	Form feed	12	\o	null	0

Variables

- A variable can be considered as a name given to the location in memory.
- The term variable is used to denote any value that is refferred to a name instead of explicit value.
- A variable is able to hold different values during execution of a program,where as constant is restricted to just one value.
- Ex..2x+3y=10;since x and y can change,They are **variables** ,whereas 2 , 3 and 10 cannot change ,hence they are **constants.**The total equation is known as **Expression.**

Rules for constructing variable name

- They must begin with a letter.some systems permit underscore as the first character.
- No special characters other than letters,digits and underscore be used in variable name.
- Commas or blanks are not allowed with a variable name.

- Uppercase and Lowercase letters are significant.That is the variable name “income”is not same as “INCOME”.Some ex of such variable names are:average,height,total,counter_1.
- Ex .of variable declarations, int

  average;

  float height;

Data Types

- A data type define set of values and the operation that can be performed on them.
- There are three classes of datatypes here: • Primitive

data types

-int,float,double,char. •

Derived data types

-arrays name under this category

-arrays can contain collectin of int or float or char or double

- User dfined data types

-structures and enum fall under this category

Size and Range of datatypes

Data types	Description	Size(no.of.bytes)	Range
char	Single character	1	0 to 255
Int	An integer	2	-32768 to +32767
Float	Floating point number	4	-2147483648 to
Double	Floating point number	8	Approximately 15 digits of
Void	No data tyoe	0	
Signed char	Character	1	-128 to 127
Unsigned char	Unsigned character	1	0 to 255
Short signed int	Short signed integer	2	-32768 to 32767
Short unsigned int	Short unsigned	3	0 to 65535
Long signed int	Long signed integer	4	-2147483648 to +2147483647

CHAPTER 3

Operators and Expressions

- Types of Operators
- Arithmetic Expressions
- Evaluation of Expressions
- Precedence of Arithmetic Operators
- Type Conversions in Expressions
- Operator Precedence and Associativity.

Operators and Expressions

- An operator is a symbol that tells the computer to perform certain methametical or logical manipulations.
- Operators are used in program to manipulate data and variables.The data items that operators act upon are called operands.
- The operators are classified into unary,binary and ternary depending on whether they operate on one,two,or three operands respectively.

Types of Operators

- C has four classes of operators

  1.Arithmetic operators.
  2.Relational operators.
  3.Logical operators. 4.Bit-wise operators.

- C has some special operators,which are unique to c, they are

  1.Incr and Decr Operators. 2.Conditional Operators. 3.Assignment Operators.

Arithmetic Operators

- There are five Arithmetic Operators in c.
- The following table lists the Arithmetic operators allowed in C:

Operators	Meaning
+	Addition
_	Subtraction ;also for unary minus
*	Multiplication
/	Division
%	Modulo Division

Relational Operators

- Relational Operators are symbol that are used to test the relationships between two variables or between a variable and constant.we often compare two quantities,and depending on their relation takes certain decisions.These comparision can be done with the help of relational operators.
- C has six relational operators as given below.

Operator	Meaning
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
==	Equal to
!=	Not equal to

Logical Operators

- Logical operators are Symbols that are used to compine or negate expressions containing relational operators.
- C has three logical operators as defined below.

Operator	Meaning
&&	Logical AND
||	LOGICAL OR
!	LOGICAL NOT

Bitwise Operators

- The lowest logical element in the memory is bit.c allows the programmer to interact directly with the hardware of a particular system through bitwise operators and expression.
- Thease operator work only with int and char datatypes and cannot be used with float and double type .
- The following table shows the bitwise operators.

Operator	Meaning
.	One’s complement
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR
<<	Shift left
>>	Shift right

Increment or Decrement Operator

- C has two very useful for adding and subtracting a variable.These are ++,--.These two operators are unary operators.
- The Increment operator ++ adds 1 to its operand and the decrement operator – subtract 1 from its operand.Therefore ,the following are equalent operators.
- ++i is equalent to i=i+1. • --i

is equalent to i=i-1.

- Thease operators are very useful in loops

Assignment Operators

- In addition to useful Assignment operator =,c has a set of short hand operators that simplifies the coding of a certain type of assignment statement.
- It is of the form

Var op=exp

- Where var is a variable,op is a c binary arithmetic operator and exp is an expression

Shorthand Assignment Operators

Statement	Equalent statement
a+=b	a=a+b
a-=1	a=a-1
a*=n+1	a=a*(n+1)
a/=n+1	a=a/(n+1)
a%=b	a=a%b

Conditional Operator

- C provides a peculiar operator ?:which is useful in redusing the code.it is ternary operator requiring three operands.
- The general format is exp1?exp2:exp3;

  Where exp1,exp2,exp3 are expressions.
- In the above conditional expression,exp1 is evaluated first.if the value of exp1 is non zero(true),then the value returned will be exp2.if the value of exp1 is zero(false),then the value returned will be exp3.

Arithmetic Expressions

- An arithmetic expression is a combination of variables,constants and operators .

Algebraic Expression	C expression
ab-c	a*b-c
(m+n)(x+y)	(m+n)*(x+y)
(ab/c)	A*b/c
3x+2x+1	3*x+2*x+1

Evaluation of Expressions

- Expressions are evaluated using an assignment statement of the form:

  variable=expression;
- Given an integer variables a,b,c,d and where a=1,b=2,c=3 and d=4.
- Evaluate the following expressions: x=a*b-c;

  y=b/c*a; z=a-

  b/c+d;

Precedence of Arithmetic operators

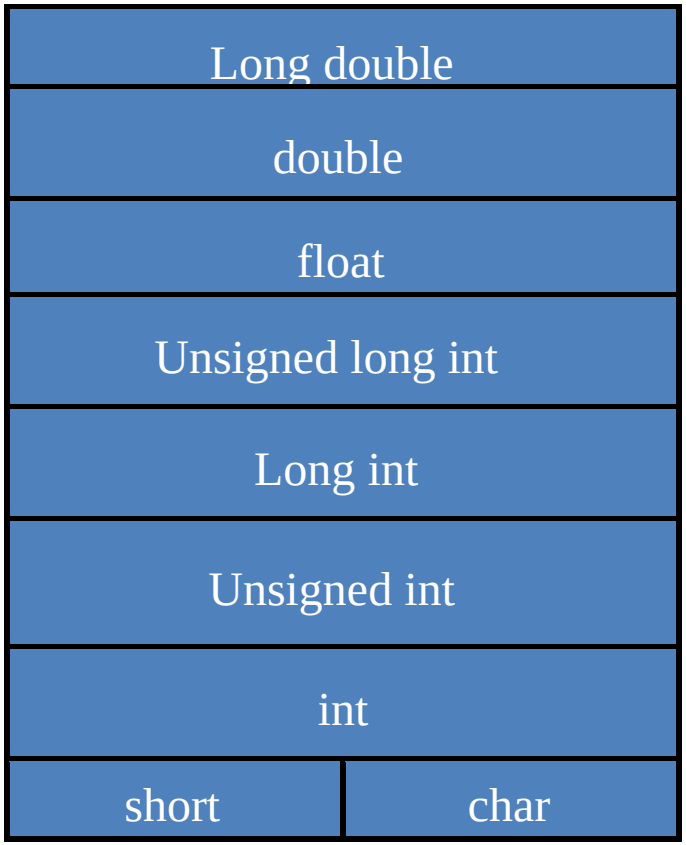
- The priority or precedence in which the operations of an arithmetic statement are performed is called the hierarchy (precedence) of operators.
- The operators of at the higher level of precedence are evaluated either from left to right or from right to left,depending on the level. This is known as associativity property of an operator

Operator	Description	Associativity	Rank
*	Multiplication	Left to right	3
/	Division	Left to right	3
%	Modulo	Left to right	3
+	Addition	Left to right	4
-	subtraction	Left to right	4

Type converssions

- There are two type of conversions in C. 1.Implicit type conversion 2.Explicit type conversion
- C performs automatic conversions of type in order to evaluate the expression. This is called implicit type conversion.
- In explicit type conversion we decide what type we want to convert the expression.
- Syntax of explicit type conversion is: (type) expression
- Where, **type** is any of the type we want to convert the expression into. Ex., x=(int)7.5., z=(double)sum/n.

Conversion hierarchy



Operator precedence and Associativity

- Operator precedence gives priorities to operators while evaluating an expression

- when 2*3+2 is evaluated output is 8 but not 12 because the * operator is having more priority than + hence 2*3 is evaluated first followed by 6+2

Operator precedence table

Operator precedence table gives the ***detail list of priorities for some operator*** •

Operators are listed from higher priority to lower

Precedence	Operator	Description
1	::	Scope resolution
2	++ --	Suffix/postfix incr and
2	*type*() *type*{}	Function-style
2	()	Function call
2	[]	Array subscripting
2	.	Element selection by
2	->	Element selection
3	++ --	Prefix incrand decr
3	+ -	Unary plus and minus

3	! ~	Logical NOT and bitwise NOT
3	(*type*)	C-style type cast

UNIT –II

MANAGING INPUT AND OUTPUT OPERATIONS

INTRODUCTION:

Three Essential Functions of a computer program

1. Reading of data
2. Processing of data
3. Writing of data

Most programs take some data as input and display the processed data that is known as result .

1.***Reading of data***

We have seen two methods of providing data to the program variables.

- One method is to assign value to variables through the assignment statement

```
x =5;
a=0;
```

- Another method is to use the input function ***scanf*** which can read data from a keyboard.

2.***Writing of data***

We have used extensively the function ***printf*** which sends results out to a terminal.

HEADER FILES:

All input/output operations are carried out through function calls such as printf and scanf .These functions are collectively known as the Standard I/O library.

The file name stdio.h is an abbreviation for standard input-output header file.The insrtruction ***#include <stdio.h>*** tells the compiler to search for a file named stdio.h and place its contents at this point in the program.the contents of the header file become part of the source code when it is compiled.

READING A CHARACTER

Reading a single character can be done by using the function ***getchar()***

•Variable_name is a valid C name that has been declared as char type.When the statement encountered,the computer waits until a key is pressed and then assigns this character as a value to getchar function.

• Getchar is used on the right-hand side of an assignment statement,the character value of getchar is in turn assigned to the variable name on the left.

char name;
name = getchar( );

Example Program:

```
#includ        e<stdio.h>
main( )
{
        char answer;
        printf("Would you like to knoe my name?\n");
        printf("Type Y for YES and N for NO:");
        answer = getchar();
        if(answer =='Y'  ||   answer == 'y')
                printf("\n\nMy name is BUSY BEE\n");
        else
        printf("\n\n You are good for nothing\n");
}
```

This program receives a character form the keyboard and tests whether it is a letter or digit and prints out a message accordingly.These tests are done with the help of the following functions:

isalpha(character)
isdigit(character)

• *isalpha* assumes a valu non-zero(TRUE) if the argument character contains an alphabet.otherwise it assumed 0(FALSE).Similar is the cae with the function isdigit.

```
#include<stdio.h>
#inclde<ctype.h>
main( )
{
        char character;
         printf("Press any key \n");
        character = getchar( );
         if(isalpha(character) > 0)
        printf("The character is a lettr.");
```

```
        else
                if(isdigit(character )  > 0)
                        printf("The character is a digit.");
                        else
                                printf("The character is not alphanumeric."):
}
```

WRITING A CHARACTER:

There is an analogous function ***putchar*** for writing character onen at a time to the terminal .

Where variable_name is a type char variable containing a character.

answer = 'Y';
Putchar(answer);

Will display the character Y on the screen.
Example program:

Uses of three new functions;

- ***islower( )***
- ***toupper ( )***
- ***tolower (***)

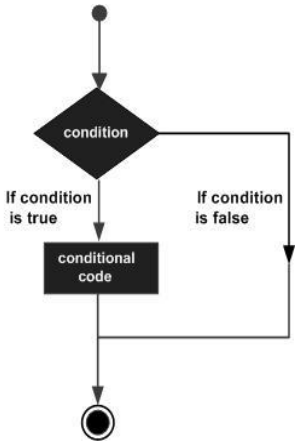
```
#Include<stdio.h>
#include<ctype.h>
        main()
        {
                        char alphabet;
                        printf("Enter an alphabet"):
                        putchar('\n');
                        alphabet = getchar( );
                        if(islower(alphabet))
                         putchar(toupper(alphabet));
                        else
                        putchar(tolowe(alphabet));
        }
```

CHAPTER – 5(DECISION MAKING AND BRANCHING)

Decision making structures require that the programmer specifies one or more conditions to be evaluated or tested by the program, along with a statement or

statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

Show below is the general form of a typical decision making structure found in most of the programming languages –



C language possesses such decision-making capabilities by supporting the following statement.

- If Statement
- Switch statement
- Conditional Operator statement
- Goto statement

These statement are popularly known as decision-making statement. Since these statement 'control' the flow of execution.

IF STATEMENT

An if statement consists of a Boolean expression followed by one or more statements.

If(test expression

If the Boolean expression evaluates to true, then the block of code inside the 'if' statement will be executed. If the Boolean expression evaluates to false, then the first set of code after the end of the 'if' statement (after the closing curly brace) will be executed.

C programming language assumes any non-zero and non-null values as true and if it is either zero or null, then it is assumed as false value.

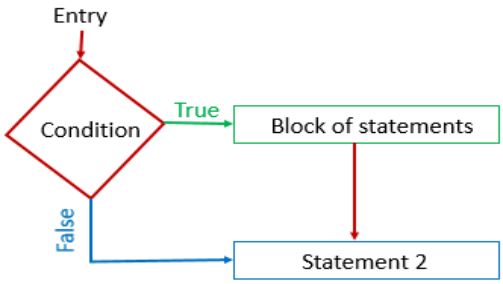
DIFFERENT FORMS OF IF STATEMENT

1. Simple **if** statement

2. **If…else** statement
3. Nested **if…else** statement
4. **else if** ladder.

1.***Simple if statement***

The general form of simple statement

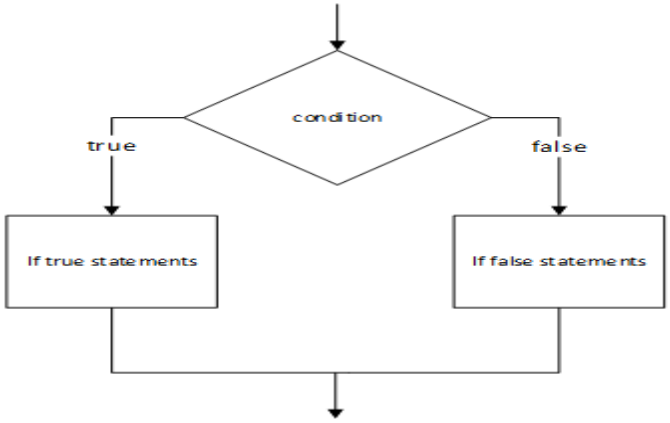



```
If(test expression)
{
        Statement – block;
}

Statement –x;
```

2.***if… else Statement***

The if…else statement is an extension of the simple if statement . The general form is




```
If(test expression)
  {
    True – block
statement
  }
Else
```

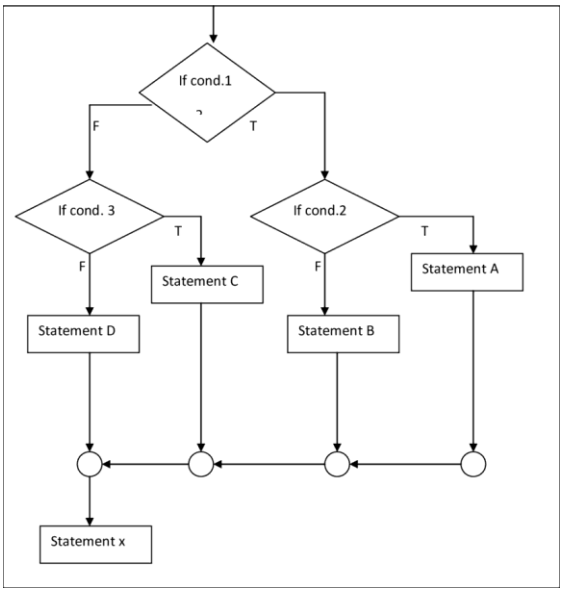
If the test expression is true ,then the *true-block statement* , immediately following the if statement are executed;

Otherwise, the *false-block statement* are executed. In either case , either *true-block or false-block* will be executed , not both .

3.***Nesting of if …else statement***

When a series of decisions are involved ,we may have to use more than one if…else statement in nested form .

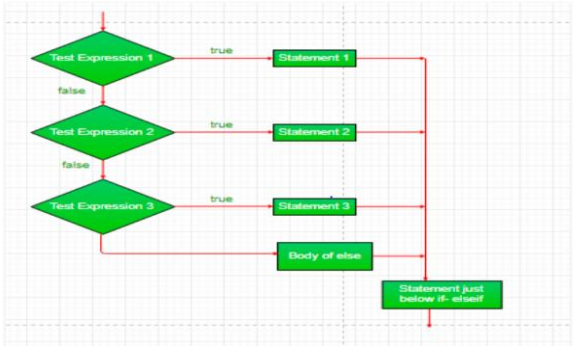
If the condition-1 is flase ,the statement-3 is executed.




```
If(test condition-1)
 {
    If(test condition-2)
     {
         Statement-1;
     }
    Else
     {
       Statement-2;
    }
 }
Else
 {
```

4.***The else if ladder***

There is another way of putting ifs together when multipath decisions are involved. A multipath decision is a chain of ifs in which the statement associated with each else is an if . it takes the following general form.

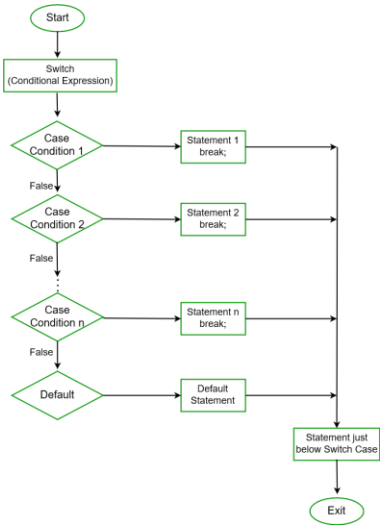



```
If ( condition 1)
        Statement-1;
Else if ( condition 2)
        Statement -2;
Else if ( condition 3)
        Statement – 3;
Else if ( condition n)
        Statement – n;
        Else
```

The if-else-if ladder statement executes one condition from multiple statements. The execution starts from top and checked for each if condition. The statement of if block will be executed which evaluates to be true. If none of the if condition evaluates to be true then the last else block is evaluated.

THE SWITCH STATEMENT

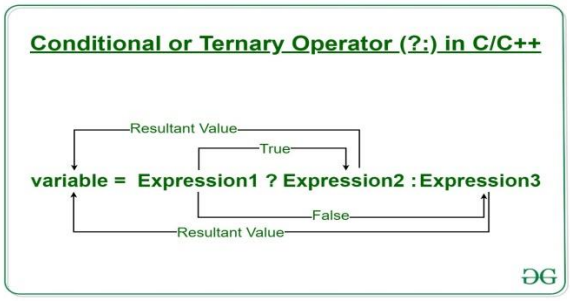
Switch statement is an alternative to long if-else-if ladders. The expression is checked for different cases and the one match is executed. break statement is used to move out of the switch. If the break is not used, the control will flow to all cases below it until break is found or switch comes to an end. There is default case (optional) at the end of switch, if none of the case matches then default case is executed.




```
Switch (expression)
{
   case value1: // statement
                sequence
                        break;
   case value2: // statement
                sequence
                        break;
                        .
```

THE ?: OPERATOR

The conditional operator is also known as a ternary operator. The conditional statements are the decision-making statements which depends upon the output of the expression. As conditional operator works on three operands, so it is also known as the ternary operator. ...



Conditional expression ? expression1 :

The conditional exprerssion is evaluated first.if the result is non-zero,expression-1 is evaluated and is returned as the value of the conditional expression.otherwise,expression2 is evaluated and its value is returned .

THE GOTO STATEMENT

The goto statement is a jump statement which is sometimes also referred to as unconditional jump statement. The goto statement can be used to jump from anywhere to anywhere within a function.

Forward jump Backward jump

CHAPTER-6(DECISION MAKING AND LOOPING)

A block of looping statements in C are executed for number of times until the condition becomes false. Loops are of 2 types: entry-controlled and exit-controlled. 'C' programming provides us 1) while 2) do-while and 3) for loop. For and while loop is entry-controlled loops.

Types of Loops in C

Depending upon the position of a control statement in a program, looping in C is classified into two types:

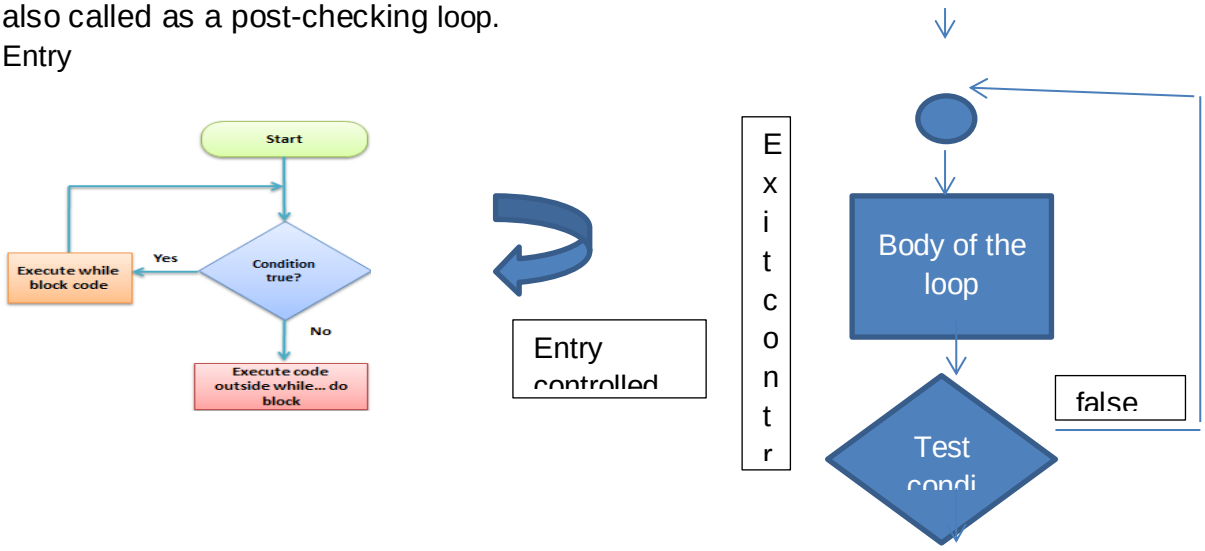
1. Entry controlled loop

2. Exit controlled loop

In an entry controlled loop, a condition is checked before executing the body of a loop. It is also called as a pre-checking loop.

In an exit controlled loop, a condition is checked after executing the body of a loop. It is also called as a post-checking loop.

Entry



The control conditions must be well defined and specified otherwise the loop will execute an infinite number of times. The loop that does not stop executing and processes the statements number of times is called as an infinite loop. An infinite loop is also called as an "Endless loop." Following are some characteristics of an infinite loop:

1. No termination condition is specified.

2. The specified conditions never meet.

The specified condition determines whether to execute the loop body or not.

'C' programming language provides us with ***three types of loop*** constructs:

1. ***The while loop***

2. The do-while loop

3. The for loop

1.While Loop in C

A while loop is the most straightforward looping structure. Syntax of while loop in C programming language is as follows:

```
while
(condition)
 {
```

It is an entry-controlled loop. In while loop, a condition is evaluated before processing a body of the loop. If a condition is true then and only then the body of a loop is executed. After the body of a loop is executed then control again goes back at the beginning, and the condition is checked if it is true, the same process is executed until the condition becomes false. Once the condition becomes false, the control goes out of the loop.

After exiting the loop, the control goes to the statements which are immediately after the loop. The body of a loop can contain more than one statement. If it contains only one statement, then the curly braces are not compulsory. It is a good practice though to use the curly braces even we have a single statement in the body.

In while loop, if the condition is not true, then the body of a loop will not be executed, not even once. It is different in do while loop which we will see shortly.

```
Sum = 0;
n = 1;
while (n <= 10)
{
      Sum = sum + n +n;
      n = n +1;
}
```

THE DO STATEMENT

The while loop construct makes a test ofcondition before the loop is executed. Therefore , the body of the loop may not be executed at all if condition is not satisfied at the very first attempt.

```
Do
{
    Body of the loop
}
While(test-condition);
```

```
I=1;
Sum = 0;
Do
{
          Sum = sum + 1;
           I=I+2;
}
While(sum<40 || I<10)
Printf("%d %d \n" , I ,sum);
```

THE FOR STATEMENT

The for loop is another entry-controlled loop that provides a more control structure.

```
For(   initialization ; test-condition ; increment)
{
      Body of the loop;
}
```

The execution of the for loop

1. initialization of the control variable is done first. Using assignment statement such as i = 1 and , count = 0. The variables i and count are known as loop-control variable.
2. The value of the control variable is tested using the test-condition .The test-condition is a relation expression.
3. When the body of the loop is executed. The control is transferred back to the for statement after evaluation the last statement in the loop

NESTING OF ***FOR LOOP***

Nesting of loop that is, one for statement within another for statement is allowed in C

```
For(  i = 1;   i  < 10;   ++ i )
{
       for(j   = 1;    j  !=5;   ++5)
                  {
                -------
               ---------
                  }
```

JUMP IN LOOP

Loops perform a set of operations repeatedly until the control variable fails to satisfy the test-condition.The number of times a loop is repeated is decided in advance and the test condition is written to achieve this.When executing a loop it becomes desirable to skip a part of the loop or to leave the loop as soon a certain occurs.

C permite a jump from one statement to another within a loop as well as a Jump out of a loop.

Jumping Out of a Loop

An early exit from a loop can be accomplished by using the break statement or the goto statement. When a break statement is encountered inside a loop is immediately exited and the program continues with the statement immediately following the loop when the loops are nested , the break would only exit the loop containing it.

That is, the break will exit only a single loop.

```
While(---)
{
        ----------
        -----------
        If(condition)
        Break;
        ------------
```

UNIT -III
ARRAYS

INTRODUCTION

We have used only the fundamental data types namely char, int, float, double, int and double. these types are very useful, they are Contained
By the fact that a variable of these types can store only one value at any given time at any given time that's so they can be used only to handle
Limited amount of data in many applications our we need to handle a large amount of data in terms of reading drafting and and printing
to process such large amounts of data we need a powerful data type that would facilitate efficient story accessing and manipulation.
data items free supports get derived data type known as Hare that can be used for such applications

An array is a fixed size frequency the collection of elements of the same data type it is simply a grouping of like type data in its simplest form an array can be used to represent a list of numbers or a list of names

Examples

where the concept of any array can be used list of temperatures recording every hour in a day or a month or a year

1. list of employees in an organisation
2.list of product and their cost sold by a store
3.test scores of a class of students
4.list of customers and the telephone number
5,.numbers table table of daily rainfall data.

provides a convenient picture of presenting data it is classified as one of the data structures in C other data structures include structure queues.

We introduce the concept of an array and entire array and discuss how to use it to create and apply the following types of arrays

- One-dimensional arrays
- Two-dimensional arrays
- Multidimensional arrays

Data Structures

C supports a rich set of derived and user- defined data types in addition to a variety of fundamental types as shown below

Data Types

Derived Types	Fundamental Type	User-Defined Type
-Arrays	-Integral types	-Structures
-Functions	-Float types	-Unions
	-Character types	-Enumerations

arrays and structures are referred to as structured data type because they can be used to represent data values that have a structure of some sort
Structured data type provide on organizational scheme that show the relationships among the individual elements and facilitate efficient data manipulations in programming partance,such data type are known as data structures

in addition to arrays and structures,C supports creation and manipulation of the following data structures

- Llinked lists
- Stacks
- Queues
- Trees

ONE-DIMENSIONAL ARRAYS

list of item can be given one variable name using only one subscript and such a variable is called ear single subscripted variable or a one dimensional array we often deal with variables thay are single subscripted

$$A=\sum_{I=1}^{n} x_i$$
$$n$$

to calculate the average of n values of x scripted variable x refers to the ith element of x In c single subscripted variables x can be Express as

x[1],x[2],x[3],.....x[n]

subscript can begin with numbers 0 that is
X[0]

is allowed for example if we want to represent a set of five numbers say(35,40,20,57,19) an array variable number then we flowers

Int number[5];

and the computer reserves five storage location as Shown below

the values to the array element can be assigned as follows
number[0];

```
number[0]  =  35;
number[1]  =  40;
number[2]  =  20;
number[3]  =  57;
number[4]  =  19;
```

this world causes the array number to store the values as shown below

number [0]	35
number [1]	40
number [2]	20
number [3]	57
number [4]	19

these elements may be used in programs just like any other variable for example the following are valid statement

```
a=number[0]   +10;
 number[4] = number[0] + number[2] ;
 number[2] =x[5]+y[10];
value[6] = number[i]*3;
```

This subscripts of an hour can be integer constants integer variables like I c perform on bounds checking and therefore, that the array indicates are within the declared limits

DECLARATION OF ONE-DIMENSIONAL ARRAYS

like any other variables must be declared before they are used so that the compiler can allocate space for them in memory the general form of Array declaration is

type variable-name[size];

the type specified as the type of elements that will be contained in the array such as int,float or char and the size indicate the maximum number of elements that can be stored inside the for example

float heightt [50];

declare the height to be on array containing 50 real element 0 to 49 are valid

Int group [10];

Declare the group as Sun are to contain maximum of an integer constants

1. Any reference to the array outside limits would not necessary cause an error It must result in program result

2.the size should be either your r numeric constant or symbolic constant

C language treats character string simply as array of characters The size of the character string represents the maximum number of characters that the string can be hold
for instant

Char name [10];

declare the name marcia character array string variable that can hold a maximum of 10 characters read the following string constant into the string variables name

";WELL DONE "

Each character of the Stringer is an element of the array name and is stored in the memory as follows

W
E
L
L
' '
D

O
N
E
'\O'

INITIALIZATION OF ONE-DIMENSIONAL ARRAYS

An array is declared Its elements must be initial they will contain garbage On array can be Initialized at ieither of the following stages

- at compile time
- at runtime

Compile Time Initialiszation

We can initialize The elements of arrayIn the same way as the ordinary variables are declaredWhen they are declared the general form of initialisation of arrays
type array-name [size]={ list of values}; the values in list are separated by commas for example this statement

Int number[3] = {0,0,0};

Will declare the variable number Ehsan array of size 3 and will zero of each element if then number of values in list less than the number of elements then only that many elements will be initialised the remaining elements will be set to 0 automatically
for instance

float total [5]={0,0,15,75,-10}

will initialize the first three elements 0,0 15.75 and -10.0 and the remaining two elements to zero the size may be omitted In such cases the compiler allocated enough space for all initialized elements

for example
the statement

int counter [ ] = {1,1,1,1};

will declare the counter array to contain four elements with initial values 1 this approach Words fine as long as initilaze every element In the array
character array may be initialized in a similar manager the statement

char name[ ] = {'j','o','h','n',\o};

declare the name to be an array of five characters initialized with the string "John "ending with the the null character we can assign the string literal directly as under

Char name [ ] = "John";

Compile time initialization and may be Partial the number of initializers may be less than the declared size the remaining elements are initialised to zero if the array type is numeric and NULL if the type is there for example

Int number [5]={10,20};

Will initialize the first two elements to 10 and 20 Respectively and the remaining elements 0 similarly the declaration

char city[5] = { B};

Will initialize the first element to B and the reminding four to NULL It is a good idea to declare the size explicit

Int number [3] = {10,20,30,40};

Will not work,It is illegal in c

Run Time Initialization

An array can be explicity initialzed at run timeThis approach is usually applied initialzing for lager array for example consider the following segment of a C program

```
 For (i=0;i<=100;i =i+1)
{
If i<50
sum[i]=0.0;
Else
sum[i]=0.0;
}
```

The first 50 elements of the array sum are Initialized to zero whilel the remaining 50 elements are initiaized to 1.0 at run time
we can also used to read functions such as scanf to Initialized an array
For example
the statement

```
Int x[3];
 scanf ("%d%d%d",&x[0],&[1],&x[2]);
```

Searching and Sorting

Searching and sorting are the two most frequent operations performed on arrays Computer scientists hava devised Several

Data structures and searching and sorting techniques that facilitate repid access to data stored in list.

Sorting is the process of arranging elements in the list according to their values ascending or descending order A sorted list is called ordered list.

Sorted list are especially important in list Searching because they facilitate rapid Search operation many sorting techniques are available the three simple and most important among them are

- Bubble sort
- Selection sort
- Insertion sort

order sorting techniques include Shell Sort, Merge sort Quick sort

searching in the process of finding the location of the specified element in a list the specified elements is often called the search key.

- Sequential search
- binary search

A detailedon these techniques is beyond the scope of this text.

TWO –DIMENSIONAL ARRAYS

The array variable that can store a list of values.there could be situations where a table of values with have to be stored.

The following data table which shows the value of sales of three items by fourr sales girl

the table contain yaar total of one values each line we can think of this table as consisting of four rows and columns each row represents.

The value of sales by the particular sales girl and each column represent the value of sales of the particular item we represent a particular value in a matrix by using two subscript such as cialis used to define Sacchi table of items by using two dimensional array

V[4][3]

type array_name[row _size][column_size];

That unlike most other languages with use one pair of parents column to separate array size see place each size in its own set of brackets

Ttwo dimensional array are stored in memory as with the single dimensional arrays each dimension of the array is Indexed from zero to its maximum size minus one; the first index selects the row and the second index select the column with in row

INITIALIZING TWO-DIMENSIONAL ARRAYS

One dimensional array two dimensional may be filled by following the declaration with stop initial value enclosed in braces
for example

```
inI  table[2][3] = {0,0,0,1,1,1,};
```

initializes the elements of the first to zero and the second row to one. The initialization is done row by row the above statement can be equaivalently written as

```
int table [2][3] ={{0,0,0}{1,1,1}};
```

we can also initialize a two dimensional array in the form of a matrix as shown below

```
int  table[2][3] ={
                    {0,0,0}
                    {1,1,1}
```

we need not specify the size of the first dimension that is the statement

```
int table [ ] [3] =
 {
        {0,0,0}
        {1,1,1}
};
```

if the values are missing in the initialized they are automatically set to zero for instance the statement

```
int table[2][3] = {
                    {1,1},
                    {2}
                    };
```

the element of each row is explicity initialized to zero while other elements are automatically initialized to zero.

MULTI-DIMENSIONAL ARRAYS

Array of three or more dimension the exact limit is determined by the compiler the general form of a multi-dimensional array is

```
type array _name [s1] [s2] [s3].…….[sm];
```

where s, is the size of the diamension

some example are

```
int survey [3] [5] [12];
float table [5] [4] [5] [3];
```

survey is a three dimensional array declared to contain 180 integer type elements
table is a four- dimensional array containing 300elements of floating point type

The array survey may represent a survey data of rainfall during the last three years from January to December inj five cities

DYNAMIC ARRAYS

we created arrays at compile time an array created at compile time by specifying size in the source code has a fixed size and time Cannot be modified at runtime

]the process of allocating memory at compile time is known as static memory allocation and the arrays that receive static memory allocation are called static arrays

Where we want to use on arrayr that can very greatly in size we must guess what will be the largest size ever needed and create the array accordingly A Difficult task in fact modern languages like C do not have this limitation

it is possible to allocate memory to run time this future is known as dynamic memory allocation and the array created at runtime are called dynamic arrays

CHARACTER ARRAYS AND STRINGS

INTRODUCTION

A string is a sequence of characters that is treated as a single data item we have used strings in a number of examples in the past

"Man is obviously made to think"

if we want to include a double quotes in the string to be printed then we may use it with blank space as shown below

" \"]n Man is obviously made to think.\"said pascal"

For example

Printf ("\"Well Done"\");

will output the string

"Well Done"

while the statement

Printf ("\"Well Done"\");

will output the string

Well Done

Character strings are often used to build meaningful and readable program the common operation performed on character strings in

- Reading and writing strings
- Comparing string together
- Copying one string to another

- Comparing string for equality
- Existing portion of a string

DECLARING AND INITIALIZING STRING VARIABLES

It allows us to represent string as character array. In c a string variable is any valid C variable name and is always declared as an array of character

The general form

```
char string _name[ size];
```

the size determines the number of characters in the string_name

some example

```
char city [10];
char name [30];
```

the compiler assigns a character string to a character array

like numeric array character array may be Initialized When they are declared C permit a character array to be Initialized in either of the following two forms

```
char city [9] = "  New York";
char city [9] =  {'N' ,'e', 'w',' ',' Y', 'o', 'r', 'k','\0'};
```

Permits us to initialize a characters array without specifying the number of elements in such cases the size of the array will be determined automatically based on the number of elements in initialized.

For example the statement

```
char string [ ] ={'G','O','O','D','\0'};
```

An array name Cannot be used as the left operation of an assignment operator

READING STRINGS FROM TERMINAL

Using scanf Function

The familiar input function **scanf** can be used %s format specification to read in a string of character

Example

```
char address[10]
scan("%s" , address);
```

the problem with the scanf function is that is termineted its input on the first white space it finds A white space includes blanks, tabs, carriage return form feeds and new lines

Therefore, if the following lion of text is type in at the terminal

NEW YORK

only the strong NEW will be read into the array address sInce the blank space after the word NEW will terminate the reading of string

The scanf function automatically terminates the string that is read with null character and therefore the character array should be large enough to hold the input string plus the null character

N	E	W	\0	?	?	?	?	?	?
0	1	2	3	4	5	6	7	8	9

note that unlike previous scanf calls.

if we want to read the enter line" New York "then we use two character arrays appropriate size that is

```
char adr1[5],adr[5];
scanf("%s %s",adr1,adr2);
```

with the line of text

NEW YORK

will assign the string "NEW" to adr1 and "YORK" adr2

we can also specify the field with using the form %ws in the scanf statement for reading a specified number of characters from the input string

Example

```
Scanf{';%ws",name);
```

two things may happen

1.The width w is equal to or greater than the number of characters typed in The entlire string will be stored in the string variable

2.The width w is less than the number of characters in the string The express characters will be turncated and left unread.

```
char name [10];
scanf("%$s",name);
```

R	A	M	\0	?	?	?	?	?	?
0	1	2	3	4	5	6	7	8	9

K	R	I	S	H	\0	?	?	?	?
0	1	2	3	4	5	6	7	8	9

Reading a Line of Text

We can seen just now that scanf with %s or %ws can read only string white spaces That is they cannot be used for reading a text containing more than one word C supports a format

specification known as the edit set conversions code %[ ] that can be used to read a line containing a variety of character including whitespaces

Recall that we have used this conversions code in
The program segment

```
char  line [80];
canf ("%[\n"]",line);
printf ("%s"',line);
```

We read a Lion of input from the keyboard and display the same on the screen.

Using getchar and gets Functions

To read single character from the terminal using the function getchar we can use this function

To read successive single character from the input and place them into a character array
An entire line of text can be ead and stored in an array

```
char ch;
ch = getchar( );
```

That the getchar function has no parameters

WRITING STRINGS TO SCREEN

Using printf Function

We have used extensively the printf function with %s format of print strings to the screen. The format %s can be used to display an array of character that is terminated by the null character.

for example

the statement

```
printf ("%s" , name);
```

can be used to display the entire contents of the array name.

we can also specify the precision with which the array is displayed.

For instance this specification

%10.4

indicates that the first four characters are to be printed in the field width of 10 columns

if we include the minus sign in the specification this string will be printed left justified

The printf on UNIX supports another nice further that allows for variable field width or precision

For instance

Printer("%*,*s\n", w, d, string);

print the first d characters of the string in the field width of w.

The future comes in handy for printing a sequence of characters.

Using putchar and puts Functions

getchar C supports another character handling function putchar to output the values of character variables

it takes the following form

```
char ch = 'A';
putchar (ch);
```

The function putchar requires one parameters. This statement is equivalents to

putchar ("%c", ch);

We have used putchar function in to write characters to the screen.

we can use this function repeatedly to output a string of characters store in an array using a loop.

Another and more convenient way of printing string values is to use the function puts declared in the header file <stdio.h> This is one parameter function and involved asc under

puts (str);

ARITHMETIC OPERATIONS ON CHARACTERS

C allows us to Manipulate characters the same way we do with number. whenever a character constant of character variable is an expression

it is automatically converted into an integer value by the system

To write a character in its interger represent representation, we may write it as an integer

For example

if the medicine uses the ASCII representation then

```
x = 'a';
printf("%d\n",x);
```

wikll display the number 97 on the screen

it is also possible to perform arithmetic operations on the character constants and variables for example

```
x = 'z'-1;
```

is a valid statement in a ASCII the value of 'z' is better 122 and therefore the statement will assign the value 121 to the variable x

we may also use character constants in relational expression For example, would test whether the character contained in the variable ch is an upper-case letter

we can convent a character digit to its equivalent integer value using the following relationships

where x is define as an integer variables and character contains the character digit

For example let us assume that the character constants that digit '7'

Then,

```
x=ASCII value of '7' – ASCII value of '0'
  =55-48
  =7
```

PUTTING STRINGS TOGETHER

Just as be cannot assign one string to another directy we cannot join two string together by the simple arithmetic addition

That is

the statement such as

```
string 3 = string1 + string2;
string2  = string1 + "hello"
```

are not valid The characters from string1 and string2 should be copied into the string3 one after the other
The size of the array string3 should be large enough to help the total characters.

The process of combining two string together is called concatenation.

The concatenation of three strings.These for loops are used to copy of the three string in the first loop the characters contained in the first_name the copied into the variable name until the null character is reached.

The null character is not copied instead it is replaced by a space by the assignment statement

```
name[i] = ' ';
```

the second_ name is copied into name, starting from the column just after the space created by the above statement.

That is achieved by the assignment statement

```
name [i+j+k] = second_name[j];
```

If first_name contains .4 characters, then the value of i at this point will be 4 and therefore the first character from second _name will be placed in the cell of name

in the same way the statement

```
name [i+J+K+2] = last_name[k];
```

is used to copy the character from last_name into the purpose locations of name

We place a null character to terminate the concatenated string name in the example

it is important to note the use of expressions i+J+1 and i+J+K+2.

COMPARISON OF TWO STRINGS

C does not permit the comparison of two string directly

Tthat is, the statements such as

```
If(name1 == name2)
If(name == "ABC")
```

it is therefore necessary to compare that two string to be e tested the character by character.

The comparison is done until there is a mismatch or one of the string terminates into a null character occurs first, the following segment of program.

STRING – HANDLING FUNCTIONS

The C library supports a large number of string - handling functions that can be used to carry out many of the string manipulations so for following the first most commonly used string _handling functions.

Strcat() Function

The strcat function joins two strings together It takes the following form

strcat(string1, string2);

string1 and string2 are character arrays.When the function strcat is executed

string2 is appended to string 1. it does so by removing the null character at the end of string 1 and placing string 2 from there.

The string at string 2 remains unchanged

for example consider the following three things

We make sure that the size of string 1 is large enough to the final string.

strcat function may also a string constant to your string variable the following is valid

strcat (part1,"Good");

C permits nesting of strcar functions for example the statement

strcat(strcat (string1, string2),string3) ;

is allowed and concatenates all the three string together result String is stored in string 1

strcmp() Function

the strcmp function compares two string identified by the arguments and value 0 if they are equal if they are not numeric difference between the first nonmatching characters in the string it takes the form

strcmp(string1, string2);

string 1and string 2 may be string variables are string constants

Examples

```
strcmp(name1, name2);
strcmp(name1, "John");
strcmp("Rom", "Ram");
```

To determine whether the string or equal is not which is alphabetically above the value of the mismatch is important for example

strcmp("their", "there");

will return a value of -9 which is the numeric difference between ASCII and that is minus y is -9 if the value is negative string one is alphabetically above string2.

strcpy() Function

strcpy function words almost like a string assignment operator it takes the form

strcpyt(string1, string2);

and assign the constents of string2 to string1.string2 may be a character array variable or string constant for example the statement

strcpyt(city, “DELHI”);

will assign the string “DELHI” to the string variable city the statement
strcpyt(city1, city2);

will assign the contents of the string variables city2 to the string variable city1

The size of the array city1 should be large enough to receive the contents of city2

strlen() Function

This function counts and return the number of characters in a string it take the form

n = strlen(string) ;

where n is an integer variable which receives the value of the length of the string.

The argument may be a string constant the counting end at the first null character

since they are not equal they are joined together and copied into S3 using the statement

the program output all three string with their lengths

Other String Functions

The header file< string.h> contains many more string manipulation function they must be useful in certain situation

Strncpy

in addition to the function **Strcpy** that copies one string to anotherWe have another function **Strncpy t**hat copies only the left most n characters of the source string to the target string variable

That is

strncpy (s1,s2,5);

this statement copies the first 5 characters of the source string2 into the target string s1

first 5 character may not include the determining null character ww have to place it explicity in the positions of s2 as shown below

s1[6] = '\0';

the string S1 contains

strncmp

A variable of the function strncmp is the function strncmp

This function has three parameters as the function call below

strncmp (s1,s2, n);

this compares the left-most characters of s1 to s2 and returns

- if they are equal ;
- negative numbers if s1 sub-string is less than S2 and
- positive numbers

strncat

this is another concatenation function that takes three parameters as shown below

Tthis call will concatenation the left-most n characters of s2to the end of s1

strstr

It is a two parameter function that can be this used to locate a substring in a string

strstr (s1,s2);
strstr (s1,"ABC");

The function strstr searches the string and s1 to see whether the string s2 is contained in s1

function returns the position of the first concatenation first occur of their it returns null pointer

Example

```
if strstr (s1,s2,) = = NULL)
print (substring is not found");
else
print ("s2 is substring of s1");
```

we also have functions to determine the extension of a character in a string the function call

strchr(s1,'n');

will locate the first appearance of the character and the call
strchr(s1,'n');

will locate the last occurrence of character in the string S1

TABLE OF STRINGS

Use list of characters string such a list of the names of students list of the names of employees in an organisation list of places list of names can be treated as a table of 3 and their to diamonds so centre list

for example character **student [30]** May be used to store list of 30 name each of length not more than 15 characters shown below table of 5 this

This table can be conveniently stored in a character

```
Char city [] {}
{
"chandigarh"
"Madras"
" Hyderabad"
" Mumbai"
"Bombay";
};
```

To access the name of the city in the list we write

City [i-1]

and therefore city [0] "Chandigarh" ,city [1] denotes "Madras" and so on

That once on store the list of string each string is read using scanner

Function with %s format if any string contain white space the part of the string after the right a space will be as another list in the described by the should read the entire list lion as the string.