Unit 1.

Date

Problem :

Any question or matter involving doubt, uncertainty or difficulty.

It is a task, a situation which is difficult to deal with.

A problem is question proposed for solution.

☆ * Steps involved in problem solving :

i. Identify the problem.
ii. Understand the problem. - Gather the information.
iii. Finding alternative ways to solve the problem.
iv. Selecting one best way to solve the problem from the list of alternative solutions.
v. Implementation
   - List of Instructions which will help to solve the problem.
vi. Evaluate the solution.

Thursday
24-08-2023

☆ Types of problems.

1. Problem with algorithmic solution.
2. Problem with heuristic solution.

1. Problem with algorithmic solution.

Some problems can be solved with a series of actions. These sol^n's are called as algorithmic solution. And, these series of actions or steps are called as Algorithm.

Examples : Making Tea, Admission process, etc.

2. Problems with heuristic solution

Some problems are not straight forward, these soln's require reasoning built on knowledge and experience and a process of trial and error.
Soln's that cannot be reached through a direct set of step are called heuristic solutions.

Examples : Expansion of company, Doing Surgery, ~~etc~~.
working for a startup, etc.

* Problem solving with computers

Solution : It is the set of instructions given in step 5 of problem solving.

Result : It is the outcome or completed computer assisted answer.

Program : It is a set of instructions used to solve the problem.

Computers are useful in solving problems with algorithmic solution.
People are better than computers at developing heuristic solutions.
The field of computers that deals with heuristic types of problems is called ~~Arit~~ Artificial intelligence.

AI enables a computer to do things like build its own knowledge band and speak in a human language. As a result the computers problem solving abilities are similar to those of a human being.

AI is an expanding computer field specially with the increased use of robotics.
Until computers can be built to think like humans people will process most heuristic solutions and computers will process any algorithmic solutions.

Difficulty with problem solving.

Some of the common difficulties people may encounter while solving problems:

1. Lack of information
2. Limited perspective
3. Cognitive Biases
4. Emotional Barriers
5. Lack of creativity
6. Ineffective communication
7. Overthinking.
8. Impatience
9. Lack of proper planning.
10. Resistance to change
11. Groupthink
12. Unrealistic expectations.
13. Lack of problem solving skills.

Page No.
Date

Problem solving aspects

Phases of problem solving.

a. Problem definition phase.
b. Getting started on a problem.
c. Use of specific examples.
d. Working backwards from the solution.

Ex. In some cases we assume that we already have solution and then try to work backwards

General problem solving by divide and conquer.

☆ Algorithm

A set of instruction which is used to ~~per~~ solve the problem or accomplishing a task.

Algorithm is a set of ordered instructions which are written in simple english language.

Characteristics of Algorithm

1. ~~Sh~~ It should accept zero or more inputs.
2. It must generate some output
3. Each instruction should be clear and easy to understd.
4. Each instruct^n should have proper meaning and effectiveness to produce the final result.
5. Algorithm must terminate after fixed number of steps.

⊙ Building blocks of Algorithm

1. Statements : It can be ~~inp~~
   i. Input ~~stmt~~. statement.

ii. Output statement
iii. Processing operations.
iv. Conditional statement.

State: The state of an algorithm at a given point in its execution is the collection of values of all variables contained in the algorithm at that point.

→ Control flow: The process of executing the individual statements in a given order is called controlled flow.

The control can be executed in three ways:
i. sequence
ii. selection
iii. Iteration.

i. Sequence

All the instructions are executed one after another is called sequence execution.

Example:

Task: To add two numbers.

step1: start
step 2: (Aim) Addition of two numbers. (c = a+b)
step 3: Taking two numbers & a and b
step 4: Display the value of c by adding a & b
step 5: End.

ii. Selection

A selection statement causes the program control to be transformed to a specific part of the program based upon the condition.

Example :

Task: To decide whether the given number is positive or negative.

Step 1 : Start
Step 2 : Input from user
Step 3 : Check ~~ether~~ whether number is greater than or less than zero.
Step 4 : If $n>0$ then print positive.
Step 5 : If $n<0$ then print negative.
Step 6 : End.

Task : To decide ~~w~~whether given number is divisible by 3 and 5.

Step 1 : start.
Step 2 : Input from user.
Step 3 : ~~If check if the number has 0 and 5 of its unit place.~~
Step 3 : ~~Print that the~~ Finding the least common multiple i.e. 15 of 3 & 5.
Step 4 : Check ~~wh~~ whether the number is divisible by that LCM or not.
Step 5 : If yes then print yes that it is divisible by 3 and 5.
Step 6 : End.

Task : To decide whether given number is divisible by 3 and 5 .

Task : To decide ~~whe~~whether the number is odd or even .

step 1 : start .
step 2 : Input by User , x .
step 3 : Dividing the input by 2 and checking the remainder .
step 4 : If the remainder is zero print even .
step 5 : If the remainder is not zero then print odd .
step 6 : End .

Tuesday.

Page No.

Date 29 08 2023

iii. Iteration

In some programs, certain ~~state~~ set of statements are executed again and again based upon conditional test i.e. executed more than one ~~kind~~ time, this type of execution is called ITERATION or LOOPING.

→ Task: To display all natural numbers upto n.

Step 1: Start
Step 2: Get value of n
Step 3: Let i = 1
Step 4: if (i<=n) then go to step 5 otherwise go to step 7.
Step 5: Print i and increase i by 1.
Step 6: Go to step 4.
Step 7: stop.

n	i	o/p
4	~~1~~	4 ↵
	~~2~~	1 2 3 4
	~~3~~	
	~~4~~	
	5	

→ Task: finding summation of all natural numbers upto n.

Step 1: Start
Step 2: Get value of n
Step 3: Let i = 1, sum = 0 (initially)
step 4: if (i<=n) then go to Step 5 otherwise go to Step 7.

Page No.
Date

Step 5 : Put sum+i and increase i by 1.
Step 6 : Go to step 4
Step 7 : Print sum
Step 8 : end.

n	i	sum	o/p
4	~~1~~	0	4
	~~2~~	1	
	~~3~~	3	10
	~~4~~	6	
	5	10	

* Task : Find summation of all natural numbers from 11 to 15.

Step 1 : start
Step 2 : Put value of n = 15.
Step 3 : Let i = 11, sum = 0
Step 4 : if (i<=n) then go to step 5 otherwise go to step 7.
Step 5 : summation = sum + i, increase i by 1.
step 6 : Go to step 4.
Step 7 : Print summation.
step 8 : End.

n	i	sum	o/p
15	~~11~~	0	4
	~~12~~	23	
	~~13~~	36	~~81~~ 65
	~~14~~	50	
	~~15~~	65	
	16	~~81~~	

Page No.
Date

* Task : Find summation of n given numbers.

Step 1 : Start
Step 2 : Get value of n.
Step 3 : Let i=1 Sum=0
step 4 : if (i<=n) then go to step 5 otherwise go to step 9
step 5 : Take a value for x.
Step 6 : sum = sum + x
Step 7 : i = i+1
step 8 : Go to step 4.
Step 9 : Print ~~Sum~~ value of sum
Step 10 : End

n	i	sum	x	O/P
4	~~1~~	~~0~~	~~20~~	4 ↵
	~~2~~	~~20~~	~~25~~	20 ↵
	~~3~~	~~45~~	~~30~~	25 ↵
	~~4~~	~~75~~	13	30 ↵
	5	88		138 ↵
				88

* Function.

Function is a subprogram which consist of a block of 4 (set of instruction) that performs a particular task.

Benefits of Using functions :

i. It reduces complexity of programs.
ii. Reduces no. of line in program.
iii. Code reuse.

iv. Easy to debug and test.
v. Easy to maintain.

Write algorithm for addition of two numbers using function

main function ().

step 1 : start.
step 2 : call the function add.
step 3 : stop.

Subfunction add ().

step 1 : Function start.
step 2 : Get values of a and b
step 3 : Add c = a+b
step 4 : Print c.
step 5 : Return.

flowchart.

Flow chart is the graphical representation of the logic for problem solving.

Symbols used to draw flow chart.


flow lines


Processing


start/stop


decision / condition

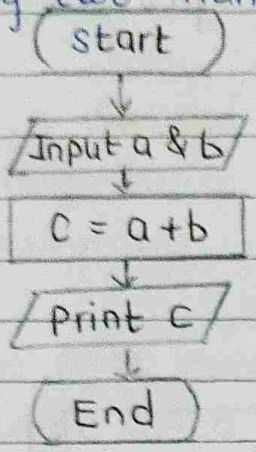

Input / Output


connector

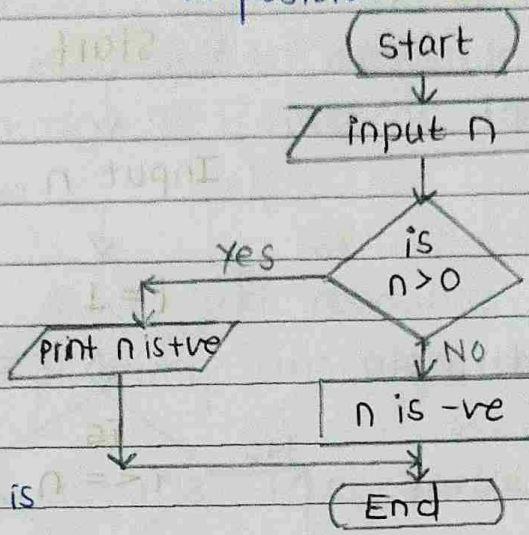

subtraction.

Page No.
Date

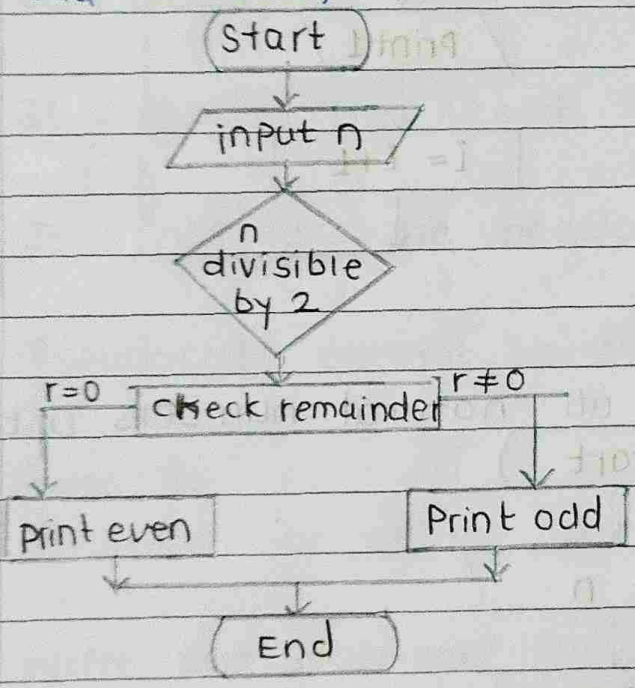
Adding two numbers.



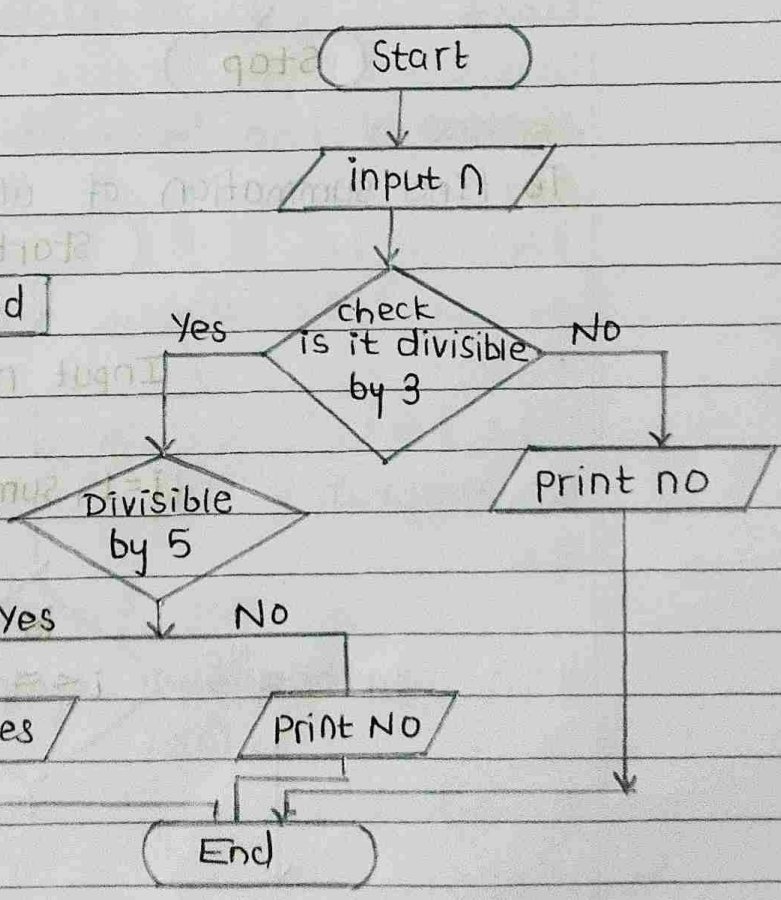
Decide whether the given number is positive and negative.



Decide whether number is odd or even



Decide whether given number is divisible by 3 and 5.

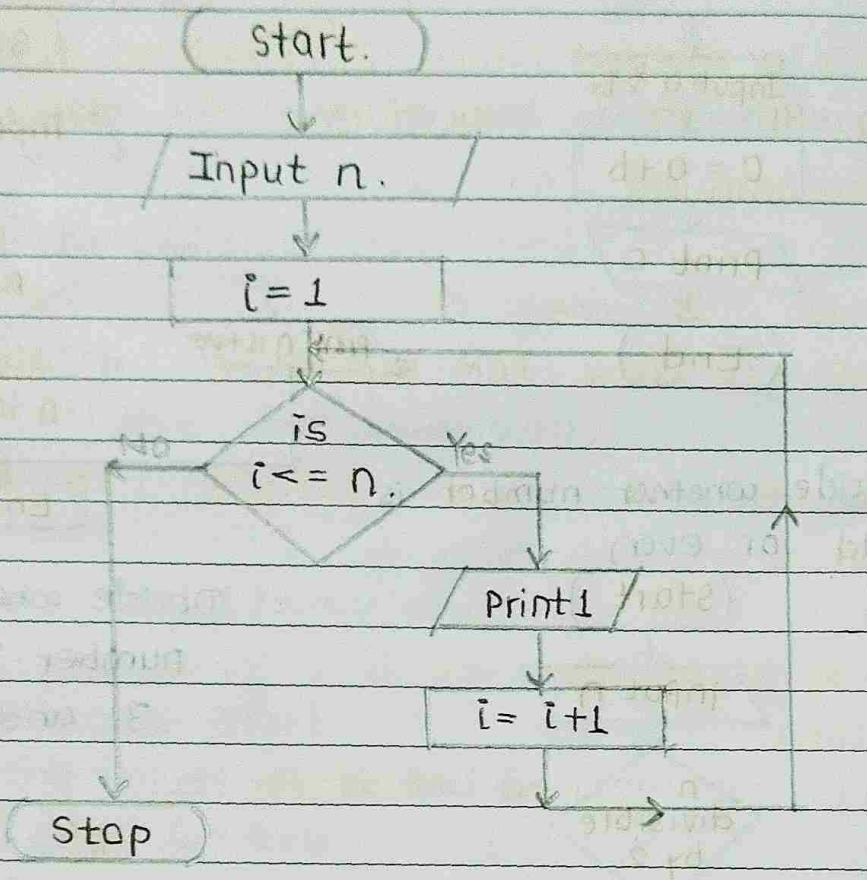


(Friday).

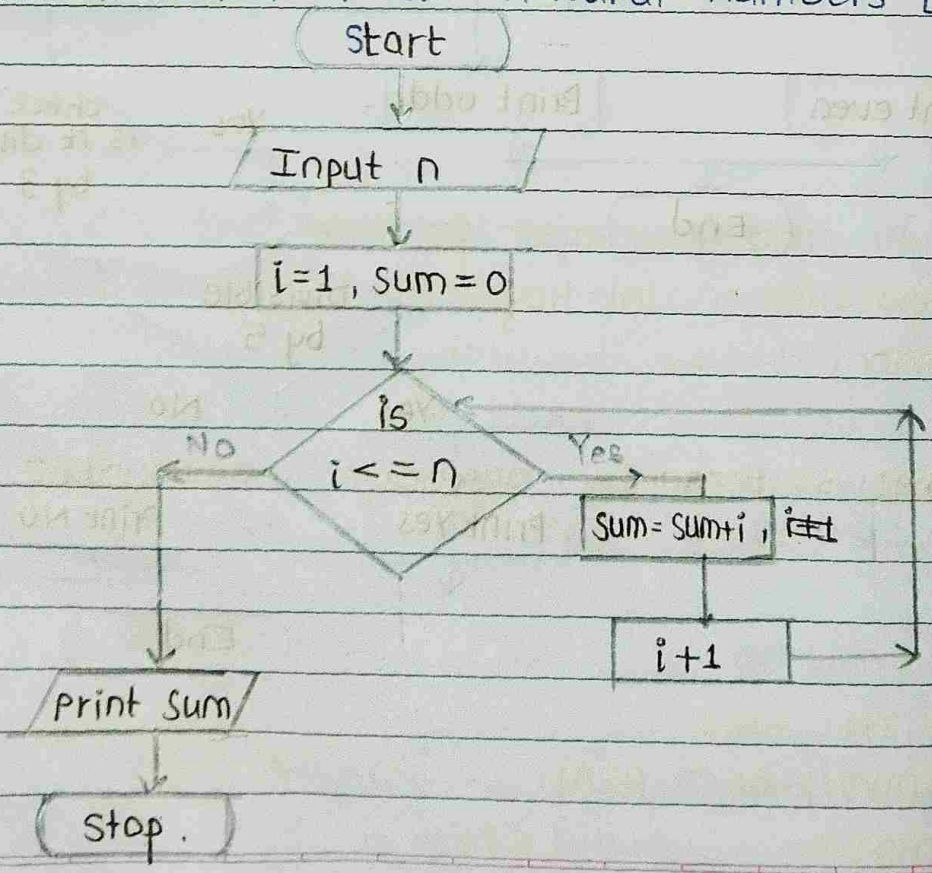
Page No.

Date 01 09 2023

To display all natural numbers upto N.



To find summation of all natural numbers upto n.



Pseudocode

It is a method which is used to describle computer algorithm to a combination of natural language, and programming language.

Pseudo code consist of short and readable english language used for explaining an algorithm.

It does not include details like variable, declaration, functions, etc.

It gives us the sketch of the program before actual coding.
It is not machine readable.

Pseudocode cannot be compiled and executed.

~~How to~~ Guidelines for writing pseudo code.

i. Write one statement per line.
ii. Capatalise initial keyword.
iii. indent to ~~be~~ hierarchy.
iv. End multiline structure.
v. Keep statements language independent.

Keywords for pseudocode.

BEGIN, END
INPUT, GET READ
COMPUTE, CALCULATE

ADD, SUBTRACT, INITIALIZE

OUTPUT: PRINT, DISPLAY

IF, ELSE, ENDIF

WHILE, ENDWHILE

FOR, ENDFOR.

Code: Addition of two numbers.

```
BEGIN
GET a, b
ADD  C = a+b
PRINT C
END.
```

Code:

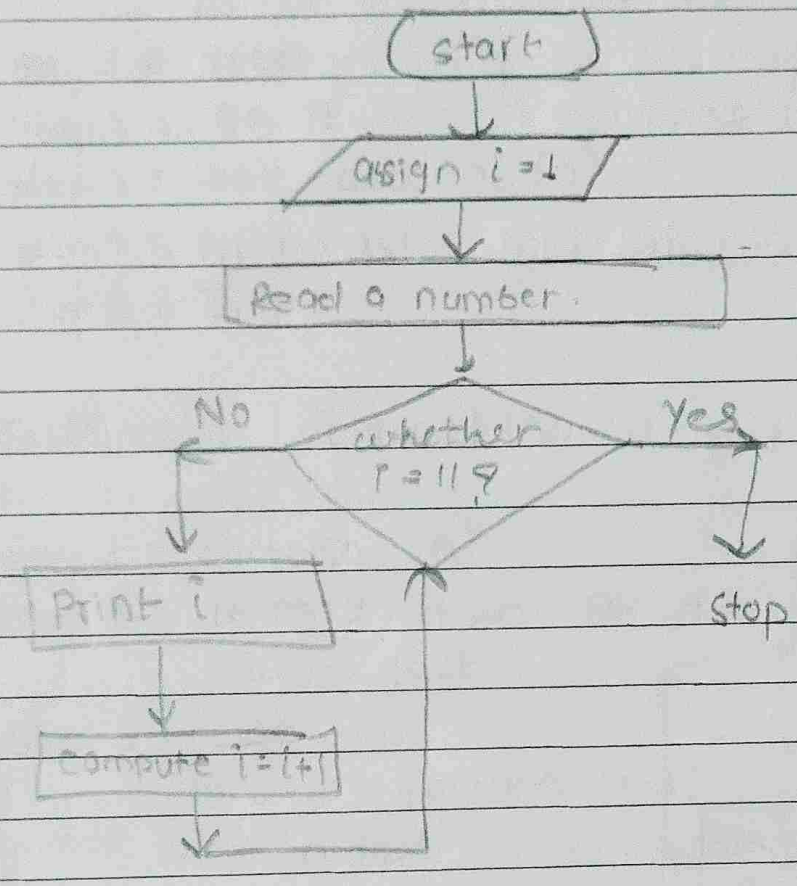
```
BEGIN
GET n
INITIALIZE  i = 1
WHILE  (i <= n) DO
PRINT  i
  i = i + 1
ENDWHILE
END
```

Task: To display multiplication table for given numb

→ Algorithm

```
BEGIN.
Get n.
Initialize  i = 1
Print  i = 1
```

increase i by 1 till i <= 10.
Print i ten times where i = 0 to 10.



05/09/2023 Simple strategies for developing algorithm.

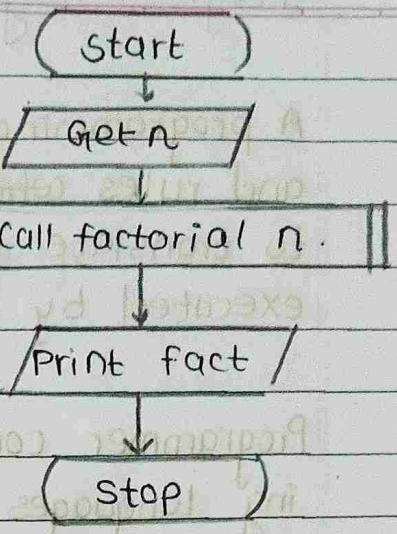
Iteration
Recursion.

* ~~Main~~ Recursion
A function that ~~cause~~ calls itself is called recursion.

Recursion is a process by which a function calls itself repeatedly until some specified condition has been satisfied.

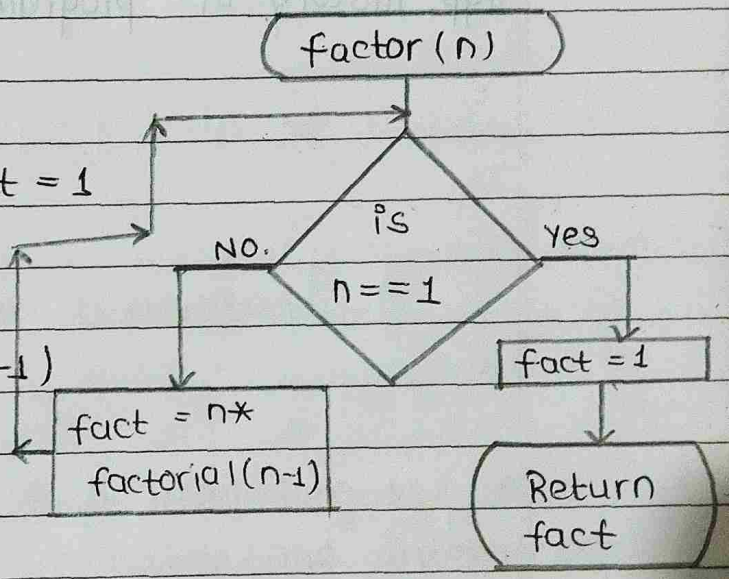
Main function

step 1 : start
step 2 : Get n.
step 3 : call factorial (n)
step 4 : print fact
step 5 : stop



SubFunction factorial (n).

step 1 : factorial (n).
step 2 : if n==1 then fact = 1
return fact.
step 3 : else
fact = n * factorial (n-1)
return fact



Unit 2.

Programming Languages.

Page No.
Date 05 09 2023

A programming language is a set of symbols, grammer, and rules ~~which~~ with the help of which one is able to translate algorithms in programs that will be executed by the computer.

Programmer communicates with a machine using programming languages.

~~Typ~~ History of programming languages.

Page No.
Date

Types of programming languages.

1. Low level language / Machine level language.

   Eg. Binary language.

2. Middle level language.

   Easier than low level language.
   Eg. Assembly language.

3. High level language.

   More easy to read, write and understand.
   Eg. C, C++, Java, Python.

Translators which translate high level language into machine level language are compiler and Interpreter.

06/09/2023. ## Another category of programming language.

1. Procedural Programming language. Ex. C
2. Object-oriented programming language, Ex. C++, Java, Python.

It

1. Procedural PRogramming language.

It is defined as a programming language derived from the structure programming and based on following procedures (functions).

It follows a step by step approach in order to breakdow a task into a set of variables and functions via. a sequence of instructions.

During the program execution a procedure can be called at any time, at any point either by itself o by other procedures.

Ex. C, Pascal, fortran...

2. Object oriented programming language.

The programming where everything is represented as object is called as object oriented programming language.

Object is anything having a set of properties.

Basic features of OOP are abstraction, class, object, encapsulation, inheritance.

We can provide solution to the real world problems usin OOP language.

Ex. C++, Java, Python.

→ C - language.

C is a general purpose high level language, that was originally developed by Dennis Ritche ~~and~~ ~~P~~ at Bell labs in 1972.

features of C - language,

i. Easy to learn.
ii. It is structured language.
iii. It produces efficient programs,
iv. It can handle low level activities,
v. It can be compiled on a variety of computer platforms.

~~Where~~ C language can be used to develop :

i. Operating System.
ii. Language compiler.
iii. Language interpreteur.
iv. Text editor.
v. Network drivers.
vi. Data bases,

Parts of C program :

i. Preprocesser.
ii. functions.
iii. Variables
iv. statements
v. expressions.
vi. Comments,

// Program to add two number.

Comment just for user not for compiler

```
# include <stdio.h>
int main ()
{
  int a,b,C ;
  a=5 ;
  b = 25 ;
  C = a+b ;
Printf ("C = %d", C) ;
return 0 ;
}
```

header file

standard i/p or o/p.

Parameters.

declaration.

Assignment statement as value is assigned

→ Comments.

// single line comment

/* .... } multiple line comment.
.... */ }

→ Variables

It is the name of a memory location that we use for storing data, its value may change during the execution of program.

→ Statement

suppose here a = 5 ;

a = b ;

a = b+c ;

a = abc () → function

→ Preprocessor

It is the part which will get processed before actual processing of the program.

friday.

Page No.

Date 08 09 2023

→ Function

Min. no of function in C program is one.

→ Expressions.

Combination of operators and operants.

Eg. a+b (a, b: operants; +: operator)

+, -, x, ÷ Arithmetic operator, <>, <, >, ≤, ≥, ≠, #, == relational operator, logical operator, assignment operator.

Relational operator.

>, <, >=, <=, =, =, <>.

Logical operator.

AND, OR, NOT

* Token.

A token is defined as the smallest individual unit present in the program.

A compiler breaks a program into tokens and then proceeds ahead to the next stages used in compilation process.

* 1. Keyword

These are predefined reserved words used in programming that have special meanings to the compiler.
Keywords are part of syntax and they cannot be used as an identifier.
Eg. IF, WHILE, DO, etc.

2. Identifier

- It is a name given to any program elements as variables, functions, etc.
- Idenfiers must be unique.
- They are created to give a unique name to an entity to identify it during the program execution.

Rule : variable + number
only one special character (_) underscore.

3. Constant

It is a named data ~~atom~~ item with a predefined value.

You cannot change the value assigned to a predefined constant.

It can be any data type.

i. Literal constant : It is a value such as a number character or ~~strea~~ string that ma be assigned to a variable.

eg. `a='5';` `a='H';` `a='PUNE'`

Here constant have same name and value

ii. Symbolic constant : It is a constant when you gi it a ~~n~~ name.

eg. `const Pi = 3.142`

Here constant have different name and different values.

4. String

It is a sequence of characters terminated with a null character / '0'.

It can contain letters, numbers, symbols and spaces. It must be enclosed in quotes. (' ') (" ").

5. Variable.

Variables are containers for storing data values like numbers and characters.

It is the name of a memory location that we use for storing data.

Variable is a named data item whose value can change during the program execution.

Structure of programming languages.

A programming language is a set of instructions and syntax used to create software programs.

some of the key features of programming languages:

i. Syntax
   - The specific rules and structure used to write code in a programming language.

ii. Data types
   - The type of values that can be stored in a program such as numbers, strings, characters,

iii. Operators.
   - Symbols used to perform operations on values such as addition, subtraction.

iv. control statement

- Statements ~~su~~ used to control the flow of a program such as if-else statements, loops and function calls.

v. Libraries

- collections of ~~free~~ pre written code that can be used to perform common task and speedup development.

* + Operators.

→ 1. Arithmetic operators.

		Description.
i.	+	Addition of operands.
ii.	−	Subtraction of operants.
iii.	x	product of operands
iv.	÷	Division of operands.
v.	% (modulus)	It preturns remainder after the division of two ope
~~vi.~~		

→ 2. Comparison (or) Relational operator.

=	equal to
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to
!=	not equal to

→ 3. Logical operators.

AND	if both side of the operator evaluates to be true then result is true otherwise it is false.
OR	if any one of the side of the operator evaluates to be true then the result is true otherwise false.
NOT	if operand is true ~~result~~ it returns false and vice versa.

→ 4. Bidwise operators.

& (and)	Bidwise anding of two specified operations are done.
| (or)	Bidwise oring of two specified operations are done.
^ (xor)	Bidwise x-oring of two specified operations is done.
<< (binary left side)	The operand will be left shifted.
>> (binary right side)	The operand will be right shifted.

Page No.
Date

Revision.

How to solve problem

Unit III :

Control flow and loop.

* If-else statement

explanation with example

If else statement in C-programming is a conditional statement that execute a different set of specifier based on the condition i.e. true or false. The 'if' block will be executed only when the specified condition is true and if the specified condition is false then the else block will be executed.

```
Syntax :    if (expression)
            {
              // statement ;
            }
            else
            {
              // statement ;
            }
```

* Nested if-else statement

A situation comes when we have to check if condition inside an 'if-else' statement then a term name term comes Nested if-else statement.

It means if-else statement inside another one 'if' statement.

Syntax : if (expression)

```
if (condition 1)
{
 if(condition 2)
 {
 // statement ;
 }
 else
 {
   //statement ;
 }
```

→ C-programming (beyond the syllabus)

In C-language print f function [PrintF()] is used to print output on the screen.
This function is a part of the c standard library "stdio.h"

In C-programming language scanf() is a function that stands for scan formatted ~~strea~~ string. It is used to read data from standard input ~~output~~ (stdin) string, the syntax of scanf() in C ~~langug~~ language is similar to the syntax of printf().

%d = for printing integers.
%f = for printing floating point values or numbers.

%c = for printing characters.
%s = for printing strings
%p = for printing memory addresses.
%x = for printing hexadecimal number.

C-program

- To display area of triangle, rectangle and circle.

```
#include<stdio.h> in
int main ()
{
   int a,b,h,l,w,r ;
   b = base
   h = hei
```

must

```
# include <stdio.h>
# include <math.h>

triangle area (b,h)
rectangle area (l,w)
circle area (r)
```

Thursday.

Page No.

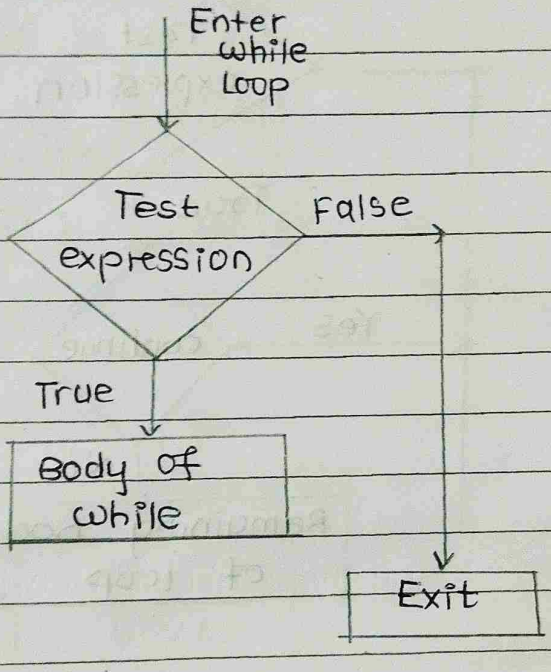
Date 05 10 2023

→ Statements. (All explanation, syntax, example, flowchart).

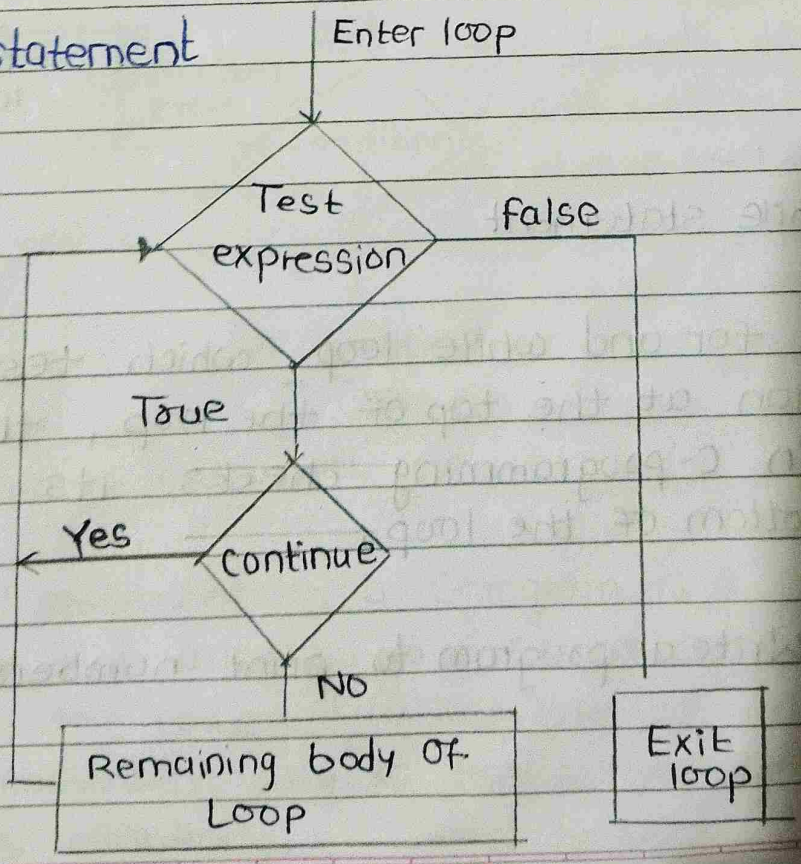
While statement.

It is the conditional looping statement and in this action block, while statement gets executed still the condition of while statement is satisfied.

Flowchart :



Break statement

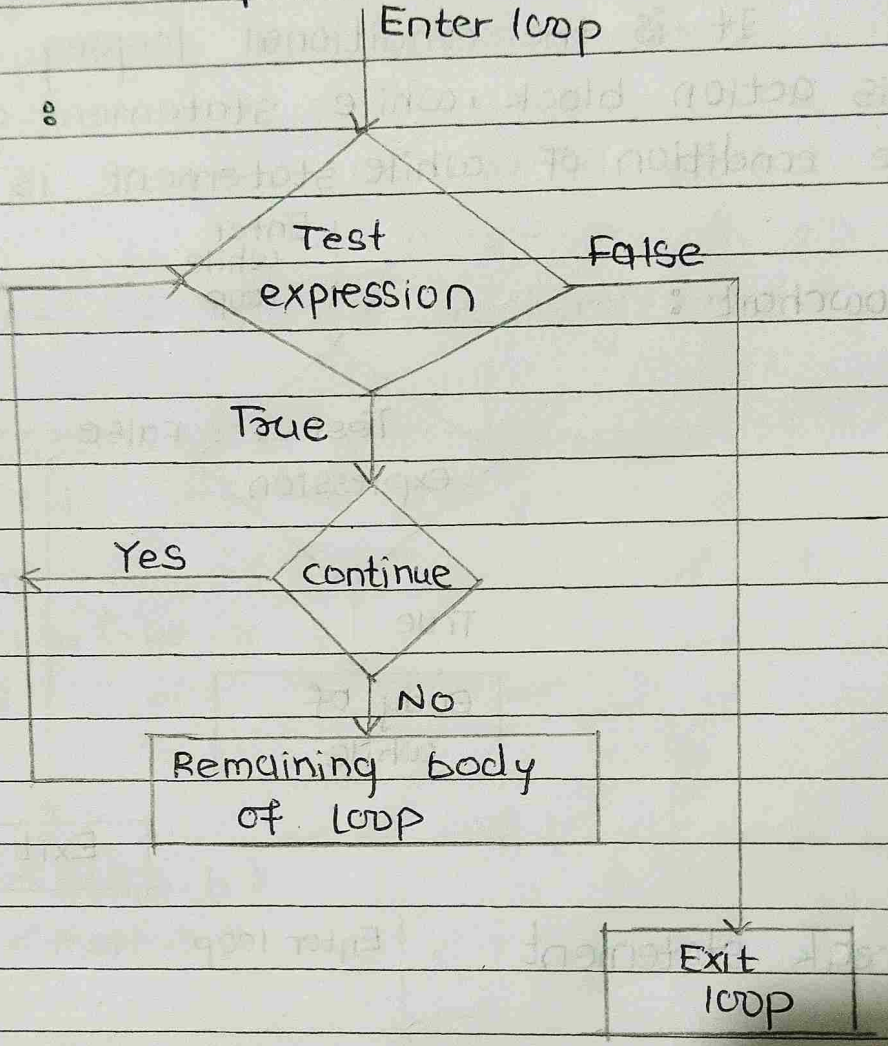


Date

Continue statement.

In ~~con~~ this we can skip action block.

Flowchart :



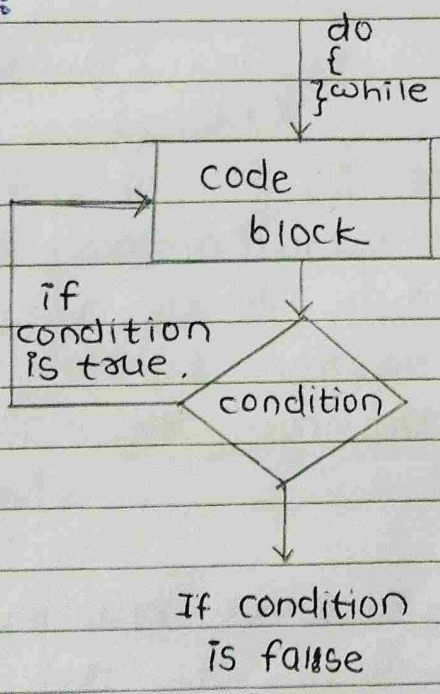
Do-while statement

Unlike for and while loop, which test the loop condition at the top of the loop, the do-while loop in C-programming checks its condition at the bottom of the loop.

P.S : Write a program to print numbers of 10-19.

```c
# include <stdio.h>
int main ()
{
  int a = 10;
 do
 {
  Printf ("value of a: %d \n", a);
     a = a+1
 }
  while (a<20);
  return (0);
  }
```

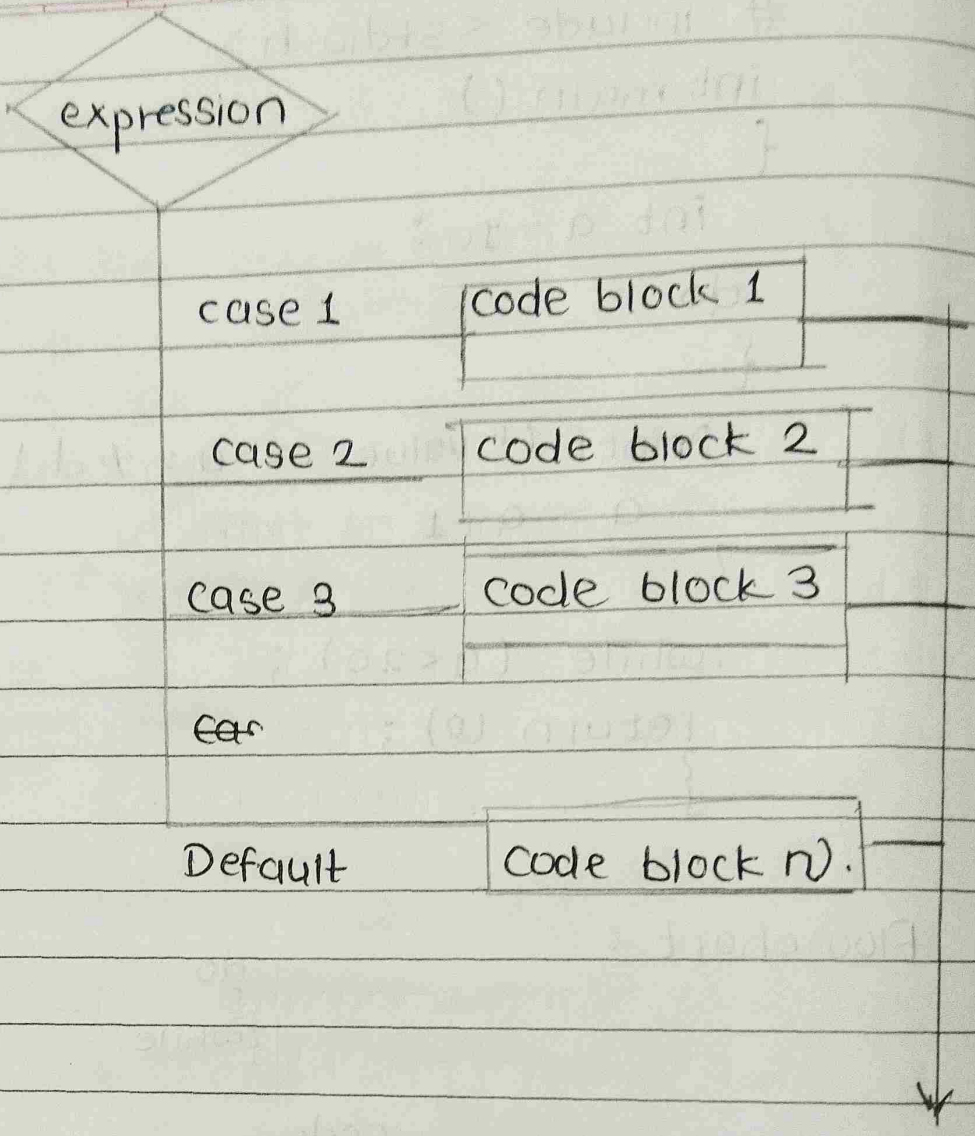
Flowchart :



V. imp

* Switch case. code with output.

A 'switch' statement in a C-program is a control flow statement that allows you to perform different actions based on the value of a variable or an expression. It's a convinient way to handle multiple cases or conditions efficiently.


expression
case 1
code block 1
case 2
code block 2
case 3
code block 3
Default
code block n).

Thursday.

Page No.

Date 12 10 23

Unit 4. Functions.

A function is a block of code only runs when its call, u can pass the data known as parameters into a function.

Functions are used to perform some actions and they are important for reusing code,
Define the code at once and use it many times.

For example: main() is a function which is used to execute d code and printf() is a function used to output / print text to the screen.

Main function.

The main function in C language is the entry point of a program where the execution of a program starts.
It is user defined function. i.e. mandatory for the execution of a program because when a C program is executed.

Important points about C main function.

It is the main function where the program execution starts.
Every program has exactly one main function.
The name of this function should be "main" not anything else.

The main function always return an integer value or void.

Function prototypes.

The C compiler prototype is a statement that tells the compiler about the function name, its return type, numbers and datatypes of its parameters, by using this information the compiler crosscheck function parameters and their data types, with function definition and function call.

Syntax :

```
return_type function_name
        (parameter_list);
```

→ Return type
It is the datatype of the value that the function returns.
It can be any datatype like int, float, etc.

→ Function name.
It is the identifier of the function.
Use appropriate names for the functions that specify the purpose of the function.

→ Parameter list
It is the list of parameters that a function expects datatype and name.

Friday.

Page No.
Date 13 10 23

Function types Explanation with examples.

i. User defined function.

A function is a block of code that performs specific task, C language allows you to define functions according to your need, these functions are known as user defined functions.

Eg. Suppose you need to create a circle and colour you can create two functions to solve this problem.

ii. In-built function.

This is a function that is already available in a programming language.

- ~~pin~~
- printf() → to print the o/t
- scanf() → read the o/t
- strlen() → to determine the length of string.
- strcmp() → used to compare two strings.
- atoi() → used to convert string into integer.

iii. Function with parameters.

Information can be passed to functions as a parameter.
Parameters act as variables inside the function, they are specified after the function name.

Syntax :

```
return_type function-name (par1, par2, par3)
{
 // code to be executed
 return 0;
}
```

Example :

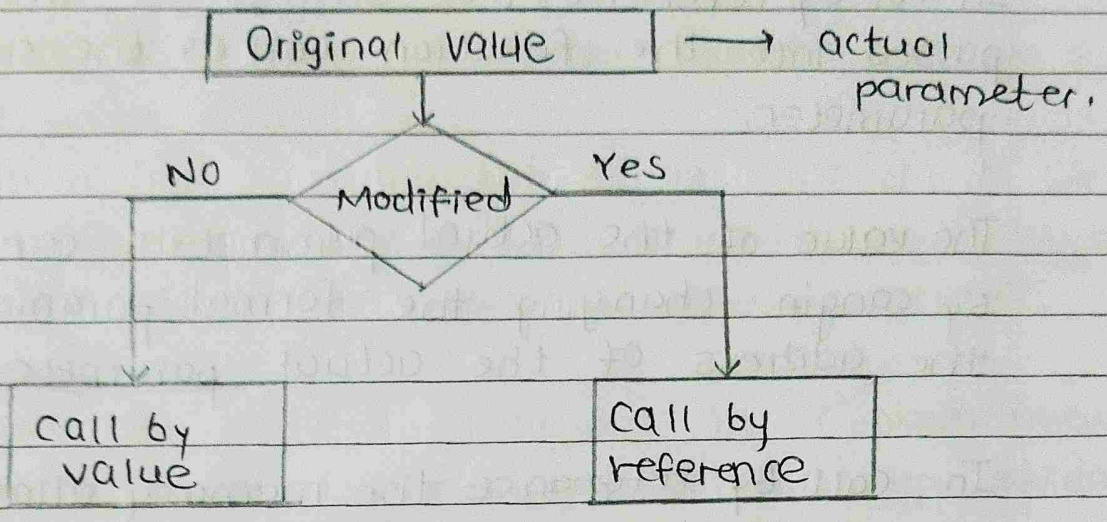
```
void myfunction (char name [])
{
 Printf ("Hello %s \n", name);
}
 int main ()
{
 myfunction ("nidhi");
 myfunction ("Kajal");
 myfunction ("Jayant");
 return 0;
}
```

iv. Function without parameters.

If a function takes no parameters, the parameters may be left empty, the compiler will not perform any function calls in this case.

Tuesday
Page No.
Date 17 10 2023

→ Call by value and call by reference.



There are two methods to pass the data into the function in C language i.e. call by value and call by reference.

Call by value

In Call by value method the value of the actual parameter is copied into the formal parameter. In other words we can say that the value of the variable is used in the function call, in the call by value method.

In call by value method we cannot modify the value of the actual parameter by the formal parameter.

In Call by value different memory is allocated for actual and formal parameters since the value of the actual parameter is copied into the formal parameter.

Page No.
Date

Call by reference

In call by reference the address of the variable is passed into the function call as the actual parameter.

The value of the actual parameters can be modified by ~~cangin~~ changing the formal parameters since the address of the actual parameters is passed.

In call by reference the memory allocation is similar for both formal and actual parameters.

* Recursion.

Recursion is the technique of making a function call itself, this technique provides a way to break complicated problems down into simple problems which are easier to solve.

* Local and Global variable / Scope.

19/10/2023
explanation with example

A ~~sp~~ scope in any programming is a region of the program where a defined variable can have its existence and beyond that variable it cannot be accessed there are two places where variables can be declared in C programming language.

Inside a function or a block which is called local variables.

Outside of all function which is called global variables.

Tuesday.

Page No.

Date 31 10 2023

Unit V. Array and Pointer.

* Array.

Array is a variable that can store multiple values with fixed size,

Array is defined as the collection of similar type of data item stored at contigious memory location.

Array are the derived datatypes in C programming language which can store the primitive type of data such as integers, character, float, double, etc.

Properties of array :

Each element of an array is same datatype and carry the same size.

i.e. int = 4 bytes.

Elements of the array are stored at contigious memory locations where the first element is stored at the smallest memory location i.e. int [0]

Elements of the array can be randomly accessed since we can calculate the address of each element of the array with the given base address and the size of the data element.

Declaration of Array / Syntax of array

The syntax of Array is as follows :

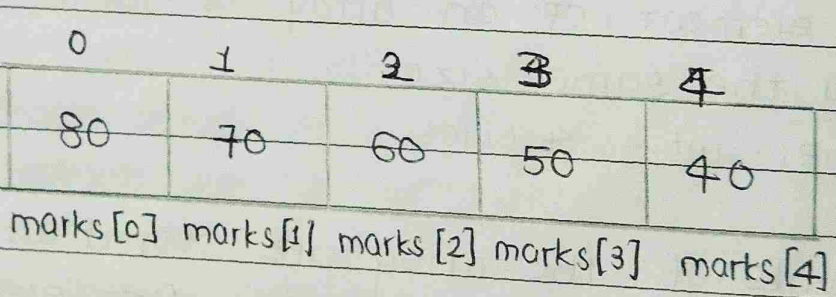
data_type array_name [array_size] ;

ex : int marks [3] = 80 ;

→ Representation of array / Initialization of array / Notation of array.

For example :

```
int marks [0] = 80 ;
int marks [1] = 70 ;
int marks [2] = 60 ;
int marks [3] = 50 ;
int marks [4] = 40 ;
```



Advantages of array.

1) Code optimization
Less code to the access the data

2) Is of traversing
By using the for loop we can retrive the elements of an array easily.

3) Random access
We can access any element randomly using the array

Disadvantage of Array.

1) Fixed size.

Whatever size we define at the time of declaration of the array we can't exceed the limit, so it does not increase the size.

Manipulating array element.

Array manipulation involves altering the contents of an array, this might involve changing the value of a specific element, adding or removing elements.

C programming being a low level language doesn't provide in-built functionality for adding or removing elements, to/from arrays.

Dynamic arrays can be created using pointers and memory management functions like malloc(), calloc(), realloc(), free().

Multidimensional arrays.

Multidimensional arrays are arrays of array, the simplest form is the two dimensional (2-D) array.

A 2-D array is defined in C using the following syntax:

data-type array-name [row][column] ;

* Character, arrays and strings.

In C programming string is a ~~one~~ 1-D array of characters and is defined as an array of characte but an array of strings in C is a 2-D array of character types.

Each string is terminated with a null character (~~/0~~) (\0)

Syntax :

char variable_r = { list of strings } ;

Program :

Problem Statement : Write a program to display character, arrays and strings.

code :

```
# include <stdio.h>
int main ()
{
  char arr [3][10] = { "Ram", "forram", "charram"};
Printf ( " string array elements are : " );
for ( int i=0 ; i<3 ; i++ )
{
```

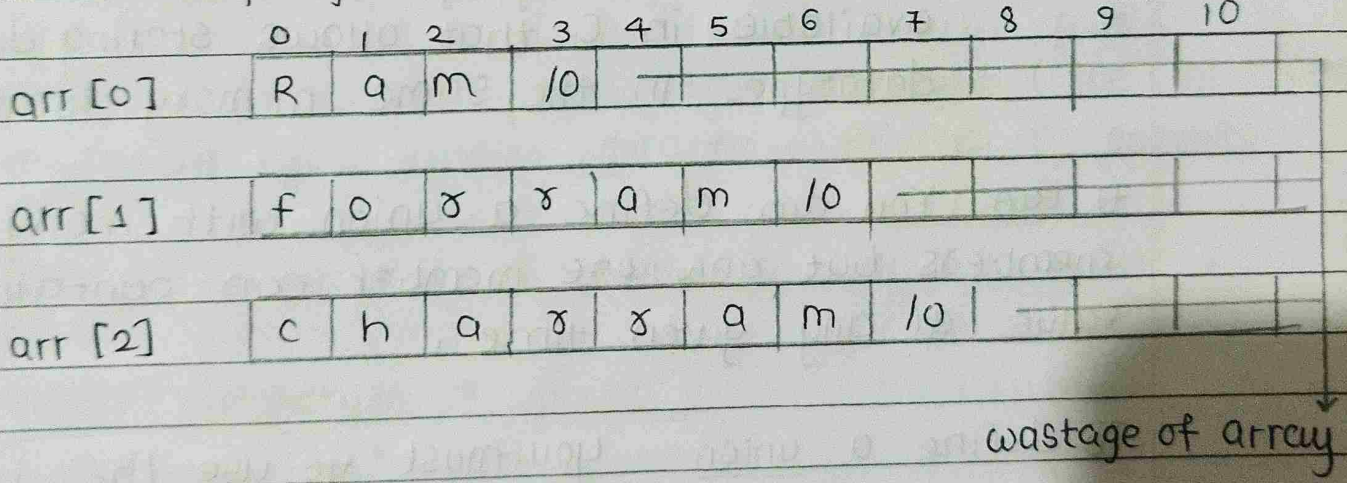
```
printf ("%s\n", arr[i]);
}
return 0;
}
```

Output :

String array elements are :
Ram
forram
charram.

* Memory representation of an array of strings.

From the program executed above :



* Structure and union in C.

Structure in C language is a user defined data type available in C that allows to combining of data items of different kinds.

Structures are used to represent a record, to define a structure you must use the struct statement.

The struct statement defines a new datatypes with more than or equal to one member.

Syntax:

```
struct [structure name]
{
  member definition ;
  member definition ;
  ...
  member definition ;
} ;
```

Union: Union in C language is a special datatype available in C that allows storing different datatypes in the same memory location.

~~U can~~ You can define a union with many members but only one member can contain a value at any given time.

To define a union you must ~~yo~~ use the union statement, in the same way as you did defining a structure.

Union statement defines a new data type with more than one member for your program.

Page No.
Date

Syntax :

```
union [union name]
{
  member definition ;
  member definition ;
  ...
  member definition ;
} ;
```

Pointers

Pointer is a variable that stores the memory address of another variable as its value.

A pointer variable points to a datatype (like int) of the same type and is created with the * operator.

Syntax / declaration of pointer :

```
        datatype * ptr ;
ex :      int *  ptr ;
```

Example :

```
int myage = 29 ;  // An int variable
int * ptr = & myage ;
// A pointer variable with the name ptr that stores
   the address of myage.
printf ("%d\n", my age ) ;
// output the value of myage.
```

```
printf ("%p\n", &myage);
// output the memory address of myage.
printf ("%p\n", ptr);
// output the memory address of myage with the
   pointer.
```

Dynamic memory allocation in C using malloc(), calloc(), realloc() and free().

1. malloc()

The malloc or memory allocation in C is used to dynamically allocate a single large block of memory with the specified size, it returns a pointer of type void which can be cast into a pointer of any form.

It does not initialize memory at execution time so that it has initialized each block with default garbage value initially.

Syntax:

```
ptr = (cast-type *) malloc (byte-size)
```

example:

```
ptr = (int *) malloc (100 * sizeof (int))
```

Since, the size of int is 4 byte this statement will allocate 400 bytes of memory and the pointer ptr holds the address of the first byte

in the allocated memory.

2. calloc()

The calloc or contigious allocation method in C is used to dynamically allocate the specified no. of blocks of memory of the specified type.

It is very much similar to malloc() function but has two different points and these are:

1) It initializes each block with the default value 0.

2) It has two parameters or arguments as compared to malloc().

Syntax:

```
ptr = ( cast_type* ) calloc( n, element_size);
```

example:

```
ptr = ( float * ) calloc ( 25, size of (float)) ;
```

Here, n is the number of elements and element size is the size of each element.

In example, this statement allocates space in memory for 25 elements ~~whi~~ with the size of the float.

3. realloc()

The realloc or re-allocation method in C is used to dynamically change the memory allocation of a previously allocated memory.

In other words if the memory previously allocated with the help of malloc or calloc is insufficient realloc() can be used to dynamically reallocate memory, reallocation of memory maintains the already present value and new block will be initialized with the default garbage value.

Syntax:

```
ptr = realloc (ptr, new size);
```

Here, ptr is reallocateed with new size.

4. Free().

free method in C is used to dynamically deallocate the memory allocate using functions malloc(), and calloc() is not deallocated on their own.
Hence the free method is used.
Whenever the dynamic memory allocation take place, it helps to reduce wastage of memory by freeing it.

Syntax: free(ptr);