INDEX

NAME: Debojit Nath STD.: 11 SEC.: 1st ROLL NO.: 29 SUB.: C Programming

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks

PURABI M

* ADDITION of Two Nos :—

① c = 2+3 → instruction

```
#include <stdio.h>
#include <conio.h>
  void main () {
      int a, b, c;
      clrscr ();
      printf ("Enter the value of a & b");
      scanf ("%d %d, &a, &b);
      c = a+b
      printf ("Sum is %d ", c);
      getch ();
  }
```

Note :— For sqrt, pow, abs we use — <math.h>
size of an integer is 2 byte.

→ amperser : (&) &a to address variables
%d : unspecified species
Scanf : input
Printf : output
} <stdio.h>

clrscr() : clear screen
getch() : to return back
} <conio.h>

void main() : main fn

conio : console input output

Note: 4 data types
(i) int
(ii) float
(iii) char
(iv) double

* Algorithm & Flowchart :—

[Program = Algorithm + Flowchart]

↳ An algorithm is a procedure used for solving a problem or performing a computation.

process:

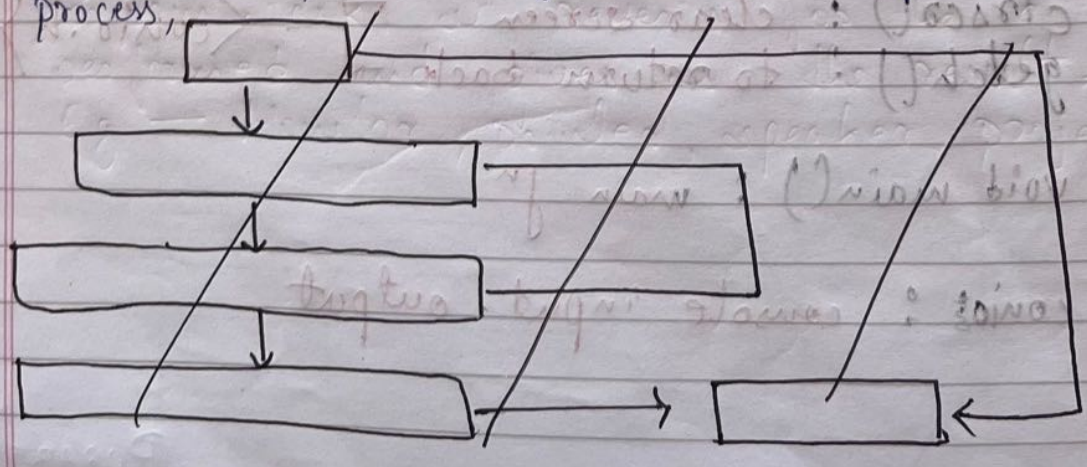
Step 1 : Read a, b, c, d;

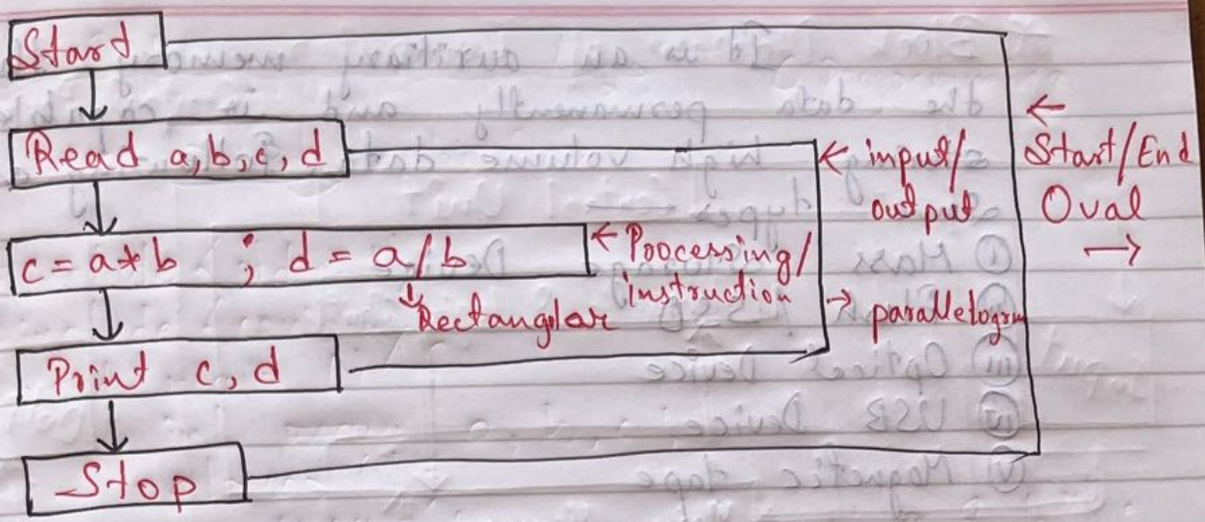
Step 2 : c = a * b
d = a / b

Step 3 : The product is c
The divid is d

Flowchart: A flowchart is a diagram that depicts a process, system or computer.

process,





* What is computer? What are the characteristics of computer.

Ans: A computer is a digital electronic machine that ~~can~~ is able to/ programmed to carry out sequences of arithmetic or logical operations automatically

Few characteristics of computer are —

(i) Computer has the accuracy of almost all data
(ii) It has high storage capacity.
(iii) It reduces the time and make day to day life easier
(iv) It reduces the paperwork.

* What is primary memory and what is secondary memory?

Ans:- It is the main memory of the computer where the currently processing data. There are two types of

(i) RAM - Random Access Memory
(ii) ROM - Read Only memory

It is an auxiliary memory that stores the data permanently and is capable of storing high volume data. Following are some types —

(i) Mass Storage Device
(ii) Flash / SSD
(iii) Optical Device
(iv) USB Device
(v) Magnetic tape

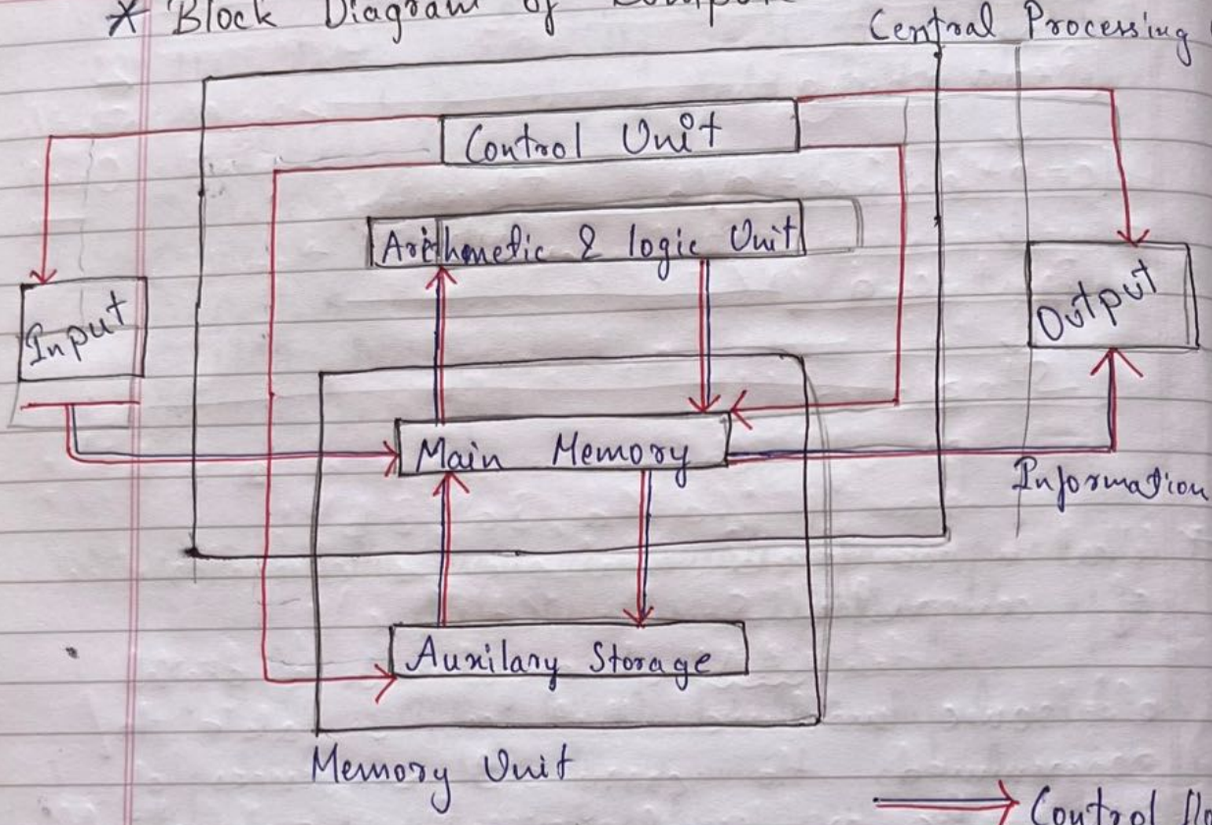
* Explain the main components of computer.

Ans:- The main components of computer are —

(i) Input Unit: The input unit or input device have to command computer at first then can only computer will respond. The information or data like no., alphabets and images can enter through input unit.
Eg — keyboard, mouse, joy stick, scanner, microphone, etc.

(ii) Output Unit: The output gives the result when we command a computer to perform a task, it reverts for the action performed and gives us the result.
Eg — Monitor, printer, speaker, projector

* Block Diagram of Computer



→ Control flow

→ Data flow

classmate
Date 2/9
Page

* Decision Making Statements :

C program is a set of statements which are normally sequentially in the order, in which they appear. This happens when no option or no repetition of certain calculations are necessary how-ever in practice we have a no. of situations where we may have to change the order of excution of of the statement based on certain condition, or need to repeat a group of statements until certain specified conditions are made.

C language passes diff^{nt} decisions making statements —

- If statement
- Switch statements
- Conditional operator statements
- Go to statements

* If Statement : It is basically a two way decision making statement & it takes the following form —

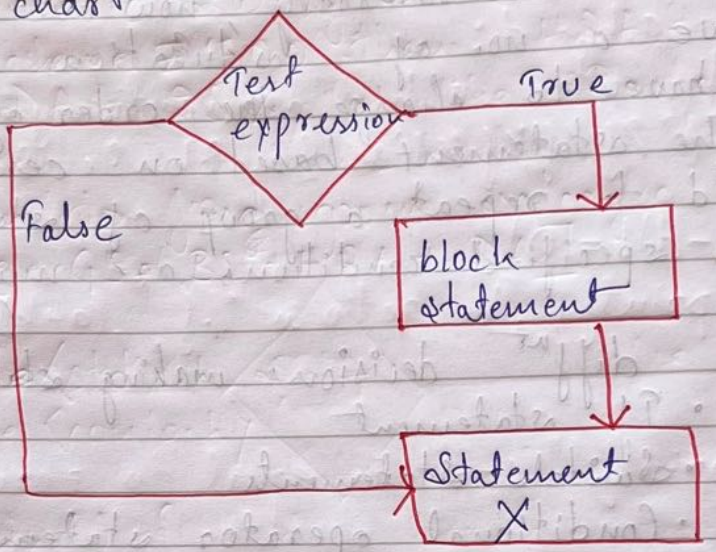
```
if (test expression) {
        [block statement]
}
```

It allows the compiler to evaluate the expression first and then depending upon the value of expression cerdian block of statements are executed.

P.T.O

Eg – (i) If (balance is zero)
borrow money
(ii) If (room is dark)
turns on the lights

Flow chart –



* If the test expression is true, the statement block is executed otherwise the statement block is skipped & there will be execution of 'X'.

* Write a Program to read the value of a, b, c, d from terminal & evaluate the ratio of (a+b)/(c-d) & print the result if c-d is a non-zero value.

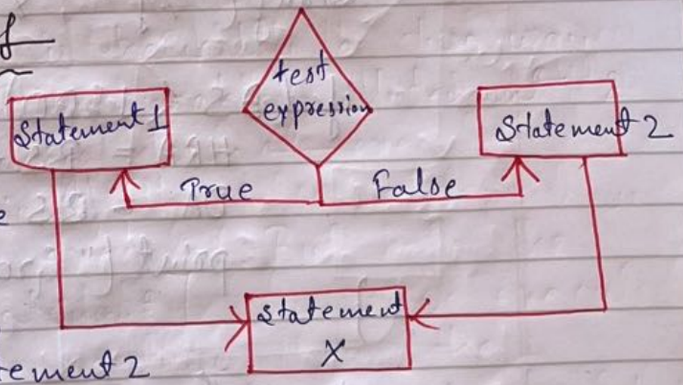
```
#include <stdio.h>
#include <conio.h>
void main () {
    float a, b, c, d;
    printf("Enter the digits/nos. :");
    scanf("%f %f %f %f", &a, &b, &c, &d);
    if (c-d != 0) {
        printf("The ratio is %f", (a+b)/(c-d));
    }
    else
        printf("No evaluation");
    print("Thank You");
    getch();
}
```

Test expression → & True →

& False →

& X →

* If & else Flow chart



If the test expression is true then the true block i.e statement 1 is executed if false, false block i.e statement 2 is executed.

In either case, the true or false block will be executed but both can't be executed together.

H/w

* In a company employee is paid under condition
(i) If Basic salary is less than 1500 ₹ then HRA = 10% & DA = 90% both of the Basic salary.

(ii) If Basic salary is either equal to or above ₹1500 then HRA = ₹ 500 & DA = 98% of the Basic salary.

Program to find gross salary
[i.e GS = BS + HRA + DA]

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

void main() {
    float BS, DA, HRA, GS;
    printf("Enter Basic Salary:");
    scanf("%f", &BS);

    if (BS < 1500) {
        HRA = BS * 0.10;
        DA = BS * 0.9;
        print("Gross Salary is %f", BS+HRA+DA);
    }

    else if (BS > 1500) {
        HRA = 500;
        DA = BS * .98;
        print("Gross Salary is %f", BS+HRA+DA);
    }
```

```
        else
            print ("wrong input");
        print ("\n Thank You");
        getch();
    }
```

7)

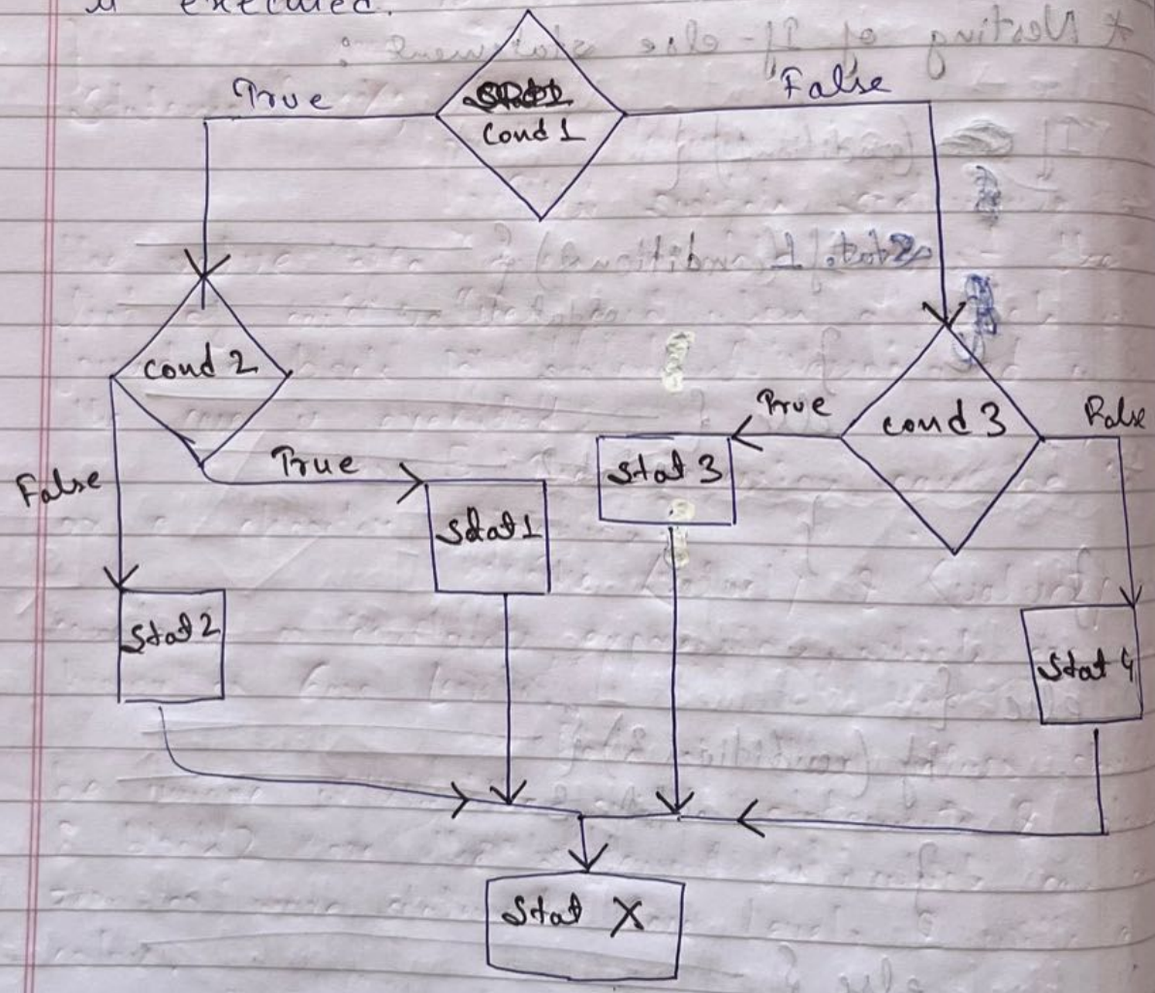
* Nesting of If-else statement :

```
If (condition1) {
    If (condition2) {
        stat 1
    }
    else {
        stat 2
    }
}
else {
    if (condition 3) {
        stat. 3
    }
    else {
        stat. 4
    }
}
stat. X
```

* When a series of decision are involved, we may have to use more than one if-else stat. in a nested form.

If the condition1 is true then the condition 2 is checked. If the condition is found true, the statement 1 is executed otherwise statement 2 is executed.

Again, if the test condition 1 is false it checks condition 3 when condition 3 is true the statement 3 is executed, else statement 4 is executed.



* A commercial bank has introduced an incentive policy by giving bonus to all deposit holders. The policy is as follows ——

① A bonus of 2% of the balance for everyone.

② A bonus of 5% of the balance is ~~more~~ given to the female account holder if their balance is more than ₹ 5000.

Soln

Date 9/9/22
Page

* Switch Statement :

The general form is

```
switch (expression)
{
    case <value1> :  block-1;
                     break;
    case <value2> :  block-2;
                     break;

    default : default block;
}
Statement X;
```

This switch statement tests the value of a given variable against the list of case values and when a match is found a block of statement associated with the case is executed.

The expression can be 'int' or a character expression.

The <value1> & <value2> are constants or constants expression, which are also known as Case Levels.

Each of these values should be unique within the switch statements & block-1 & block-2 are the statement list which can contain zero or more statements. Case Levels & with a colon.

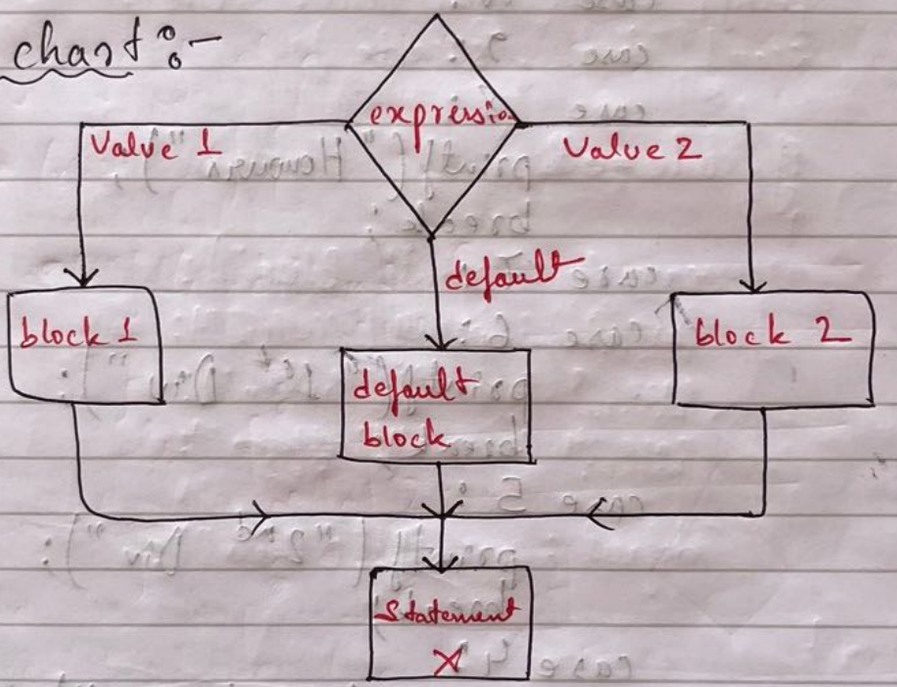
When the switch is executed the value of the expression is compared against the case level values. If a case is found

then the block of statements against that case are executed.

The break statement at the end of each block indicates that the particular case is executive & exits from the switch statement after that the control goes to the statement 'x'.

The default case is optional, when it will be executed if the value of the expression doesn't match with any of the case values.

* Flow chart :-



9 Write a C Program to display the division obtained by a student based on the following condition:

Per	Index	Div			
100	10	Honours	70-79	A	1st
90-99	9		60-69	B	
80-89	8		50-59	C	2nd
			40-49	D	3rd
			Rest	Fail	

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int per;
    printf("Enter your marks: ");
    scanf("%d", &per);

    switch (per/10)
    {
        case 10:
        case 9:
        case 8:
            printf("Honours");
            break;
        case 7:
        case 6:
            printf("1st Div");
            break;
        case 5:
            printf("2nd Div");
            break;
        case 4:
            printf("3rd Div");
            break;
        default:
            printf("Fail");
    }

    printf("Thank You");
    getch();
}
```

The Else If Ladder

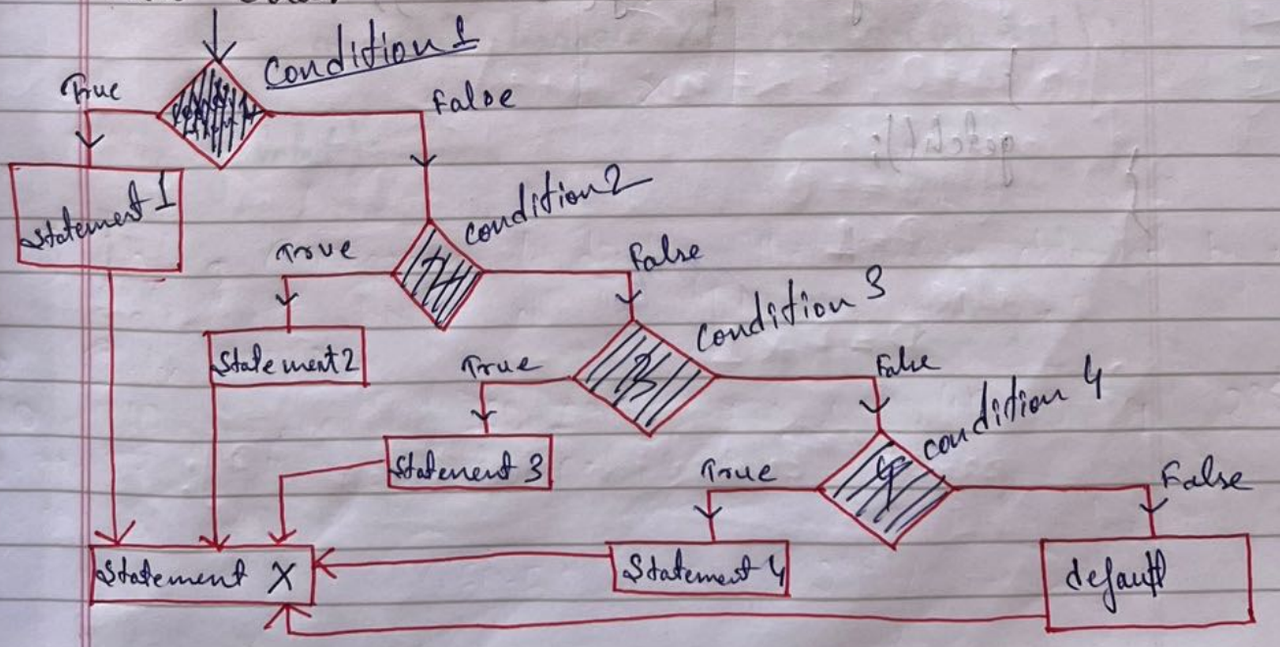
classmate
Date 12/9/22
Page

→ The general form

```
{
  if (condition 1)
        statement 1;

  Else if (condition 2)
        statement 2;

  else if (condition 3)
        statement 3;

  else
        default statement;
}
statment X;
```

→ When multiple decisions are involved a multipath decision is a chain of if's statement associated with each else is if.

→ Flow chart



* Write a C program for colour name print, reading the colour codes
1 → blue, 2 → Yellow, 3 → Green, 4 → Red,
0 → others.

```
#include <stdio.h>
#include <conio.h>
void main ()
{
    int code;
    printf("Enter Your Colour Code : ");
    scanf("%d", &code);

    switch (code)
    {
        case 1 : printf("Blue");
        break;
        case 2 : printf("Yellow");
        break;
        case 3 : printf("Green");
        break;
        case 4 : printf("Red");
        break;

        default : printf("Others");

    }
    getch();
}
```

* A electric p
domestic con
Unit
0 - 200
201 - 400
401 - 600
601 - above

```
#include <stdio
void main()
{
    float u
    printf("
    scanf("
    if(uni
        p
    else if
        p
    else i
        p
    else
        p
    getch
}
```

* A electric power distribution company charges its domestic consumes as follows,

Unit	Charge
0 - 200	₹0.5 per unit
201 - 400	₹100 plus ₹0.65, excess of 200 units
401 - 600	₹230 " ₹0.8, " " 400 "
601 - above	₹390 " ₹1, " " 600 "

```
#include <stdio.h>
void main()
{
    float unit;
    printf("Enter your unit: ");
    scanf("%f", &unit);
    if (unit < 201)
        printf("Charge = %f", unit * 0.5);
    else if (unit < 401)
        printf("charge = %f", unit-200 * 0.65);
    else if (unit < 601)
        printf("charge = %f", unit - 400 * 0.8);
    else
        printf("charge = %f", unit - 600 * 1);

    getch();
}
```

* Print the list of no. in the range 0-20, which are divisible by 2 aswellas 3.

* For loop
General

for (intia
{
}

Q Print :

#include <
#include
→ void main
{
for (
}
getch();

Q 6 12

for (int i
prin

Q Print the
which are

* for loop

General form —

```
for (intialisation ; condition ; updation)
{
    // do statement
}
```

Q Print : 1 3 5 7 9 11 13 15

```
#include <stdio.h>
#include <conio.h>
void main
{
    for (int i=1; i<16; i+=2)
        printf ("%d ", i);
}
getch();
```

Q 6 12 18 24

```
for (int i=1; i<5; i++)
    printf ("%d ", i*6);
```

Q Print the list of nos. in the range (0-20) which are divisible by 2 & 3

```
for (int i=0; i<21; i++)
    if (i%(2*3)==0)
        printf("%d ", i);
```

```
if (i%2==0 && i%3==0)
    printf(---);
```

for divisible by 5

```
for (int i=0; i<21; i++)
{
    if (i%5==0)
        printf("%d\n");
}
```

Q) C Program to evaluate the following f^n

$f(n) = 2^n$, where $0 \le n \le 10$

[without pow f^n]

```
int i, n, P=1;
printf("Enter the value of n: ");
scanf("%d", &n);
for (i=0; i<=n; i++)
{
    if (i==0)
        P=1
    else
        P=P*1
}
printf("Evaluation gives: %d", P);
```

double

Write

(i) 1
2
3
4

#incl
#inc
void

(i) {

}

(ii) {

}

Date 16/09

Write a C Programe to generate following:

(i)
1
2 2
3 3 3
4 4 4 4

(ii)
1
1 2
1 2 3
1 2 3 4

```
#include<stdio.h>
#include<conio.h>
void main()
```

(i)
```
{
    for(int i=1; i<=4; i++)
    {
        for (int j=1; j<=i; j++)
            printf("%d ", j);
    }
    printf("\n");
    getch();
}
```

(ii)
```
{
    for(int i=1; i<=4; i++)
    {
        for(int j=1; j<=i; j++)
            printf("%d ", j);
    }
    printf("\n");
    getch();
}
```

Q Write a C program to find the sum of the given series, read 'n'
1 + (1+2) + (1+2+3) + ... + (1+2+...+n)

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int n,i,j,term,sum=0;
    printf("Enter n:");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        term=0;
        for(j=1;j<=i;j++)
        {
            term=term+j;
        }
        sum=sum+term;
    }
    getch();
}
```

* Find the roots of a quadratic eqn (ax^2+bx+c)

```
#include <stdio.h>
#include <conio.h>
void main()
{
    float a, b, c, D, S;
    printf("Enter coefficients as (ax^2+bx+c) respectively:");
    scanf("%f %f %f", &a, &b, &c);

    D = (b*b) - (4*a*c);

    if (D>0)
    {
        printf("There are two real roots");
        S = sqrt(D);
        printf("The roots are %f & %f", (b+S)/2*a,
→ (b-S)/2*a);
    }

    else if (D==0)
    {
        printf("There are 2 equal roots")
        printf("\n The roots are %f and %f",
→ b/2*a, b/2*a);
    }

    else
        printf("The roots are imaginary");

    getch();
}
```

* Do while

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int i=1,sum=0;
    do{
        sum = sum + i
        i = i+2      // i+=2
      }
    while (i<=40);
    printf("%d", sum);
    getch();
}
```

* Goto Statement

→ goto Label;

...

Label:

Statement block//

→ Label:

Statement block//

...

goto Label;

The language C support the 'goto' statement to branch unconditionally from one point to another in the program.

The 'goto' statement requires a label in order to identify. The place where the branch is to be made.
A label is any valid variable name & it is followed by code.

Goto breaks the normal execution of the program if the label column is before the statment 'goto' label.

A loop will be formed & some statement repeatedly such a jump is backward jump.
On the otherhand if the label is placed after the goto label some statement will be skipped & the jump is forward jump.

Eg:

```
void main()
{
    double x, y;
    read:
    printf("Enter no.:");
    scanf("%f", &x);
    if (x<0) goto read;
    {
        y = sqrt(x);
        printf("%f → %f \n", x, y);
        goto read;
    }
    getch();
}
```

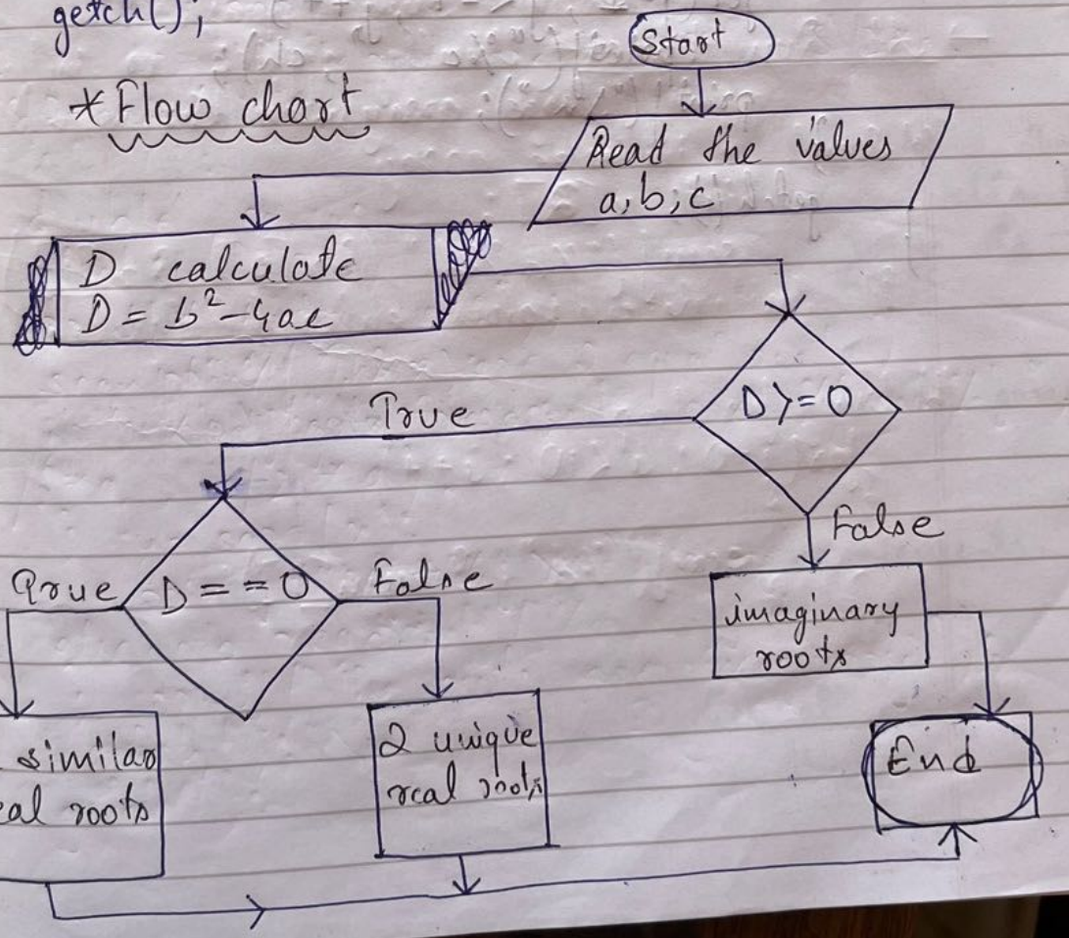
Q WAP in C to determine the roots of a quadratic eqn.

{nested if}

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
    float a,b,c,D,S;
    printf("Enter the coefficient as a,b,c (ax^2+bx+c):");
    scanf("%f%f%f",&a,&b,&c);
    D=b*b-(4*a*c)
    if(D>=0)
    {
        if(D==0){
            printf("There are two real similar roots-\n");
            printf("%f and %f",-b/2*a,-b/2*a);
        }

        else {S=sqrt(D)
            printf("There are two real unique roots-\n");
            printf("%f and %f",(-b+S)/2*a,(-b-S)/2*a);
        }

    else
        printf("there are two imaginary roots");
    getch();
}
```

```
void main()
{
    float a,b,c, D, S;
    . . . .
    if (D > 0){
        S = sqrt(D);
        printf("%f and %f", (-b+S)/2*a, (-b-S)/2*a);
    }
    else if (D == 0){
        printf("%f and %f", -b/2*a, -b/2*a);
    else
        printf("there are 2 imaginary roots");
    getch();
}
```

*Flow chart



* Display following using C program.

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

→

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int i,j;
    char ch='*';
    for(i=5; i>=1; i--)
    {
        for(j=1; j<=i; j++)
            printf("%c ", ch);
        printf("\n");
    }
    getch();
}
```

ARRAY

classmate Date 23/09 Page (154)

* **Array**: An array is a fixed-size sequenced collection of homogenous data. Individual ~~element~~ value is called element.

Types of arrays —

(i) One-dimensional arrays
(ii) Two-dimensional arrays
(iii) Multi-dimensional arrays

* **1-D Array**: A list of items can be given one variable name using only one subscript such a variable is called a single-subscripted variable or a 1-D array.

Eg. `int numbers[5];`

- `int` → <array data-type>
- `numbers` → <array name>
- `[5]` → <size>

Declaration: Arrays must be declared before they are used so the compiler can allocate space for them in memory. General form is —

`type variable-name [size];`

→ **Type**, states the type of element in the array such as int, float, char etc.

→ **Size**, indicate the maximum no. of elements that can be added

Initialization: Can be done either at compile time or at run time.

→ At compile time – The general form is

type array-name [size] = {list of values};

Eg:- int num [8] = {0, 3, 7};

where size is three and the value are 0, 3, 7 respectively, & remaining five elements will be initialised to zero.

Some times size may be omitted,

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

the compiler allocates enough space for all initialized elements

It is usually used for small array.

→ At Run time – Usually applied for initializing large arrays in C.

(a) Scanf to initialize array

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

It initialized values entered through keyboard

(b) Using for loop

```
for(i=0; i<10; i++)
{
    scanf("%d", &num[i]);
}
```

Declaration & Initialization

Date 26/09/22

* WAP to read 10 nos. into an array & display the list (using array)

Declaration & Initialization

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int i;
    int num[10];
    printf("Enter the nos.:");
    for(i=0; i<10; i++)
    {
        scanf("%d",&num[i]);
    }
    printf("The nos. are \n");
    for(i=0; i<10; i++)
    {
        printf("%d \n", num[i]);
    }
    getch();
}
```

Q WAP to read 5 no. & display its list & sum too (array)

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int i;
    int num[5]; printf("Enter nos.:");
    scanf("%d %d %d %d %d", &num[0], ..., num[4]);

    int sum;                    or  int sum = 0;
    sum=0
    for(i=0; i<5; i++)
    {
        sum = sum + num[i];
    }
    printf("The sum is: %d\n", sum);
```

Q WAP
read
[Decla

}

Q (mi
[De

classmate
Date 30/09/22
Page

Q WAP to display the maximum value in the array. read 5 values

[Declaration] & [Initialization]

```
max = num[0]
for(i=0; i<5; i++)
  {
    if (num[i] > max)
      max = num[i];
  }
printf("The array is\n");
for(i=0; i<5; i++)
    printf("%d \n", num[i]);

printf("The max is - %d", max);
getch();
}
```

Q (minimum)

[Declaration]

```
min = num[0]
for(i=0; i<5; i++)
  {
    if (num[i] > min)
      min = num[i];
  }
printf("The nos. is the list are:");
for(i=0; i<5; i++)
  {
    printf("%d \n", num[i]);
  }
printf("%d is the minimum", min);
getch();
}
```

* Find the even nos.

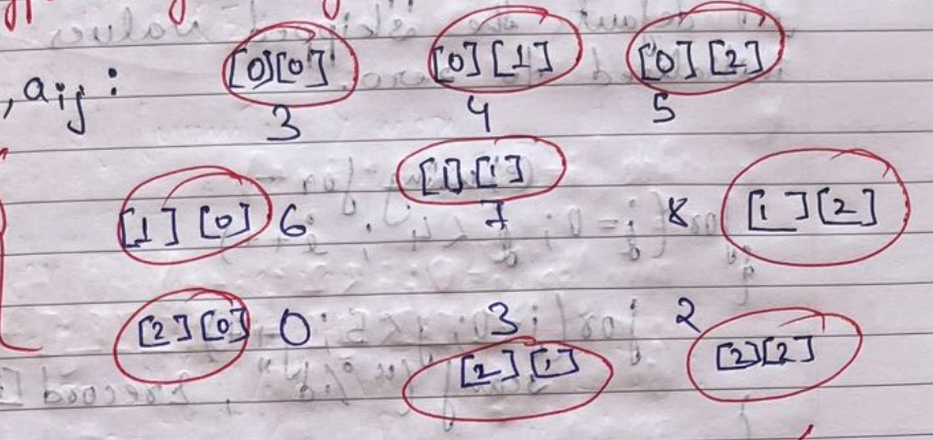
```
[Declaration]
[int initialization]
printf("The even nos. in the array are:");
for(i=0;i<5;i++)
{
    if(num[i]%2==0)
        printf("%d\n", num[i]);
}
getch();
}
```

* 2-D Array :- C allows us to define such tables of items by using 2-D arrays.

General form.

type_array array_name [row_size][column_size];

indexing, a[i][j] :



ie A (rows, columns)

* Initializing : (a) with in brackets —

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

which initializes the first row as all zeros & the second all 1's.

(b) in matrix form —

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

commas separates the values and values are stored in the table.

(c) skip size —

```
int table [][3] = {
                  }
```

when the array is completely initialized with values, we need not to specify the first dimension.

(d) skipped values
At default the skipped values will be initialized to zero.

(e) using for -

```
for(i=0; i<4; i++)
{
    for(j=0; j<5; j++)
        scanf("%d", &record[i][j]);
}
```

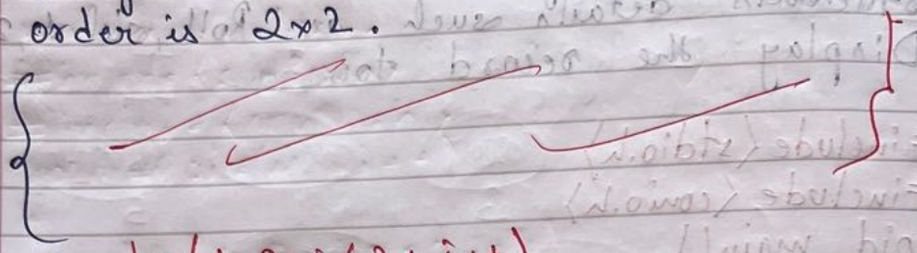
[4] [5]

Q WAP to create 2 dimensional array of size 4x5. Then 2-D Array holds the record of 4 students details such as Roll, sub 1, 2, 3 & 4 Display the record too.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int record[4][5], i, j;
    printf("Enter the details of the student:");
    for(i=0; i<4; i++)
    {
        for(j=0; j<5; j++)
            scanf("%d", &record[i][j]);
    }
    printf("The record stored is---");
    for(i=0; i<4; i++)
    {
        for(j=0; j<5; j++)
            printf("%3d", record[i][j]);
        printf("\n");
    }
    getch();
}
```

Note:
- Domain should be similar.
- order " " " .

* WAP to find the sum of a same order matrix.
Let first table be A & 2nd be B.
order is 2×2.



```
for(i=0; i<2; i++)
{
    for(j=0; j<2; j++)
        C[i][j] = A[i][j] + B[i][j];
}

for(i=0; i<2; i++)
{
    for(j=0; j<2; j++)
        printf("%3d", C[i][j]);
    printf("\n");
}
getch();
}
```

Slly, for product

```
C[i][j] = A[i][j] * B[i][j];
```

* WAP to

```
#include
#includ
void main()
{
int i,j
int A
int B
int A
for(i=0
{
    for
}
for(i=
{
}
}getch
```

Date 13/10
Page

* WAP to find the transpose of a matrix

```
#include <stdio.h>
#include <conio.h>
void main()
{
int i,j;
int A[3][2];
int B[2][3];
int A[3][2] = {1,2,3,4,5,6};

for(i=0, i<3, i++)
{
    for(j=0, j<2, j++)
        B[j][i] = A[i][j];
}

for(i=0, i<2, i++)
{
    for(j=0, j<3, j++)
        printf("%3d", B[i][j]);
}
getch();
}
```

classmate
Date 12/10
Page

* WAP to check if a matrix is a sq. or rect. reading the order also ~~input~~ read the matrix values.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int m,n,i,j;
    int A[10][10];
    printf("Enter the no of rows:");
    scanf("%d",&m);
    [redacted]
    printf("Enter the no of columns:");
    scanf("%d",&n);

    if(m==n)
        printf("The given matrix is square.");
    else
        printf("The given matrix is rectangular.");

    getch();
}
```

* WAP to find the sum of rows & columns of a given matrix.

```
printf("Sum of rows in Matrix");
for(i=0; i<m; i++)
{
    sum = 0;
    for(j=0; j<n; j++)
    {
        sum += A[i][j];
        for(k=0; k<1; k++)
        {
            R[i][k] = sum;
        }
    }
}
```

printing statement

Sily, for columns

```
    sum += A[j][i];
    for(l=0; l<1; l++)
    {
        C[l][i] = sum;
    }
}
}
```

* WAP to print the multipt product of two given matrices. $(m \times n \times n \times p)$

```
<Declaration & Initialization>
for (i=0; i<m ; i++)
{
    for (j=0; j<p; j++)
    {
        C[i][j] = 0;
        for (k=0; k<n; k++)
        {
            C[i][j] = C[i][j] + A[i][k] * B[k][j];
        }
    }
}
```

Character Array & String

Date 20/10/22

String is a sequence of character as a single data item.

String constant is a group of character which is defined bet" double quotation mark.

Eg (a) char string[6]
string = "Hello";

(b) printf("Hello"); → Hello

(c) printf("\"Well done"\"); → "Well done"

* Basic operation on Strings :

(i) Reading & Writing

(ii) Combing

(iii) Copying

(iv) Comparing for equality

(v) Extracting

* Declaration & Initialization of a String Variable:

Language C doesn't support string as a data type. Therefore, we need to represent string as a character array.

Generally,

char {string name} {size}

char city [20];

When the compiler assign a character string to a character at the end of the string '\0'.

Therefore, the size should be equal to the maximum of character plus one.

C permits of character array to be initialized in the following two form-

→ city = "New York";

→ city = {'N', 'e', 'w', ' ', 'Y', 'o', 'r', 'k', '\0'};

char city[] = {'G', 'O', 'O', '\0'};

* How to
① Scanf ("
can be use
a string
process wh
encountered
tab, etc

H/W
* Why do
end of

* How to read a string with termination?

① scanf ("%s", add);

The input f" 'scanf' can be used with "%s" specifier to read a string from the terminal!

It terminates it's process when the first wide-space is encountered such as blank space, new line, tabs, etc

H/w

* Why do we use null character at the end of a initializing string code:

classmate
Date 21/10
Page

* WAP to check if a given matrix is a identity matrix.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int i,j,m,n,sum;
    int A[10][10];

    printf("Enter the dimensions of the matrix--\n");
    scanf("%d%d",&m,&n);
    printf("Enter the matrix values (row wise):\n");
    for(i=0;i<m;i++)
    {
        for(j=0;j<n;j++)
        {
            scanf("%d",&A[i][j]);
        }
    }

    sum=0;
    if(m==n)
    {
        printf("The matrix is a Square Matrix");
        for(i=0;i<m;i++)
        {
            for(j=0;j<n;j++)
            {
                sum+=A[i][j];
            }
        }
```

```
opt = 0;
if (sum == m)
{
    for(i=0; i<m; i++)
    {
        for(j=0; j<m; j++)
        {
            if(i==j && A[i][j]==1)
                opt++;
        }
    }
    if(opt == m)
        printf("Identity Matrix");
    else
        printf("Not a Identity Matrix");
}
else
    printf("Rectangular Matrix");
getch();
}
```

31/10/22

* WAP in C to read a series of words from terminal using scanf().

```
#include <stdio.h>
#include <conio.h>
void main()
{
    char word1[40], word2[40], word3[40], word4[40];
    printf("Enter the words : ");
    scanf("%s %s", word1, word2);
    scanf("%s", word3);
    scanf("%s", word4);
    printf("\n word1 = %s \nword 2 = %s", word1, word2);
    printf("\n word3 = %s \n word4 = %s", word3, word4);

    getch();
}
```

* gets();

This function repeatedly read successive characters from the input and place them into the character array. Thus the entire line of text can be read and stored in an array the reading is terminated when the '\n' is entered and a null character is inserted at the end of the string.

* WAP to read a line of text using gets()

```
void main()
{
    char PAN[30];
    printf("Enter your PAN no. : ");
    gets(PAN);
    printf("Your PAN no. is -- %s", PAN);
    getch();
}
```

* getchar():

This function repeatedly reads successive character from the input and place them into the character array. Thus, the entire line of text can be read & stored in an array. The reading is terminated when the newline character '\n' is encountered and the null character '\0' is inserted on the end of the string.

* WAP to read a line of text containing a using a series of words from terminal using 'getchar();'

```
void main()
{
    char line[81], cha;
    int c;
    c = 0;
    printf("Enter your text");
    do
    {
        cha = getchar();
        line[c] = cha;
        c++;
    }
    while(cha != '\n');
    c--;
    line[c] = '\0';
    printf("\n %s \n", line);
    getch();
}
```

* WAP to copy string from a variable to another

1/12/22

* WAP to compine three string to a single variable.

```
void main()
{
    char fname [] = {" Viswanath "};
    char mname [] = {" Pratap "};
    char lname [] = {" Singh "};
    int i, j, k;
    char name [30];

    for(i=0; fname[i] != '\0'; i++)
    {
        name [i] = fname [i];
    }
    name [i] = " ";

    for(j=0; mname[j] != '\0'; j++)
    {
        name [i+j+1] = mname [j];
    }
    name [i+j+1] = ' ';

    for(k=0; lname[k] != '\0'; k++)
    {
        name [i+j+k+2] = lname [k];
    }
    name [i+j+k+2] = '\0';

    printf("%s\n", name);
    getch();
}
```

Functions

1* strcpy():
This f" takes the following form
strcpy(str1, str2);
It assigns the content of str 2 to str1. str2 may be a character, array variable or a string constant.
Eg: strcpy (str1, "hiye");

2* strcmp(): This f" compares two strings identifical by the argument and has a value zero if they are equal. If they are not, it has numeric difference between the first non-matching character in the string.
Eg: strcmp(str1, str2);
strcmp(str1, "Ram");
strcmp("Ram", "ram");

3* Strlen(): This f" counts and returns the no. of character present in a string
len = strlen(str1);

4* strcat(): This f" joins two strings together.
strcat(str1, str2);
where str1 & str2 are the character arrays. When the f" is executed str2 is appended to str1. It does show by removing the

null character at the end of str1 and starting placing the str2 from there.

* String Holding Fun°: The C language supports a large no. of string handling fun that can be used to carry out different string manipulation fun.

5* strstr(): This function searches whether the string str2 in content in string str1 or not. If yes, the function returns the positive of the 1st occurence in the string otherwise it returns a null pointer.
Eg- strstr (str1, str2);

* WAP
values

* WAP to count the size of a string & display the values as well as the string:

* WAP to read 2 String constant into S1 and S2 & compare, they if equal or not. If not join together, then copy S1 to S3. At the end we need to print all 3 strings & their lengths.

* Structu

of log

represe

"Name,

of di

comple

is nee

* Define

Their

declar

"Name,

its

Date 10/11/22

* Structure :

Convenient tool for handling a group of logically related data structures.

Eg- A structure can be used to represent a set of attributes such as "Name, Roll, Marks".

It is a mechanism of placing data of diff^nt types. It is a undefined data-types.

(i) It helps to organize a complex data in more synchronized way

(ii) It is a powerful concept that is needed to use in our program.

* Define a structure.

Ans- A structure must be defined in their format and later it may be used to declare the structure variables.

Consider a BOOK database consisting of "Name, Author, Pages and Price"

We can define a structure to hold its information as follows—

```
Struct Book
{
    char Name [30];
    char Author [30];
    int Pages;
    float Price;
}
```

The keyword "struct" declares a Structure to hold the details provided data fields namely "Name, Author, Page, Price". These fields are "structure element or Members" each element may belong to diff type of data-type.

Here "Book" is called the "Structure Tag".

A structure variable can be declared in following ways —

(i) struct Book b1, b2, b3 ;

(ii) struct Book
{
}b1, b2, b3;

(iii) struct Book S[10];

* Access
to the
link
using
Period
Eg:

* Define a
Salary

* Access a Value:

We can access an assigned value to the members of a struct in diff. ways the link bet" a member and a variable is established using the member operator '.' ie) Dot Operator or Period Operator.

Eg:

```
b1.page = 500;
strcpy (b2.Name, "C++");
scanf ("%s", b1.Author);
```

* Define a struct "Personal" that holds person's name, doJ and Salary.

Date 24/11/22

* WAP to read records of 5 students and display them —

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

struct student
{
    char NAME[30], ADDRESS[30];
    int ROLL [illegible] PHONE [illegible];
};

void main()
{
    int i; struct student s[10];
    for(i=0; i<5; i++)
    {
        printf("Enter student details respectively:");
        scanf("%s %s %d %d", s[i].NAME,
&s[i].ADDRESS, &s[i].ROLL, &s[i].PHONE);
    }
    printf("The provided details —");
    for(i=0; i<5; i++)
    {
        printf("NAME-%s ADDRESS-%s
ROLL-%d Phone-%d", s[i].NAME, s[i].ADDR
s[i].ROLL, s[i].PHONE);
    }

    getch();
}
```

* Define a struct "Time_struct" containing three var, int hr, int min, int sec. Develop a programe to assign value and print time.

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

struct Time_Struct
{
   int hr, min, sec;
}t1;

void main()
{
   printf("Enter the HOUR: ");
   scanf("%d", &t1.hr);
   printf("Enter the MINUTES: ");
   scanf("%d", &t1.min);
   printf("Enter the SECONDS: ");
   scanf("%d", &t1.sec);

   printf("Entered time is- ");
   printf("%d::%d::%d", t1.hr, t1.min, t1.sec);
   getch();
}
```

classmate
Date 14/11/22
Page

* Function: A function is a self contained block of code that performs a particular task. Once a f^n has been designed and packed it can be treated as a black box that take some data from the main program and return a value.

Elements of a used defined f^n are divided into three elements.

1. Function Definition
2. Function Call
3. Function Declaration

* Function Definition: It is also known as Function implementation & it may contain the following elements.

a. function name
b. function type
c. function's list of parameters
d. Local variable declaration
f. function statements
g. return statements

The general format is as follows:-

```
func_type func_name (parameter list)
{
    local variable declaration;
    statements;
    .....
    return statement;
}
```

Eg void

{

}

* Function

invoked.

a. functi

b. funct

c. paran

d. termi

The ge

Eg- in

(or)

* Functio

expres

and

the f^n

conta

paran

in

Eg
```
void sum(int a, int b)
{
    printf("sum = %d", a+b)
    return;
}
```

* Function Declaration : Like variables all fⁿ in C must be declared before they are invoked. A fⁿ declaration of 4 sub-elements
a. function type
b. function name
c. parameter list
d. terminating by ;

The ~~genral~~ general form —

func-type func-name (parameter list);

Eg- int sum (int, int);
(or) (int a, int b);

* Function Call : A function call is a post fix expression. It is used as a part of an expression and it will be evaluated first. In fⁿ call the fⁿ name is the operand & the paranthesis contains list of actual parameters. The actual parameters must match fⁿ format parameters in type, order and number.

* WAP to solve a multiplication using user defined f^n :

discrete
Date
Page

* Matrix sum

* Matrix multiply

* What is Recursion?

Ans: When a declared function calls another function occurs. A recursion is a ~~special~~ special function which can call itself.

```
main()
{
    printf("——");
    main();
}
```

* WAP to find factorial of a no. using f^n.

* Passing array to a function: To pass an 1-D array to a called fn it is sufficient to list the name of the array without any subscript and size of the array as arguments.

* WAP to find the largest value from an array.

17/11/22

* Nesting Of Functions:

C permits nesting of fns freely the main fn can call another fn. say fn1 which can call fn2 and so on.

* WAP to find ratio of three nos(a,b,c).