matrix Sum 2D

DATE : / 20
PAGE No

Array → coll. of data of same type

- An array is a collection of data of the same type that are referenced by a common name
- Elements stored in contiguous memory locations
- first element is stored at starting address (called Base address)
- Elements accessed using index value
- Index start from 0.

1-D Array:

- Declaration:
  - data_type var_name [size];
    int array [5];
- Initialization:
  - data_type array [size] = { value list };
    int array [5] = {1, 2, 3, 4, 5};
  - data_type array [] = { value list }
    int array [] = {1, 2, 3, 4, 5};

DATE: / 20
PAGE No

- array[0] = 1, array[1] = 2, array[2] = 3, array[3] = 4, array[4] = 5

- O/P:

```
int arr[5] = {10, 20, 30, 40, 50};
for (i=0; i<5; i++)
{
    printf("%d", arr[i]);
}
```

I/P:

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

- Memory address calculation:

Address of A[i] = B + w * [i - LB]

where

B = Base address
w = Storage size (in byte)
i = Subscript of element
LB = Lower limit / Lower Bound of subscript, if not specified assume 0 (zero)

DATE : / 20
PAGE No

Eg int A[5]; 1-D, 2-D.

A[3] B = 1000

Add = 1000 + 2 x (3 - 0)

= 1006

2-D array :

→ data_type var_name[row][col];
int matrix[3][3];

- Declaration :

Initialization :

- data_type matrix[row][column] = {value list};

~~Initialization~~ int ~~matrix[3][3];~~ data type

int matrix[2][2] = {1, 2, 3, 4};

int matrix[2][2] = {{1,2}, {3,4}};

- data_type matrix[][column] = {value list};

int matrix[][2] = {1, 2, 3, 4};

- matrix[0][0] = 1, matrix[0][1] = 2,
matrix[1][0] = 3, matrix[1][1] = 4.

DATE: / 20
PAGE No

- o/p;

```
int mat[2][2] = {10,20,30,40};
for(i=0; i<2; i++)
{
    for(j=0; j<2; j++)
    {
        Printf("%d\t", mat[i][j]);
    }
    Printf("\n");
}
```

[10 20
30 40

- P/P:

```
int mat[2][2];
for(i=0; i<2; i++)
{
    for(j=0; j<2; j++)
    {
        Scanf("%d", &mat[i][j]);
    }
}
```

DATE : / 20
PAGE No

* Memory Address calculation:

- Row major:

Address of A[I][J] = B + w * [N*(I-Lr) + (J-Lc)]

- Column major

Address of A[I][J] = B + w * [(I-Lr) + M*(J-Lc)].

Where
B = Base address
I = row subscript of element
J = column subscript of element
w = storage size (in byte)
Lr = lower limit of row / start row index of matrix, if not given assume 0 (zero)
Lc = lower limit of column / start column index of matrix, if not given assume 0 (zero)
M = No. of row
N = no. of column

DATE: / 20
PAGE No

Q. Input 20 integer no. from user & find out their sum and average.

```
int arr[20] = {1,2,3,4 - - - - - - 20};

for( i=0; i<20; i++)
{
    printf("%d", arr[i]);
}
    sum = sum + i
    printf("
```

→

```
void main()
{
    int a[20], i, sum=0;

for( i=0; i<20, i++)
{
    scanf("%d", &a[i]);

    sum = sum + a[i];

}

float avg;
  avg = sum/20.0;
```

DATE : / 20
PAGE No

Q. Input two matrix of order 3 x 3 Calculate their sum and Print the Resultant matrix.

```
int mat[3][3], i, j;
for ( i=0 ; i<3 ; i++)
{
    Scanf("%d", & mat[3][3] );
}
for ( i=0 ; i<3 ; i++)
{
}
Pnt mat[3][3] , i , j ;
for ( i=0 ; i<3 ; i++)
{
    for ( j=0 ; j<3 ; j++)
    {
        Scanf("%d", & mat[i][j]);
        Sum = mat1[i][j] + mat2[i][j]
    }
}
```

DATE : / 20
PAGE No

Ans

```
Void main()
{
    int m1[3][3], m2[3][3], m3[3][3],
                                i, j;

(a) for (i=0; i<3 ; i++)
        for( j=0; j<3, j++)
        Scanf("%d", &m1[i][j]);

for(i=0; i<3 ; i++)
    for ( j=0; j<3, j++)
    scanf("%d", &m2[i][j]);

for (i=0; i<3; i++)
    for (j=0, j<3, j++)
    m3[i][j] = m1[ij][j] + m2[i][j];

for
```

DATE : / 20
PAGE No

* String and character Arrays:-

- A string is a sequence of characters that is treated as a single data item and terminated by null character '\0'.

- String is actually 1-D array of characters.

- Initialization:

- char arr[] = "Hello";

- char arr[] = {'H', 'e', 'l', 'l', 'o', '\0'};

* Reading strings:

- scanf()

```
char str[50];
scanf("%s", str);
```

Reads the sequence of characters until it encounters a whitespace.

- gets()

```
char str[50];
gets(str);
```

Reads whole string.

DATE: / 20
PAGE No

(String.h)

* Writing String:
  - Printf()
    Printf("%s", str);
    If the string has formatting characters like "%", then Printf() would give unexpected Result
  - Puts()
    Puts(str);

* String function:
  - Strcpy (S1, S2);
    Copies string S2 into string S1
  - Strcat (S1, S2);
    Concatenates string S2 onto the end of string S1
  - Strlen (S1);
    Return the length of S1.
  - Strcmp (S1, S2);
    Returns 0 if S1 and S2 are the same; less than 0 if S1 < S2; greater than 0 if S1 > S2

DATE : / 20
PAGE No

Q. Input a String from user Print the length of the this string without using strlen function.

Ans:

```
i=0;
while (str[i] != '\0')
    i++;
```