Math Series

C Programming

UG Sem-3 Major (Kalyani University)

Day - 07

Digit Sum

```
while(n != 0){
    sum = sum + (n % 10); // Add the last digit to sum
    n = n / 10;     // Remove the last digit
}
```

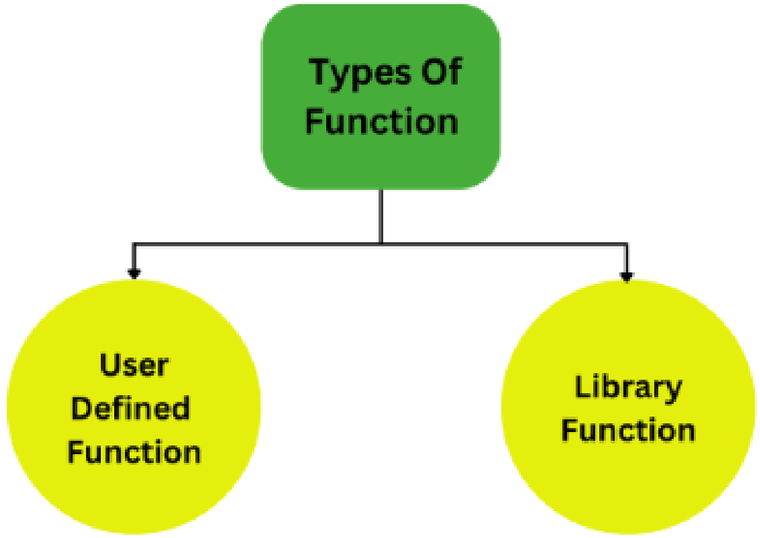
Digit Sum

If the user enters `123`, the loop proceeds as follows:

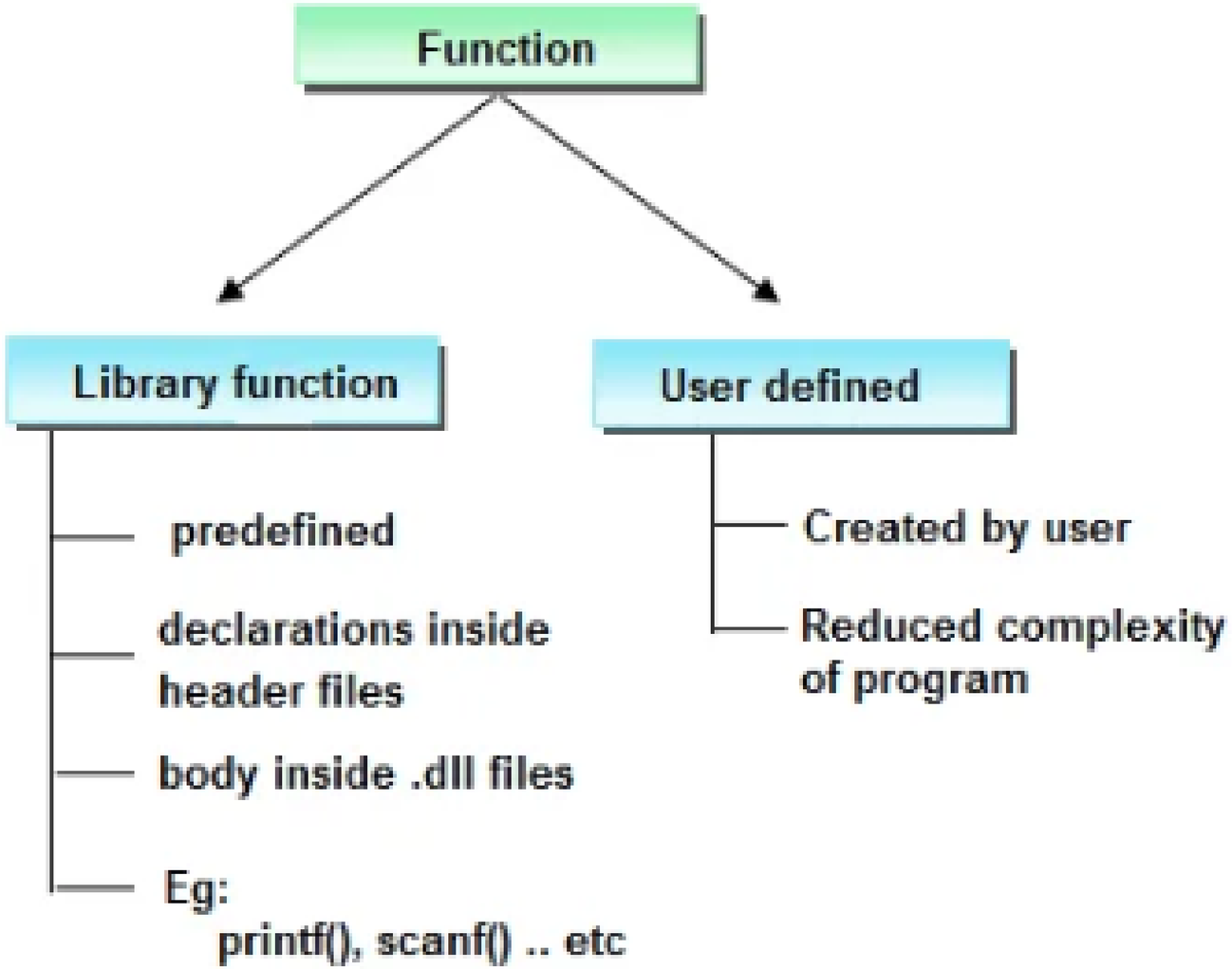
Iteration	`num`	`num % 10` (`digit`)	`sum`	`num /= 10` (next `num`)
1	123	3	3	12
2	12	2	5	1
3	1	1	6	0

Functions in C

A function is a block of code which only runs when it is called.
You can pass data, known as parameters, into a function.
Functions are used to perform certain actions, and they are important for reusing code: Define the code once, and use it many times.



Functions in C



Functions in C

Syntax

```
returnType functionName(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

Functions in C

```c
#include <stdio.h>

// Create a function
void calculateSum() {
  int x = 5;
  int y = 10;
  int sum = x + y;
  printf("The sum of x + y is: %d", sum);
}

int main() {
  calculateSum();  // call the function
  return 0;
}
```

```c
// Create a function
void myFunction() {
  printf("I just got executed!");
}

int main() {
  myFunction(); // call the function
  return 0;
}

// Outputs "I just got executed!"
```

Print an Array

int arr[5] = {4,8,7,9,2}

float arr[5] = {4.2,8.5,7.1,9.5,2.6}

```c
// float array of size 3
float arrf[3] = {1.1, 2.2, 3.3};
printf("\n");
for(int i = 0; i < 3; i++){
    printf("%.1f ", arrf[i]);
}
```

```c
int arr[5] = {10, 20, 30, 40, 50};

// print integer array elements
for(int i = 0; i < 5; i++){
    printf("%d ", arr[i]);
}

// reverse order printing
printf("\n");
for(int i = 4; i >= 0; i--){
    printf("%d ", arr[i]);
}
```

Print an Array

int arr[5] = {4,8,7,9,2}

1) Update one element then print all array

2) Print arr[3] element

Print an Array

Give an array of marks of 10 students, if the marks of any student is less than 35, print its roll number. [here roll number refers to index of array]

```c
int marks[10] = {45, 32, 67, 29, 50, 80, 22, 90, 33, 55};

for(int i = 0; i < 10; i++){
    if(marks[i] < 35){
        printf("Student with roll number %d has failed.\n", i);
    }
}
```

Print an Array

1 D ARRAY:

C	O	D	I	N	G	E	E	K
0	1	2	3	4	5	6	7	8

single row of elements

2 D ARRAY:

i \ j	col 0	col 1	col 2
row 0 → 0	A	A	A
row 1 → 1	B	B	B
row 2 → 2	C	C	C

column

array elements

rows

Print an Array

int arr[3][2] ={{1,2,3},{3,2,1}}

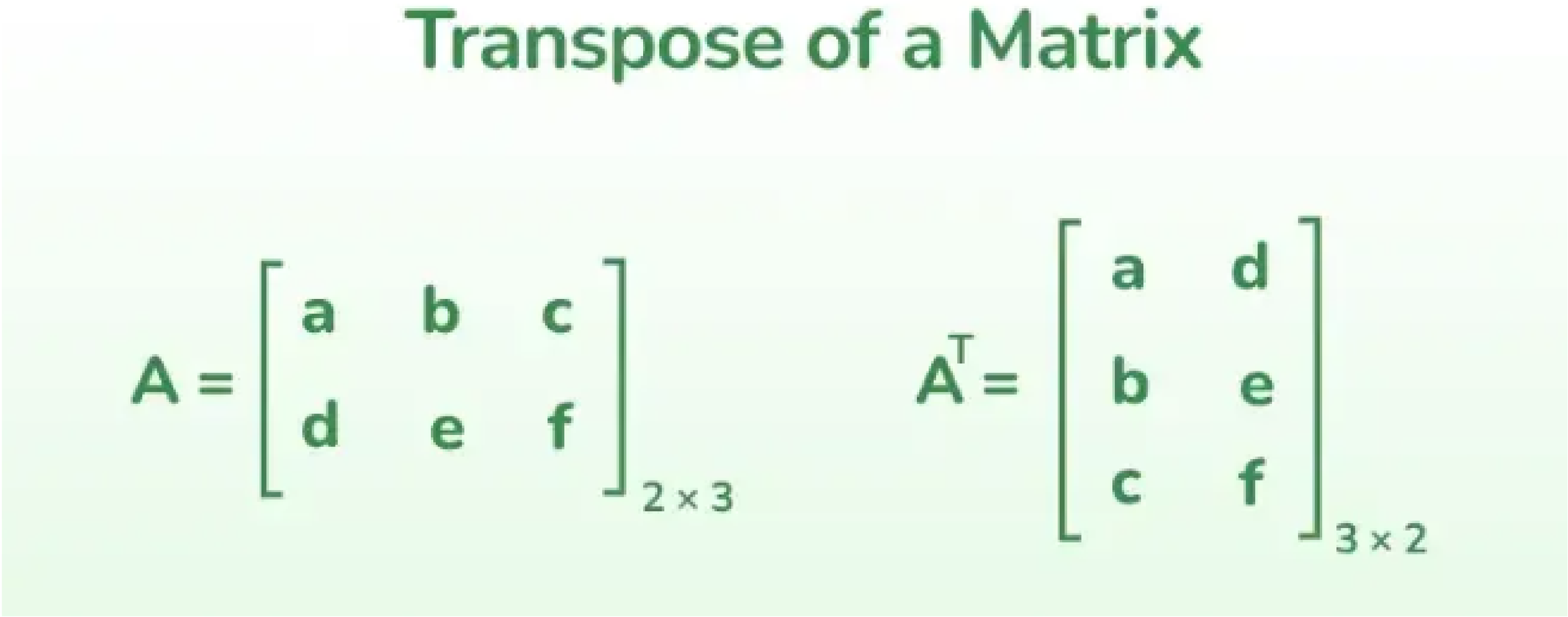
int arr[3][2] ={1,2,3,3,2,1}

```c
int arr[2][3] ={1,2,3,3,2,1};
    for(int i = 0; i < 2; i++){
        for(int j=0; j<3; j++){
        printf("%d\t", arr[i][j]);
        }
         printf("\n");
    }

    /* output:

    1       2       3
    3       2       1

    */
```

Matrix

1) Take input for rows and cols
2) Formation of new matrix
3) input elements for matrix
4) Output

Transpose Matrix

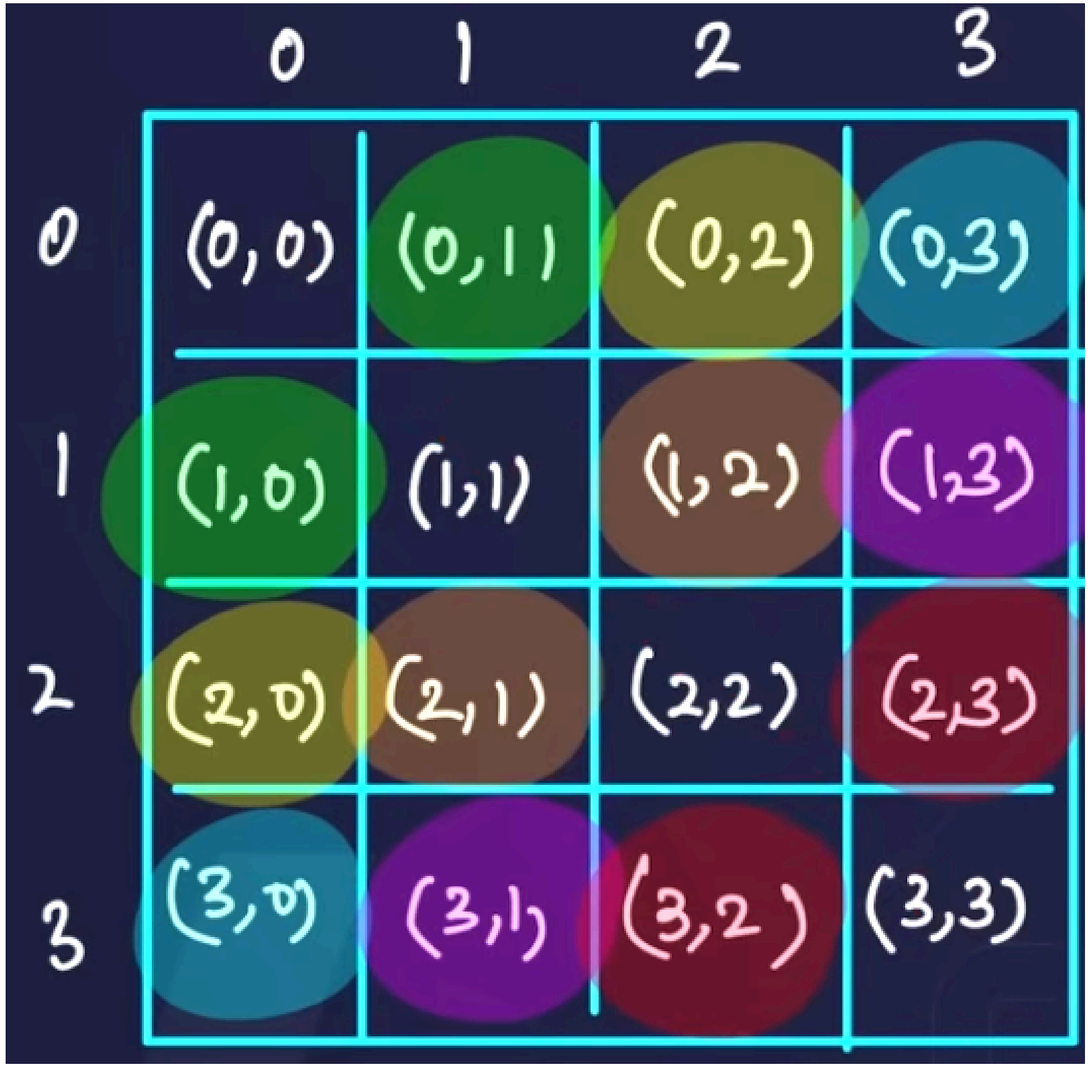


Transpose of Rectangular Matrix

```c
int m = 2, n = 3;

int a[2][3] = {{1, 2, 3}, {4, 5, 6}};

int t[3][2];

// Find transpose
for (int i = 0; i < m; i++)
{
    for (int j = 0; j < n; j++)
    {
        t[j][i] = a[i][j];
    }
}
```

```c
// Display transpose
printf("Transpose of the matrix:\n");
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < m; j++)
    {
        printf("%d ", t[i][j]);
    }
    printf("\n");
}
```

Transpose of Square Matrix




```java
// Transpose in-place
for (int i = 0; i < n; i++) {
    for (int j = i; j < n; j++) {
        int temp = a[i][j];
        a[i][j] = a[j][i];
        a[j][i] = temp;
    }
}
```

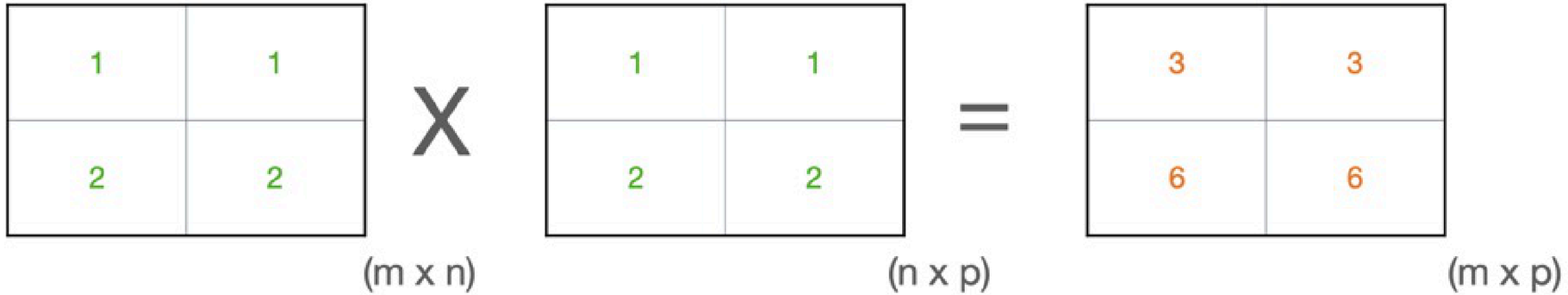
Transpose of Square Matrix

```
int a[3][3] = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9}
};

// Transpose in-place
for (int i = 0; i < n; i++) {
    for (int j = i; j < n; j++) {
        int temp = a[i][j];
        a[i][j] = a[j][i];
        a[j][i] = temp;
    }
}
```

```
// Display transpose
printf("Transpose of the matrix:\n");
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        printf("%d ", a[i][j]);
    }
    printf("\n");
}
```

Matrix Multiplication

Square Matrix:



Rectangular Matrix: