Class 5

Class notes

2D Array & Function

2D Array

A **2D array** is like a grid (matrix) — rows and columns.

```
int a[2][2] = {
  {3, 5},
  {1, 9}
};
```

- **Syntax:**
  `data_type array_name[row][column];`
- **Total size:** `row × column`
- **Accessing elements:**
  `a[0][1]` → element in 1st row, 2nd column

Traversing a 2D Array

```
for(int r = 0; r < 2; r++) {
    for(int c = 0; c < 2; c++) {
        printf("%d ", a[r][c]);
    }
}
```

Sum of All Elements

```
long long int sum = 0;
for(int r = 0; r < 2; r++) {
    for(int c = 0; c < 2; c++) {
        sum += a[r][c];
    }
}
printf("The sum is = %lld\n", sum);
```

Count Even & Odd Elements

```
int even = 0, odd = 0;
for(int r = 0; r < 2; r++) {
    for(int c = 0; c < 2; c++) {
        if(a[r][c] % 2 == 0)
            even++;
        else
            odd++;
    }
}
```

Condition:

Even → `a[r][c] % 2 == 0`

Odd → `a[r][c] % 2 != 0`

2. Built-in Functions

Header File

To use mathematical operations, include:

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

Common Functions

Function	Description	Example	Output
pow(x, y)	x raised to power y	pow(2,3)	8
sqrt(x)	Square root	sqrt(16)	4
abs(x)	Absolute value	abs(-5)	5
ceil(x)	Round up	ceil(4.2)	5
floor(x)	Round down	floor(4.9)	4
round(x)	Round nearest	round(4.6)	5

Example Code

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

int main() {
    int n = 16;
    double root = sqrt(n);
    int power = pow(n, 5);
    int neg = -100;
    int a = abs(neg);
    float f = 4.7;
    int c = ceil(f);
    int fl = floor(f);
    int r = round(f);
    printf("%d", r);
}
```

3. Functions

Definition

A **function** is a block of code that performs a specific task.

Types:

1. **Built-in functions** – provided by C (e.g., `printf()`, `sqrt()`)
2. **User-defined functions** – created by the programmer

Syntax

Declaration:

```
return_type function_name(parameters);
```

Definition:

```
return_type function_name(parameters) {
    // body
}
```

Calling:

```
function_name(arguments);
```

Example: Function with No Return

```
void add(int x, int y) {
    int sum = x + y;
    printf("The sum is = %d\n", sum);
}

int main() {
    int a = 1, b = 5;
```

```
    add(a, b);
}
```

- **`void`** → no return value
- Passes values from `main()` to `add()`

Example: Function with Return Value

```
int add(int x, int y) {
    int sum = x + y;
    return sum;
}

int main() {
    int a = 1, b = 5;
    printf("Result = %d\n", add(a, b));
}
```

Summary Table

Term	Meaning	Example
Function Declaration	Tells compiler about function	`int add(int, int);`
Function Definition	Actual code	`int add(int x, int y) {...}`
Function Call	Executes it	`add(a, b);`