Cardamom Planters' Association College
Pankajam Nagar, Bodinayakanur

STUDY MATERIAL

(2025-2026)

Presented by

C.KONDALRAJ

Dept. Of. CS&IT

Basic Structure of C Program

Sample Code:

```
#include <stdio.h>

int main() {
    printf("Hello, World!");
    return 0;
}
```

Explanation:

- `#include <stdio.h>`: Includes standard I/O functions
- `int main()`: Entry point of the program
- `printf(...)`: Prints text to screen
- `return 0;`: Ends the program

What does `#include <stdio.h>` do?

Answer: It tells the compiler to include standard input/output library functions like `printf` and `scanf`.

Data Types and Variables

Example:

```
int age = 25;
float weight = 55.5;
char grade = 'A';
```

Explanation:

- `int`: Integer numbers
- `float`: Decimal numbers
- `char`: Single characters

Which data type is used to store decimal numbers?

Answer: `float` or `double`

Input and Output

Example:

```
#include <stdio.h>

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    printf("You entered: %d", num);
    return 0;
}
```

Explanation:

- `scanf("%d", &num);`: Takes input from user
- `&` means “address of” the variable

What is used to take user input?

Answer: `scanf()`

Arithmetic Operators

Operators:

- `+` (Addition)
- `-` (Subtraction)
- `*` (Multiplication)
- `/` (Division)
- `%` (Modulus - remainder)

Example:

```
int a = 10, b = 3;
printf("Sum: %d\n", a + b);
```

What will be the output of `10 % 3`?

Answer: `1` (remainder after division)

Conditional Statements (`if`)

Example:

```
int num = 5;
if(num > 0) {
    printf("Positive number");
}
```

What does `if` do in C?

Answer: It checks a condition and runs code only if the condition is true.

Loops (`for` loop)

Example:

```
for(int i = 1; i <= 5; i++) {
    printf("%d\n", i);
}
```

Explanation:

- Starts from 1
- Prints until 5
- Increases by 1 each time

How many times will the loop run?

Answer: 5 times (for values 1 to 5)

What is an Array?

An array is a **collection of variables** of the **same type**, stored together in memory.

Syntax:

```
data_type array_name[size];
```

Example: One-Dimensional Array

```
#include <stdio.h>

int main() {
    int numbers[5] = {10, 20, 30, 40, 50};

    for(int i = 0; i < 5; i++) {
        printf("Element %d: %d\n", i, numbers[i]);
    }

    return 0;
}
```

Output:

```
Element 0: 10
Element 1: 20
...
```

Key Points:

- Index starts from **0**
- You can access elements using: `array_name[index]`
- You can also initialize all elements manually or with a loop

2D Arrays (Matrices)

Example:

```
#include <stdio.h>

int main() {
    int matrix[2][3] = {{1, 2, 3}, {4, 5, 6}};

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

    return 0;
}
```

Functions in C

What is a Function?

A function is a block of code that performs a specific task. It helps make code reusable and modular.

Syntax of a Function

```
return_type function_name(parameters) {
    // code
    return value;
}
```

Example: Function to Add Two Numbers

```
#include <stdio.h>

int add(int a, int b) {
    return a + b;
}

int main() {
    int result = add(5, 3);
```

```
    printf("Sum = %d", result);
    return 0;
}
```

Key Terms:

- **Function Declaration**: Tells the compiler about the function
- **Function Definition**: Actual body/code of the function
- **Function Call**: When you use the function

Q: What will this return?

```
int multiply(int x, int y) {
    return x * y;
}
```

A: It will return the product of `x` and `y`.

Function with Array as Argument

Example:

```
#include <stdio.h>

void printArray(int arr[], int size) {
    for(int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
}

int main() {
    int nums[3] = {1, 2, 3};
    printArray(nums, 3);
    return 0;
}
```