📘 **C Programming Notes – AetherCode (Notion Template)**

🖥️ Cover Page

Course Title: C Programming – Master the Fundamentals\ **Prepared By:** AetherCode Team\ **Website:** www.aethercode.com

> *"Code. Notes. Clarity."*

🕐 Table of Contents

Introduction

C is a foundational language developed by Dennis Ritchie in 1972. It's renowned for system-level programming and underpins the development of modern operating systems, compilers, and embedded software.

💡 Features of C

- **Procedural** and structured
- **Fast execution** due to compiled nature
- **Portability** across platforms
- **Modular programming** with functions
- **Extensive standard library**
- **Low-level memory access** with pointers

🧰Structure of a C Program

```c
#include <stdio.h>

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

Breakdown:

- `#include <stdio.h>`: Preprocessor directive
- `main()`: Entry point
- `printf()`: Outputs data
- `return 0`: Exit status

📊Data Types

Type	Size	Example
int	2 or 4B	int num = 10;
float	4B	float pi = 3.14;
char	1B	char ch = 'A';
double	8B	double e = 2.71828;

Primary, Derived, Enum, Void

🧾Variables and Constants

🕰️Variable: A named memory location.

```c
int marks = 90;
```

🕰️Constant: Immutable value.

```c
const float pi = 3.14;
```

Operators

- **Arithmetic:** `+ - * / %`

- **Relational:** `== != > < >= <=`
- **Logical:** `&& || !`
- **Bitwise:** `& | ^ ~ << >>`
- **Assignment:** `= += -= *= /=`
- **Increment/Decrement:** `++ --`

🕐Control Flow

Conditional Statements

```c
if (a > b) {
    printf("A is greater");
} else {
    printf("B is greater");
}
```

Looping

```c
for (int i = 0; i < 5; i++) {
    printf("%d", i);
}
```

Jump Statements: `break`, `continue`, `goto`

🔧Functions

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

- Return type
- Function name
- Parameters
- `main()` is also a function

🧮Arrays

```c
int numbers[5] = {1, 2, 3, 4, 5};
```

- 1D, 2D, and multidimensional arrays

Strings

```
char str[] = "Hello";
```

- Functions: `strlen()`, `strcpy()`, `strcat()`

🕐Pointers

```
int a = 5;
int *p = &a;
```

- `*` for dereferencing
- `&` for address
- Pointer to pointer, dynamic memory: `malloc()`, `free()`

📦Structures & Unions

```
struct Student {
  int id;
  char name[30];
};
```

- `union` shares memory, `struct` allocates separate space

📁File Handling

```
FILE *fp = fopen("test.txt", "w");
if (fp != NULL) {
    fprintf(fp, "Hello File");
    fclose(fp);
}
```

- Modes: `r`, `w`, `a`, `r+`, etc.
- Functions: `fopen()`, `fclose()`, `fgets()`, `fprintf()`

🗂️Storage Classes

Type	Scope	Lifetime
auto	Local	Function block

Type	Scope	Lifetime
static	Local/Global	Entire program
extern	Global	Whole program
register	CPU Register	Fast access

🛠️ Preprocessor Directives

- `#define PI 3.14`
- `#include <stdio.h>`
- `#ifdef`, `#ifndef`, `#endif`

🗜️ Applications

- UNIX/Linux kernel
- Compiler design
- Embedded development
- Databases (MySQL core)
- Firmware in electronics

🔗 Summary

C teaches low-level memory management, modular design, and builds a strong foundation for understanding how computers operate internally. It's crucial for systems programming and continues to influence modern languages.

Next: Proceed to C++ for OOP, Classes, and STL mastery.

© AetherCode – Code. Notes. Clarity.