1. Introduction to C Programming

1.1 History of C

- Developed by Dennis Ritchie at Bell Labs in 1972
- Evolved from B language (based on BCPL)
- Used to rewrite UNIX OS

1.2 Features

- Procedural Language
- Fast and Efficient
- Low-Level Manipulation (pointers)
- Portable
- Rich Library

1.3 Applications

- Operating Systems
- Compilers
- Embedded Systems
- System Software
- Game Development

1.4 Advantages

- Simplicity
- Speed
- Portability
- Large Community Support

1.5 Disadvantages

- No Object-Oriented Support
- Manual Memory Management
- No Exception Handling

2. Setting up C Environment

2.1 IDEs and Compilers

- Windows: Code::Blocks, Turbo C, Dev C++
- Linux: gcc, Vim
- Online: https://www.programiz.com/c-programming/online-compiler/

2.2 First Program

```c
#include <stdio.h>

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

---

3. Basic Structure of a C Program

- Preprocessor Commands
- Functions
- Variables and Declarations
- Statements and Expressions

---

4. Data Types and Variables

4.1 Data Types

- int, float, double, char
- Derived: array, pointer, structure

4.2 Variable Declaration

```c
int age = 25;
float height = 5.9;
char grade = 'A';
```

---

5. Operators in C

- Arithmetic (+, -, *, /, %)
- Relational (==, !=, >, <, >=, <=)
- Logical (&&, ||, !)
- Assignment (=, +=, -=, *=, /=)
- Increment/Decrement (++ --)

---

6. Input and Output

```c
#include <stdio.h>

int main() {
    int age;
    printf("Enter your age: ");
    scanf("%d", &age);
    printf("You entered: %d\n", age);
    return 0;
}
```

---

7. Control Statements

7.1 if / if-else / nested if

```c
if (age >= 18) {
    printf("Adult\n");
} else {
    printf("Minor\n");
}
```

7.2 switch

```c
switch(choice) {
    case 1: printf("Start\n"); break;
    case 2: printf("Stop\n"); break;
    default: printf("Invalid\n");
}
```

8. Loops in C

8.1 for Loop

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

8.2 while Loop

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

8.3 do-while Loop

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

---

9. Functions

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

int main() {
    int result = add(3, 4);
    printf("Sum = %d\n", result);
    return 0;
}
```

---

10. Arrays

```c
int arr[5] = {1, 2, 3, 4, 5};
for (int i = 0; i < 5; i++) {
    printf("%d ", arr[i]);
}
```

---

11. Strings

```c
char name[20];
printf("Enter name: ");
scanf("%s", name);
printf("Hello, %s\n", name);
```

---

12. Pointers

```c
int a = 10;
int *p = &a;
printf("Value: %d, Address: %p\n", *p, p);
```

---

13. Structures

```c
struct Student {
    int id;
    char name[20];
};

struct Student s1 = {1, "John"};
printf("%d %s", s1.id, s1.name);
```

---

14. File Handling

```c
FILE *fp = fopen("data.txt", "w");
fprintf(fp, "Hello File");
fclose(fp);
```

---

15. Dynamic Memory Allocation

```c
int *p = (int*)malloc(5 * sizeof(int));
if (p == NULL) {
    printf("Memory not allocated");
}
free(p);
```

---

16. Preprocessor Directives

- #define, #include, #ifdef, #endif

```c
#define PI 3.14
printf("PI = %f", PI);
```

---

17. Sample Programs

- Factorial, Prime Number, Palindrome, Fibonacci Series

---

18. Viva Questions

1. What is the difference between call by value and call by reference?
2. What is a pointer?
3. What is the difference between structure and union?
4. What is recursion?
5. What is the use of malloc and free?

---

19. Assignments & Projects

- Create a student record system
- Write a mini calculator
- File-based marksheet management

This guide can be used to prepare slides, handouts, lab work, and exam material.