21-Day C Programming Challenge

Learning Material (Days 1–3)

Beginner-Friendly Student Notes with Examples

This guide is written as if you are learning C for the first time. Read each section in order, type every program yourself, run it, and observe the output.

Day 1 - Introduction to C

What is C?

C is a **general-purpose programming language** developed by **Dennis Ritchie** at Bell Labs in **1972**.

C is called the **mother of programming languages** because many modern languages (C++, Java, C#, etc.) were influenced by it.

Why Learn C?

- Fast programming language
- Foundation for C++, Java, Python
- Used in Operating Systems
- Embedded Systems
- Device Drivers
- Compilers
- Game Engines

Your First C Program

```
#include <stdio.h>

int main()
{
    printf("Hello World");

    return 0;
}
```

Output

```
Hello World
```

Understanding Each Line

Line 1

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

This is called a **Header File**.

It tells the compiler

"Use the Standard Input Output Library."

Without this header,

```
printf();
scanf();
```

will not work.

Line 2

```
int main()
```

This is called the **Main Function**.

Every C program starts execution from

```
main()
```

Think of it as the **main door** of a house.

Line 3

```
{
```

Opening brace

Everything inside the braces belongs to the function.

Line 4

```
printf("Hello World");
```

`printf()` displays information on the screen.

Syntax

```
printf("Message");
```

Example

```
printf("Welcome");
```

Output

```
Welcome
```

Line 5

```
return 0;
```

Means

"The program ended successfully."

Line 6

```
}
```

Closing brace

Ends the function.

Structure of a C Program

```
Header Files

↓

Main Function

↓

Statements

↓

Return Statement
```

General Structure

```
#include<stdio.h>

int main()
{

    statements;

    return 0;
}
```

Character Set of C

C understands different characters.

Alphabets

```
A-Z

a-z
```

Example

```
A
B
X
a
b
z
```

Digits

```
0 1 2 3 4 5 6 7 8 9
```

Special Symbols

```
+
-
*
/
%
=
<
>
!
&
|
^
;
,
.
()
{}
[]
#
```

White Spaces

```
Space

Tab

New Line
```

Keywords

Keywords are **reserved words** that have predefined meanings.

You cannot use them as variable names.

Examples

```
int
float
char
if
else
for
while
switch
break
continue
return
void
struct
union
const
```

Example Program 1

Print Your Name

```
#include<stdio.h>

int main()
{
    printf("My Name is Rahul");

    return 0;
}
```

Example Program 2

Print Address

```
#include<stdio.h>

int main()
{
    printf("Hyderabad");

    return 0;
}
```

Example Program 3

Print College Name

```
#include<stdio.h>

int main()
{
    printf("ABC Engineering College");

    return 0;
}
```

Challenge

Print

```
******
******
******
```

Solution

```
#include<stdio.h>

int main()
{
    printf("******\n");
    printf("******\n");
    printf("******");

    return 0;
}
```

Notice the use of `\n`, which moves the cursor to the next line.

Day 1 Summary

You learned

- What is C
- Structure of C program
- Header files
- main()
- printf()
- return statement
- Character Set
- Keywords

Day 2 - Data Types, Variables and Constants

What is Data?

Data means information.

Examples

```
10

98.5

'A'

"Hello"
```

Data Types : Different types of data need different data types.

Data Type	Meaning	Example
int	Integer	25
float	Decimal	25.50
char	Single Character	'A'
double	Large Decimal	45.6789

Integer

```
int age = 20;
```

Stores whole numbers.

Examples

```
10
25
100
-5
```

Float

```
float salary = 25000.50;
```

Stores decimal numbers.

Examples

```
12.5
20.75
100.25
```

Character

```
char grade = 'A';
```

Stores only **one character**.

Correct

```
'A'
'B'
'9'
```

Wrong : `"ABC"`

(Double quotes are for strings.)

Variables

A variable is a **named memory location** used to store data.

Example

```
int age = 20;
```

Here

```
age
```

is the variable.

Rules for Variable Names

Correct

```
age
```

```
marks
```

```
student1
```

```
total_marks
```

Wrong

```
1age
```

```
float
```

```
int
```

```
student name
```

Constants

A constant's value cannot be changed.

Example

```
const float PI = 3.14159;
```

Printing Variables

```
#include<stdio.h>

int main()
{
    int age = 21;

    printf("%d", age);

    return 0;
}
```

Output

```
21
```

Format Specifiers

Data Type	Format
int	%d
float	%f
char	%c

Example

```
printf("%d",10);

printf("%f",12.5);

printf("%c",'A');
```

Program 1

Store Age

```
#include<stdio.h>

int main()
{
    int age = 19;

    printf("Age = %d", age);

    return 0;
}
```

Program 2 : Store Marks

```
#include<stdio.h>

int main()
{
    float marks = 95.5;

    printf("Marks = %.2f", marks);

    return 0;
}
```

Program 3 : Swap Two Numbers (Using Third Variable)

```
#include<stdio.h>

int main()
{
    int a = 10;
    int b = 20;
    int temp;

    temp = a;
    a = b;
    b = temp;

    printf("%d %d", a, b);

    return 0;
}
```

Output : `0 10`

Program 4 : Area of Rectangle

Formula

```
Area = Length × Breadth
#include<stdio.h>

int main()
{
    int l = 10;
    int b = 5;

    int area = l * b;

    printf("Area = %d", area);

    return 0;
}
```

Challenge

Find Area and Perimeter of Rectangle

Formula

```
Area = l × b
```

```
Perimeter = 2(l+b)
```

Try writing this program on your own before checking the solution.

Day 2 Summary

You learned

- Data Types
- Variables
- Constants
- Format Specifiers
- Storing values
- Printing variables

Day 3 - Operators

Operators perform operations on data.

Example

```
10 + 20
```

Here

```
+
```

is an operator.

Arithmetic Operators

Operator	Meaning
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus (Remainder)

Example

```
#include<stdio.h>

int main()
{
    int a = 10;
    int b = 3;

    printf("Addition = %d\n", a + b);
    printf("Subtraction = %d\n", a - b);
    printf("Multiplication = %d\n", a * b);
    printf("Division = %d\n", a / b);
    printf("Remainder = %d\n", a % b);

    return 0;
}
```

Output

```
Addition = 13
Subtraction = 7
Multiplication = 30
Division = 3
Remainder = 1
```

Note: Integer division (`10 / 3`) gives `3`, not `3.333...`, because both operands are integers.

Assignment Operator

Assigns a value to a variable.

Example

```
int x;

x = 10;
```

Compound assignment operators:

```
x += 5;   // x = x + 5
x -= 2;   // x = x - 2
x *= 3;   // x = x * 3
x /= 2;   // x = x / 2
x %= 3;   // x = x % 3
```

Increment Operator

Increases a value by 1.

```
int a = 10;

a++;

printf("%d", a);
```

Output : `11`

Decrement Operator

Decreases a value by 1.

```
int a = 10;

a--;

printf("%d", a);
```

Output

```
9
```

Pre-Increment vs Post-Increment

```
int a = 5;

printf("%d\n", ++a);  // Output: 6

a = 5;

printf("%d\n", a++);  // Output: 5
printf("%d", a);      // Output: 6
```

- `++a` increments first, then uses the value.
- `a++` uses the current value first, then increments.

Program 1 - Simple Calculator

```
#include<stdio.h>

int main()
{
    int a = 20;
    int b = 10;

    printf("Addition = %d\n", a + b);
    printf("Subtraction = %d\n", a - b);
    printf("Multiplication = %d\n", a * b);
    printf("Division = %d\n", a / b);

    return 0;
}
```

Program 2 - Celsius to Fahrenheit

Formula

```
F = (C × 9/5) + 32
#include<stdio.h>

int main()
{
    float c = 30;
    float f = (c * 9 / 5) + 32;

    printf("Fahrenheit = %.2f", f);

    return 0;
}
```

Program 3 - Fahrenheit to Celsius

Formula

```
C = (F - 32) × 5/9
#include<stdio.h>

int main()
{
    float f = 86;
    float c = (f - 32) * 5 / 9;

    printf("Celsius = %.2f", c);

    return 0;
}
```

Challenge

Convert a given number of days into years, months, and days.

Assume:

- 1 year = 365 days
- 1 month = 30 days

Hint:

1. Years = `days / 365`
2. Remaining days = `days % 365`
3. Months = `remaining_days / 30`
4. Days = `remaining_days % 30`

Try to solve it before looking at a solution.

Practice Exercises (Days 1–3)

1. Print your name, course, and city on separate lines.
2. Store your age, height, and grade, then print them.
3. Find the sum, difference, product, quotient, and remainder of two numbers.
4. Calculate the area and perimeter of a square.
5. Swap two numbers using a third variable.
6. Convert kilometers to meters.
7. Convert minutes into hours and minutes.
8. Convert Celsius to Fahrenheit and vice versa.
9. Demonstrate the difference between `++a` and `a++`.
10. Solve the "days into years, months, and days" challenge.

These three days establish the foundation for the remaining 18 days. Once you are comfortable with variables, operators, and the basic structure of a C program, learning input/output, conditions, loops, arrays, functions, and pointers becomes much easier.