Lab Experiment- C Language

# Objective:

* To gain practical experience with advanced pointer concepts in C, including pointer arithmetic, pointers and arrays, and function pointers. Master usage of gcc compiler and gdb.

# Materials needed:

* Computer with a C compiler (e.g., GCC)
* Debugger (GDB)
* Text editor or IDE

**Use this template code for this part onwards:** [**template\_code\_Day2.c**](https://drive.google.com/file/d/1RwNPYgFRwIKD7MxhSorviKl5X5jT8bHe/view?usp=drive_link)

# Part 1: Pointer Basics and Arithmetic

## Task 1.1

Create a program that demonstrates basic pointer usage:

* Declare an integer variable and a pointer to it
* Print the value of the variable using both direct access and the pointer
* Modify the value using the pointer and print the new value

## Task 1.2

Implement a function that swaps two integers using pointers

## Task 1.3

Create an array of integers and use pointer arithmetic to:

* Print all elements of the array
* Calculate the sum of all elements
* Reverse the array in-place

# Part 2: Pointers and Arrays/Strings

## Task 2.1 – Custom String Functions

Implement your own versions of:

* strlen()
* strcpy()
* strcmp()

Use pointer arithmetic only (no <string.h>).

## Task 2.2 – String Reversal

Write a program that checks if a given string is a palindrome (case insensitive)

# Part 3 – Preprocessor & File I/O

## Task 3.1 – Preprocessor Macros

Write macros for:

* SQUARE(x)
* MAX(a,b)
* MAX(a,b,c)
* MAX(a,b,c,d)
* TO\_UPPER(c) (convert char to uppercase if lowercase).

Demonstrate with test cases.

## Task 3.2 – File I/O (Text File)

Write a program that:

* Define a struct Student { char name[50]; int roll; float gpa; };
* Store details of 5 students in an array.
* Print the student with the highest GPA.
* Saves them to a text file (students.txt).
* Reads them back and prints.

# Part 4: Advanced Challenge

## Task 4.1

Implement a simple linked list with the following operations:

* Insert a node at the beginning
* Delete a node by value
* Print the list

Part 5: Dynamic Memory Allocation

Task 5.1

Create a program that:

* Dynamically allocates an array of integers
* Allows the user to input the size of the array and its elements
* Calculates and prints the sum and average of the elements

Task 5.2

Implement a function that uses realloc() to extend an existing dynamically allocated array

Task 5.3

Create a simple memory leak detector:

* Write functions to allocate and free memory
* Keep track of allocated memory addresses
* Print a warning if the program ends with unfreed memory

# Final Tasks of C-language:

## Task Y: Sequential Multiplier in C with Unit Tests

* Write C code to implement [Booth's multiplication algorithm](https://www.geeksforgeeks.org/computer-organization-architecture/computer-organization-booths-algorithm/). Write different functions for shifting and adding so that you can later visualize functions call stack.
* The function should take two signed integers as input and return their product.
* Use bit manipulation operators for efficient multiplication.
* Write a test function to verify the correctness of your Booth multiplier function.
* Create test cases for various scenarios, including positive, negative, zero inputs, multiplication by zero, multiplication by 1, and edge cases (e.g., overflow).

## Helping Material:

* Vim:
  + [MIT OpenCourseware](https://missing.csail.mit.edu/2020/editors/)
* C Language:
  + The C Programming Language by Kernighan & Ritchie ([available here](https://pern-my.sharepoint.com/:b:/g/personal/meds_uet_edu_pk/EWeK7XdzgrVFrVWS0ieNZlkB6LoOyWm5QGFpAv6MG8zb6Q?e=cMmL5f))
  + C Tutorial – Tutorialspoint ([available here](https://www.tutorialspoint.com/cprogramming/index.htm))
* GCC:
  + Opensource: [click here](https://opensource.com/article/22/5/gnu-c-compiler)
  + IOFlood: [click here](https://ioflood.com/blog/gcc-linux-command/)
  + GNU GCC Guide: [click here](https://gcc.gnu.org/onlinedocs/gcc/Invoking-GCC.html)
* GDB:
* Geeks for Geeks: [click here](https://www.geeksforgeeks.org/gdb-step-by-step-introduction/)
* Medium: [click here](https://medium.com/@amit.kulkarni/gdb-basics-bf3407593285)
* Baeldung: [click here](https://www.baeldung.com/linux/gdb-debug)
* How to forge: [click here](https://www.howtoforge.com/tutorial/how-to-debug-c-programs-in-linux-using-gdb/)