POINTERS AND FUNCTION CALLS - SUMMARY

Problem 1 - Even Digits in a String

Pointer Type: char *

The function takes a string and prints even digits using pointer-style indexing.

Explanation:

The function receives a string (char str []), which is a pointer to the first character of the array. It iterates through the characters and prints only the even digits.

Problem 2 - Arithmetic Using Pointers

Pointer Type: int *

Performs addition, subtraction, multiplication, and division using pointer dereferencing.

Explanation:

Here, ptr1 and ptr2 are integer pointers. They hold the addresses of num1 and num2, respectively. Arithmetic is done by dereferencing the pointers.

Problem 3 - Swapping Using Pass by Reference

Pointer Type: int *

Swaps two variables by passing their addresses to the function.

Explanation:

swap (&a, &b); passes the **addresses** of a and b. This is **pass by reference**, meaning the actual variables in main () are modified.

Problem 4 - Different Pointer Types

Pointer Types: int *, char *, float *, long *

Demonstrates how to declare and use pointers of different data types.

Explanation:

Each variable has a corresponding pointer. You can access values using *ptrX and addresses using ptrX. This shows how pointers work with various data types.

Problem 5 – *String Length Using Array Traversal*

Concept: String (character array) traversal with pointer-style indexing.

Explanation:
The function accepts a string (which is a pointer to a character array). The array is accessed via indexing, effectively using pointer arithmetic.

Feature	Pass by Value	Pass by Reference
What is passed?	A copy of the variable	Address of the variable
Usage	Basic calculations, constants	When you want to change real values