SWT11022: Practical for Fundamentals of Programming

Department of Information & Communication Technology
Faculty of Technology
South Eastern University of Sri Lanka

Time: - 08.30am - 10.30 am Labsheet 12

Title: Introduction to the Pointers

Objective:

- Learn the syntax and structure of pointers.
- Understand the usage of pointers for storing address.
- Understand accessing pointers.

Pointers in C

What is a Pointer?

- A pointer is a variable that stores the memory address of another variable as its value.
- Instead of storing a value directly, it stores where that value is located in memory.

Why Use Pointers?

- To directly access memory
- To work with arrays and strings efficiently
- To pass large variables to functions (by reference)
- To work with dynamic memory (e.g., malloc)

Declaration: Use the * symbol before the data type to declare a pointer.

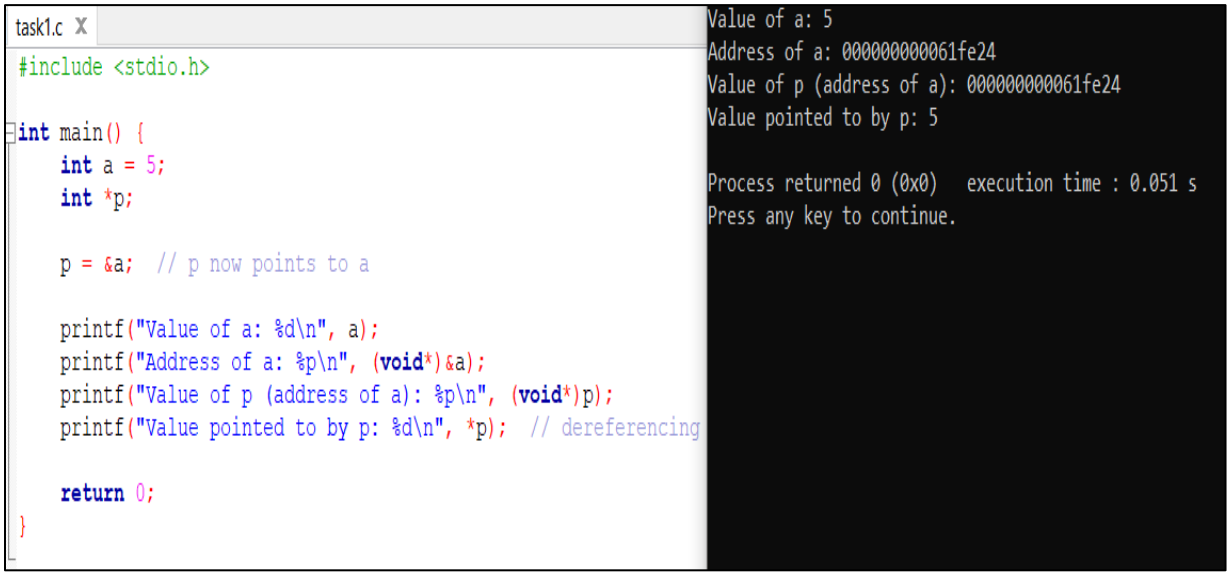
```
int *ptr;  // pointer to int
```

Initialization: Assign the memory address of a variable using the address-of operator (&).

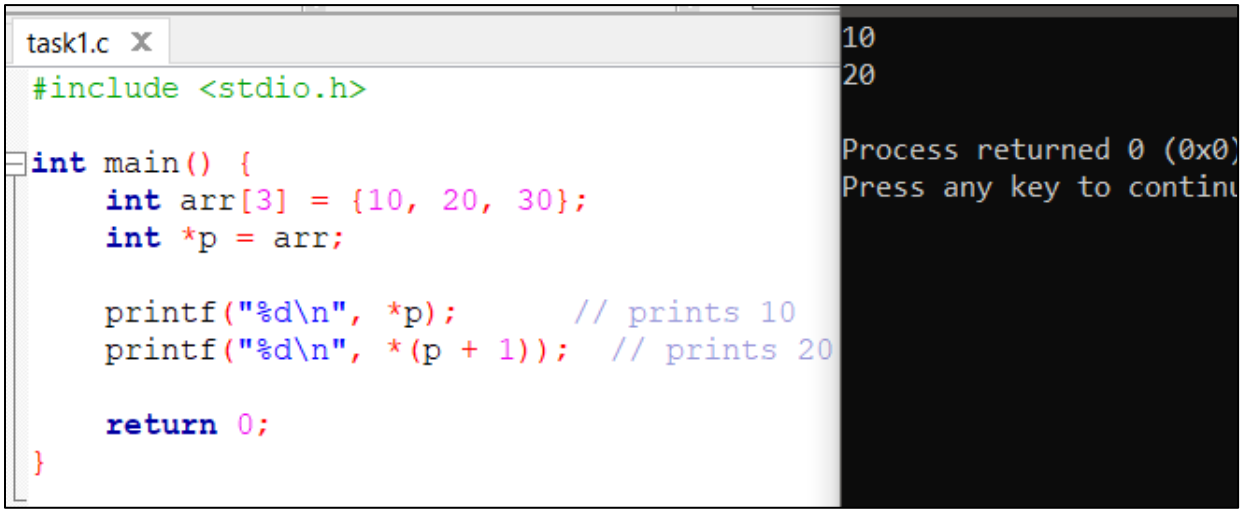
```
int a = 10;
ptr = &a;  // now ptr holds the address of a
```

Dereferencing: Access the value stored at the memory address pointed to by the pointer using the dereference operator (*).

printf("%d", *ptr); // prints 10, which is the value of a



Pointer Arithmetic: Add or subtract integers to a pointer to move it within the same memory block (use with caution). We can move a pointer forward or backward in memory using arithmetic.



- p + 1 moves the pointer to the next element of the array.

Pointers and Arrays: Arrays and pointers are closely related.

task1.c

```c
#include <stdio.h>

int main() {

    int arr[3] = {1, 2, 3};
    int *p = arr;  // same as &arr[0]

    printf("%d\n", *p);       // 1
    printf("%d\n", *(p + 1)); // 2

    return 0;
}
```

"G:\Demo\C\2025 labsheets\Updated\Labweek11\

```
1
2

Process returned 0 (0x0)   execution time
Press any key to continue.
```

Task: Swap Two Numbers Using Pointers

task1.c

```c
#include <stdio.h>

int main() {
    int num1 = 10, num2 = 20;
    int *ptr1, *ptr2;
    int temp;

    // Initialize pointers
    ptr1 = &num1;
    ptr2 = &num2;

    // Print original values
    printf("Before swapping:\n");
    printf("num1 = %d, num2 = %d\n", num1, num2);

    // Swapping using pointers
    temp = *ptr1;     // Store value at ptr1 (num1) in temp
    *ptr1 = *ptr2;    // Copy value at ptr2 (num2) to where ptr1 points (num1)
    *ptr2 = temp;     // Copy value from temp to where ptr2 points (num2)

    // Print swapped values
    printf("\nAfter swapping:\n");
    printf("num1 = %d, num2 = %d\n", num1, num2);

    return 0;
}
```

Exercise 01 :

1. Write a C program to following scenario.

a) Declare and initialize variables int_var, float_var, char_var with data types int, float and char respectively.

int_var = 45;

Float_var = 57.234;

Char_var = ‘A’;

b) Print the values of int_var, float_var, char_var.

c) Print the address of above variables.

d) Declare the pointers int_var_pointer, float_var_pointer, char_var_pointer with the data type int, float, char respectively.

e) Initialize the above pointers int_var_pointer, float_var_pointer and char_var_pointer with address of int_var, float_var and char_var respectively.

f) Print pointer variables.

g) Print value of int_var, float_var and char_var by dereferencing the address by using pointers.

h) Perform the following arithmetic operatiobns with pointers.

- int_var_pointer+1;
- float_var_pointer+1;
- char_var_pointer+1;

Exercise 02 :

1. **Write a C program to get 10 integer numbers from user and find the sum of 10 numbers using arrays and pointers**

Discussion :

- Dereferencing
- Pointer arithmetic

24/04/2025

Report Submission Guidelines

- Submit the **Report** by **28/04/2025.**
- Late submissions will not be accepted.

Report Structure

- Practical No
- Date of Submission
- Title
- Objective of the practical.
- Exercise
- Challenges
- Discussion
- References