1. Introduction to Pointers

1.1 What are Pointers?

A **pointer** is a variable that stores the memory address of another variable. Instead of holding a data value directly, a pointer "points to" the location in memory where the data is stored.

Analogy: Think of computer memory like a street with houses:

- Each house (memory location) has an address (memory address)
- Each house contains something (data value)
- A pointer is like writing down a house address on paper
- You can use that address to find and access the house

Why Use Pointers?

1. **Dynamic Memory Allocation:** Create variables at runtime
2. **Efficient Array/String Manipulation:** Access elements without copying
3. **Function Parameter Passing:** Modify variables from within functions
4. **Data Structures:** Build linked lists, trees, graphs, etc.
5. **System Programming:** Direct memory access and hardware interaction

1.2 Memory Addresses

Every variable in C is stored at a specific memory location, identified by a unique address.

```
#include <stdio.h>

int main() {
    int age = 25;
    float height = 5.9f;
    char grade = 'A';

    printf("Value of age: %d\n", age);
    printf("Address of age: %p\n", (void*)&age);

    printf("Value of height: %.1f\n", height);
    printf("Address of height: %p\n", (void*)&height);
```

```
    printf("Value of grade: %c\n", grade);
    printf("Address of grade: %p\n", (void*)&grade);

    return 0;
}

/* Output (addresses will vary):
Value of age: 25
Address of age: 0x7ffd5c8e4a3c
Value of height: 5.9
Address of height: 0x7ffd5c8e4a38
Value of grade: A
Address of grade: 0x7ffd5c8e4a37
*/
```

Key Points:

- Memory addresses are typically displayed in hexadecimal (base 16)
- The `&` operator gets the address of a variable
- Format specifier `%p` prints pointer/address values
- Adjacent variables may have addresses close to each other

Revision #1
Created 2025-10-01 03:21:17 UTC by DS
Updated 2025-10-01 03:24:57 UTC by DS