3. Array Indexing and Access

3.1 Basic Indexing

Python vs C Indexing:

Operation	Python	C
First element	`list[0]`	`array[0]`
Last element	`list[-1]`	`array[size-1]`
Nth element	`list[n]`	`array[n]`
Modify element	`list[0] = 10`	`array[0] = 10;`

3.1.1 Valid Indexing Example

```
int numbers[5] = {10, 20, 30, 40, 50};

printf("First element: %d\n", numbers[0]);    // Output: 10
printf("Third element: %d\n", numbers[2]);    // Output: 30
printf("Last element: %d\n", numbers[4]);     // Output: 50

// Modifying elements
numbers[1] = 99;
printf("Modified second element: %d\n", numbers[1]);  // Output: 99
```

3.1.2 Index Bounds and Common Errors

Critical Difference from Python:

```
# Python - Safe bounds checking
my_list = [1, 2, 3, 4, 5]
print(my_list[10])  # Raises IndexError: list index out of range
```

```
// C - NO automatic bounds checking!
int my_array[5] = {1, 2, 3, 4, 5};
printf("%d\n", my_array[10]);  // Undefined behavior! May print garbage
my_array[10] = 999;           // Buffer overflow! May crash program
```

Common Index-Related Errors:

1. **Off-by-One Error:**

```
int arr[5] = {1, 2, 3, 4, 5};
// WRONG: Accessing index 5 (valid indices: 0-4)
for (int i = 0; i <= 5; i++) {    // ERROR: i goes up to 5
    printf("%d ", arr[i]);
}

// CORRECT:
for (int i = 0; i < 5; i++) {     // i goes from 0 to 4
    printf("%d ", arr[i]);
}
```

2. **Negative Index Error:**

```
int arr[5] = {1, 2, 3, 4, 5};
int index = -1;
printf("%d\n", arr[index]);  // Undefined behavior! C has no negative indexing
```

3. **Uninitialized Index:**

```
int arr[10];
int i;                      // Uninitialized variable
printf("%d\n", arr[i]);     // Using uninitialized i as index - dangerous!
```

3.2 Safe Array Access Patterns

3.2.1 Bounds Checking Function

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

bool is_valid_index(int index, int array_size) {
    return (index >= 0 && index < array_size);
}

int safe_access(int arr[], int size, int index) {
    if (is_valid_index(index, size)) {
        return arr[index];
```

```
    } else {
        printf("Error: Index %d out of bounds (0-%d)\n", index, size-1);
        return -1;  // Return error value
    }
}

int main() {
    int numbers[5] = {10, 20, 30, 40, 50};

    printf("Safe access: %d\n", safe_access(numbers, 5, 2));   // Valid
    printf("Safe access: %d\n", safe_access(numbers, 5, 10));  // Invalid

    return 0;
}
```

Revision #1
Created 2025-09-15 00:53:25 UTC by DS
Updated 2025-09-15 00:55:05 UTC by DS