7. Character Arrays and Strings

7.1 Character Arrays vs Strings

Understanding C Strings: In C, strings are arrays of characters terminated by a null character (`'\0'`).

```
// Character array (not necessarily a string)
char letters[5] = {'H', 'e', 'l', 'l', 'o'};

// String (null-terminated character array)
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};

// Easier string initialization
char message[] = "Hello";  // Automatically adds '\0'
char name[20] = "Alice";   // name[0]='A', name[1]='l', ..., name[5]='\0'
```

7.2 String Input/Output

7.2.1 String Input Methods

```
#include <stdio.h>

int main() {
    char name[50];

    // Method 1: scanf (stops at whitespace)
    printf("Enter your first name: ");
    scanf("%s", name);  // No & needed for arrays

    // Method 2: fgets (reads entire line)
    printf("Enter your full name: ");
    fgets(name, sizeof(name), stdin);

    // Method 3: scanf with character set
    printf("Enter your name: ");
```

```
    scanf("%[^\n]", name);  // Read until newline

    printf("Hello, %s!\n", name);
    return 0;
}
```

7.2.2 String Output

```
char message[] = "Programming in C";

// Method 1: printf with %s
printf("Message: %s\n", message);

// Method 2: puts (automatically adds newline)
puts(message);

// Method 3: Character by character
for (int i = 0; message[i] != '\0'; i++) {
    printf("%c", message[i]);
}
printf("\n");
```

7.3 String Manipulation Functions

7.3.1 String Length

```
#include <string.h>

// Using library function
int len = strlen(str);

// Manual implementation
int string_length(char str[]) {
    int length = 0;
    while (str[length] != '\0') {
        length++;
    }
    return length;
}
```

7.3.2 String Copy

```
#include <string.h>

// Using library function
strcpy(destination, source);

// Manual implementation
void string_copy(char dest[], char src[]) {
    int i = 0;
    while (src[i] != '\0') {
        dest[i] = src[i];
        i++;
    }
    dest[i] = '\0';  // Don't forget null terminator!
}
```

7.3.3 String Concatenation

```
#include <string.h>

// Using library function
strcat(destination, source);

// Manual implementation
void string_concatenate(char dest[], char src[]) {
    int dest_len = string_length(dest);
    int i = 0;

    while (src[i] != '\0') {
        dest[dest_len + i] = src[i];
        i++;
    }
    dest[dest_len + i] = '\0';
}
```

7.3.4 String Comparison

```
#include <string.h>
```

```
// Using library function
int result = strcmp(str1, str2);
// Returns: 0 if equal, <0 if str1 < str2, >0 if str1 > str2

// Manual implementation
int string_compare(char str1[], char str2[]) {
    int i = 0;
    while (str1[i] != '\0' && str2[i] != '\0') {
        if (str1[i] < str2[i]) return -1;
        if (str1[i] > str2[i]) return 1;
        i++;
    }

    if (str1[i] == '\0' && str2[i] == '\0') return 0;
    return (str1[i] == '\0') ? -1 : 1;
}
```

7.4 Common String Operations

7.4.1 Count Characters/Words

```
int count_character(char str[], char ch) {
    int count = 0;
    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] == ch) {
            count++;
        }
    }
    return count;
}

int count_words(char str[]) {
    int words = 0;
    bool in_word = false;

    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] != ' ' && str[i] != '\t' && str[i] != '\n') {
            if (!in_word) {
                words++;
```

```
                in_word = true;
            }
        } else {
            in_word = false;
        }
    }

    return words;
}
```

7.4.2 String Reversal

```
void reverse_string(char str[]) {
    int len = string_length(str);

    for (int i = 0; i < len / 2; i++) {
        char temp = str[i];
        str[i] = str[len - 1 - i];
        str[len - 1 - i] = temp;
    }
}
```

7.4.3 Case Conversion

```
#include <ctype.h>

void to_uppercase(char str[]) {
    for (int i = 0; str[i] != '\0'; i++) {
        str[i] = toupper(str[i]);
    }
}

void to_lowercase(char str[]) {
    for (int i = 0; str[i] != '\0'; i++) {
        str[i] = tolower(str[i]);
    }
}

// Manual implementation for uppercase
void manual_to_uppercase(char str[]) {
    for (int i = 0; str[i] != '\0'; i++) {
```

```
        if (str[i] >= 'a' && str[i] <= 'z') {
            str[i] = str[i] - 'a' + 'A';
        }
    }
}
```

Revision #1
Created 2025-09-15 00:57:57 UTC by DS
Updated 2025-09-15 00:59:15 UTC by DS