string.h Header File in C

The string.h header file in C provides various functions for string manipulation and comparison. Below is an overview with examples for each function:

♠ String Manipulation Functions

1. **strcpy()**

   - **Description**: Copies a string from the source to the destination.
   - **Syntax**: char *strcpy(char *dest, const char *src);

Example:

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

int main() {
    char src[] = "Hello, World!";
    char dest[50];

    strcpy(dest, src);
    printf("Source: %s\n", src);
    printf("Destination: %s\n", dest);

    return 0;
}
```

2. **strncpy()**

   - **Description**: Copies up to n characters from the source to the destination.
   - **Syntax**: char *strncpy(char *dest, const char *src, size_t n);

Example:

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

int main() {
    char src[] = "Hello, World!";
    char dest[50];

    strcpy(dest, src);
    printf("Source: %s\n", src);
    printf("Destination: %s\n", dest);

    return 0;
}
```

3. **strcat()**

   - **Description**: Concatenates (appends) the source string to the destination string.
   - **Syntax**: char *strcat(char *dest, const char *src);

Example:

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

```c
#include <string.h>

int main() {
    char dest[50] = "Hello";
    char src[] = ", World!";

    strcat(dest, src);
    printf("Concatenated String: %s\n", dest);

    return 0;
}
```

4. **strncat()**

   - **Description**: Appends up to n characters from the source string to the destination string.
   - **Syntax**: char *strncat(char *dest, const char *src, size_t n);

Example:

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

int main() {
    char dest[50] = "Hello";
    char src[] = ", World!";

    strncat(dest, src, 5);  // Append only the first 5 characters
    printf("Concatenated String: %s\n", dest);

    return 0;
}
```

♠ String Comparison Functions

1. **strcmp()**

   - **Description**: Compares two strings lexicographically. Returns 0 if equal, a positive value if the first string is greater, or a negative value if the second string is greater.
   - **Syntax**: int strcmp(const char *str1, const char *str2);

Example:

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

int main() {
    char str1[] = "Apple";
    char str2[] = "Banana";

    int result = strcmp(str1, str2);

    if (result == 0) {
        printf("Strings are equal.\n");
    }
    else if (result < 0) {
        printf("'%s' comes before '%s'.\n", str1, str2);
    }
    else {
        printf("'%s' comes after '%s'.\n", str1, str2);
```

```
        }

        return 0;
    }
```

2. **strncmp()**

- **Description**: Compares up to n characters of two strings lexicographically.
- **Syntax**: int strncmp(const char *str1, const char *str2, size_t n);

Example:

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

int main() {
    char str1[] = "Application";
    char str2[] = "Apple";

    int result = strncmp(str1, str2, 3);  // Compare first 3 characters

    if (result == 0) {
        printf("The first 3 characters are equal.\n");
    }
    else if (result < 0) {
        printf("The first 3 characters of '%s' come before '%s'.\n", str1, str2);
    }
    else {
        printf("The first 3 characters of '%s' come after '%s'.\n", str1, str2);
    }

    return 0;
}
```

---------- X ----------