1,
```c
#include <stdio.h>
#include <ctype.h>

void expandString(char *input, char *output) {
    int i = 0, j = 0;

    while (input[i] != '\0') {
        char currentChar = input[i++];
        int count = 0;

        while (isdigit(input[i])) {
            count = count * 10 + (input[i++] - '0');
        }

        for (int k = 0; k < count; k++) {
            output[j++] = currentChar;
        }
    }

    output[j] = '\0';
}

int main() {
    char input[100], output[100];

    printf("Enter the input string: ");
    scanf("%s", input);

    expandString(input, output);

    printf("Output: %s\n", output);
```

```
    return 0;
}
2, #include <stdio.h>
#include <string.h>

void compressString(char *input, char *output) {
    int length = strlen(input);
    int count = 1;

    for (int i = 0; i < length; i++) {
        // If the current character is the same as the next one
        if (input[i] == input[i + 1]) {
            count++;
        } else {
            // Append the character and its count to the output string
            sprintf(output, "%s%c%d", output, input[i], count);
            count = 1; // Reset the count for the next character
        }
    }
}

int main() {
    char input[100], output[100];

    printf("Enter the input string: ");
    scanf("%s", input);

    compressString(input, output);

    printf("Output: %s\n", output);
```

```
    return 0;
}
```

3,
```
#include <stdio.h>

void printNumberInWords(int num) {
    char *ones[] = {"", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine"};
    char *teens[] = {"", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen"};
    char *tens[] = {"", "Ten", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety"};

    if (num < 0 || num > 99999) {
        printf("Number out of range (0-99999)\n");
        return;
    }

    if (num == 0) {
        printf("Zero\n");
        return;
    }

    int thousands = num / 1000;
    int hundreds = (num % 1000) / 100;
    int tensPlace = (num % 100) / 10;
    int onesPlace = num % 10;

    if (thousands > 0) {
        printf("%s Thousand ", ones[thousands]);
    }

    if (hundreds > 0) {
```

```
        printf("%s Hundred ", ones[hundreds]);
    }

    if (tensPlace == 1 && onesPlace > 0) {
        printf("%s ", teens[onesPlace]);
    } else {
        if (tensPlace > 0) {
            printf("%s ", tens[tensPlace]);
        }

        if (onesPlace > 0) {
            printf("%s ", ones[onesPlace]);
        }
    }

    printf("\n");
}

int main() {
    int num;

    printf("Enter a number (0-99999): ");
    scanf("%d", &num);

    printNumberInWords(num);

    return 0;
}
```

4, #include <stdio.h>

#include <string.h>

```
void compareStrings(char *str1, char *str2) {
    int length = strlen(str1);

    printf("Output:\n");
    for (int i = 0; i < length; i++) {
        if (str1[i] != str2[i]) {
            printf("%c, %c\n", str1[i], str2[i]);
        }
    }
}

int main() {
    char str1[100], str2[100];

    printf("Enter the first string: ");
    scanf("%s", str1);

    printf("Enter the second string: ");
    scanf("%s", str2);

    if (strlen(str1) != strlen(str2)) {
        printf("Error: Strings are not of equal length.\n");
        return 1;
    }

    compareStrings(str1, str2);

    return 0;
}
```

5, #include <stdio.h>

#include <string.h>

```
void justifyText(char *text, int length, int padding) {
    int spaceCount = padding - strlen(text);
    int wordCount = 0;

    // Count the number of words in the text
    for (int i = 0; text[i] != '\0'; i++) {
        if (text[i] == '_') {
            wordCount++;
        }
    }

    // Calculate the number of spaces between words
    int spacesBetweenWords = (wordCount > 1) ? spaceCount / (wordCount - 1) : spaceCount;

    // Calculate the remaining spaces after distributing evenly
    int remainingSpaces = (wordCount > 1) ? spaceCount % (wordCount - 1) : spaceCount;

    // Print the justified text
    for (int i = 0; text[i] != '\0'; i++) {
        if (text[i] == '_') {
            // Print spaces between words
            for (int j = 0; j < spacesBetweenWords; j++) {
                printf(" ");
            }

            // Print remaining spaces, if any
            if (remainingSpaces > 0) {
                printf(" ");
                remainingSpaces--;
            }
```

```
        } else {
            printf("%c", text[i]);
        }
    }

    printf("\n");
}

int main() {
    char text[100];
    int padding;

    printf("Enter the text: ");
    scanf("%s", text);

    printf("Enter the desired padding: ");
    scanf("%d", &padding);

    justifyText(text, strlen(text), padding);

    return 0;
}
```

6,
```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

// Function to check if a character is a special character
int isSpecialChar(char ch) {
    return !(isalpha(ch) || isdigit(ch));
}
```

```
// Function to check if a string is a palindrome
int isPalindrome(char *str) {
    int left = 0;
    int right = strlen(str) - 1;

    while (left < right) {
        // Skip special characters from the left
        while (left < right && isSpecialChar(str[left])) {
            left++;
        }

        // Skip special characters from the right
        while (left < right && isSpecialChar(str[right])) {
            right--;
        }

        // Compare the actual characters
        if (tolower(str[left]) != tolower(str[right])) {
            return 0; // Not a palindrome
        }

        left++;
        right--;
    }

    return 1; // Palindrome
}

int main() {
    char str[100];
```

```
    printf("Enter the string: ");
    fgets(str, sizeof(str), stdin);

    // Remove the newline character from the input
    if (str[strlen(str) - 1] == '\n') {
        str[strlen(str) - 1] = '\0';
    }

    if (isPalindrome(str)) {
        printf("Output: True\n");
    } else {
        printf("Output: False\n");
    }

    return 0;
}
```

7,
```
#include <stdio.h>
#include <string.h>

// Function to swap characters at position i and j in the string
void swap(char *str, int i, int j) {
    char temp = str[i];
    str[i] = str[j];
    str[j] = temp;
}

// Function to generate permutations of a string
void generatePermutations(char *str, int start, int end) {
    if (start == end) {
        printf("%s\n", str);
    } else {
```

```
        for (int i = start; i <= end; i++) {
            swap(str, start, i);
            generatePermutations(str, start + 1, end);
            swap(str, start, i); // Backtrack
        }
    }
}

int main() {
    char str[100];

    printf("Enter the string: ");
    scanf("%s", str);

    printf("Output:\n");
    generatePermutations(str, 0, strlen(str) - 1);

    return 0;
}
8, #include <stdio.h>
#include <string.h>

// Function to find mismatched substrings
void findMismatchedSubstrings(char *str1, char *str2) {
    int len1 = strlen(str1);
    int len2 = strlen(str2);

    printf("Output:\n");

    for (int i = 0; i < len1 && i < len2; i++) {
        if (str1[i] != str2[i]) {
```

```
            int j = i;
            while (j < len1 && j < len2 && str1[j] != str2[j]) {
                printf("%c", str1[j]);
                j++;
            }
            printf(" ");
            i = j - 1;
        }
    }
    printf("\n");
}

// Function to count mismatched characters
void countMismatchedCharacters(char *str1, char *str2) {
    int len1 = strlen(str1);
    int len2 = strlen(str2);

    printf("Character Counts:\n");

    for (int i = 0; i < len1 && i < len2; i++) {
        if (str1[i] != str2[i]) {
            printf("%c: %d\n", str1[i], i);
        }
    }
}

int main() {
    char str1[100], str2[100];

    printf("Enter the first string: ");
    scanf("%s", str1);
```

```
    printf("Enter the second string: ");
    scanf("%s", str2);

    if (strlen(str1) != strlen(str2)) {
        printf("Error: Strings are not of equal length.\n");
        return 1;
    }

    findMismatchedSubstrings(str1, str2);
    countMismatchedCharacters(str1, str2);

    return 0;
}
```

9,

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

int countVowels(char *str) {
    int vowelsCount = 0;
    int length = strlen(str);

    for (int i = 0; i < length; i++) {
        char currentChar = str[i];

        // Check if the character is a vowel (considering both uppercase and lowercase)
        if (currentChar == 'a' || currentChar == 'e' || currentChar == 'i' || currentChar == 'o' ||
currentChar == 'u' ||
            currentChar == 'A' || currentChar == 'E' || currentChar == 'I' || currentChar == 'O' ||
currentChar == 'U') {
            vowelsCount++;
        }
    }
```

```
    return vowelsCount;
}

int main() {
    char str[100];

    printf("Enter the string: ");
    scanf("%s", str);

    int vowelsCount = countVowels(str);

    printf("Number of vowels in the string: %d\n", vowelsCount);

    return 0;
}
10, #include <stdio.h>
#include <string.h>

// Function to find the next palindrome number
void getNextPalindrome(char *num) {
    int length = strlen(num);
    int mid = length / 2;
    int i, j;

    // Check if the number has all 9s
    int allNines = 1;
    for (i = 0; i < length; i++) {
        if (num[i] != '9') {
            allNines = 0;
            break;
```

```
    }
}

if (allNines) {
    // If all digits are 9, increment the number and add 1 at the beginning and end
    printf("Output: 1");
    for (i = 0; i < length - 1; i++) {
        printf("0");
    }
    printf("1\n");
    return;
}

// If the number has an even length
if (length % 2 == 0) {
    i = mid - 1;
    j = mid;
}
// If the number has an odd length
else {
    i = mid - 1;
    j = mid + 1;
}

// Check for symmetry and increment if necessary
while (i >= 0 && j < length && num[i] == num[j]) {
    i--;
    j++;
}

// If the left half is greater than the right half or we reached the middle digit in odd length
```

```
    int isLeftGreater = (i < 0 || num[i] > num[j]);

    // Copy the left half to the right half to make it a palindrome
    while (i >= 0) {
        num[j] = num[i];
        i--;
        j++;
    }

    // Increment the middle digit and carry if necessary
    if (isLeftGreater) {
        int carry = 1;
        i = (length % 2 == 0) ? mid - 1 : mid;

        // Process the carry
        while (i >= 0 && carry > 0) {
            int digit = num[i] - '0' + carry;
            num[i] = (digit % 10) + '0';
            carry = digit / 10;
            i--;
        }
    }

    printf("Output: %s\n", num);
}

int main() {
    char num[100];

    printf("Enter the number: ");
    scanf("%s", num);
```

```
    getNextPalindrome(num);


    return 0;
}
```