Problem 1: Palindrome Checker
Problem Statement:
Write a C program to check if a given string is a palindrome. A string is considered a palindrome if it reads the same backward as forward, ignoring case and non-alphanumeric characters. Use functions like strlen(), tolower(), and isalpha().
Example:
Input: "A man, a plan, a canal, Panama"
Output: "Palindrome"

```c
#include<stdio.h>
#include<string.h>
#include<ctype.h>
int main(){
    char s[100];
    char r[100];
    printf("Enter a String::");
    scanf("%s",s);
    int l=strlen(s);
    for(int i=0;i<l;i++){
        r[i]=s[l-i-1];
    }
    r[l]='\0';
    for (int i = 0; i < l; i++) {
        s[i] = tolower(s[i]);
        r[i] = tolower(r[i]);
    }
    if(strcmp(r,s)!=0){
        printf("\nNot Palindrome");

    }
    else{
        printf("\n Palindrome");
    }
}
```

==========

Problem 2: Word Frequency Counter
Problem Statement:
Write a program to count the frequency of each word in a given string. Use strtok() to tokenize the string and strcmp() to compare words. Ignore case differences.

Example:
Input: "This is a test. This test is simple."
Output:
Word: This, Frequency: 2
Word: is, Frequency: 2
Word: a, Frequency: 1
Word: test, Frequency: 2
Word: simple, Frequency: 1

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>
void toLowerCase(char *str);
int main() {
    char str[1000];
    char words[100][50];
    int freq[100] = {0};
    int wc = 0;

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

    toLowerCase(str);

    char *token = strtok(str, " ");
    while (token != NULL) {
        int found = 0;
        for (int i = 0; i < wc; i++) {
            if (strcmp(words[i], token) == 0) {
                freq[i]++;
                found = 1;
                break;
            }
        }
        if (!found) {
            strcpy(words[wc], token);
            freq[wc] = 1;
            wc++;
        }
        token = strtok(NULL, " ");
    }

    printf("\nWord Frequencies:\n");
    for (int i = 0; i < wc; i++) {
```

```
        printf("Word: %s, Frequency: %d\n", words[i], freq[i]);
    }

    return 0;
}
void toLowerCase(char *str) {
    for (int i = 0; str[i]; i++) {
        str[i] = tolower(str[i]);
    }
}
```

==========

Problem 3: Find and Replace

Problem Statement:

Create a program that replaces all occurrences of a target substring with another substring in a given string. Use strstr() to locate the target substring and strcpy() or strncpy() for modifications.

Example:

Input:

String: "hello world, hello everyone"

Target: "hello"

Replace with: "hi"

Output: "hi world, hi everyone"

```
#include <stdio.h>
#include <string.h>
void findAndReplace(char *str, const char *target, const char
*replacement);
int main() {
    char str[1000], target[100], replacement[100];


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

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

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


    findAndReplace(str, target, replacement);
```

```
    printf("Modified string: %s\n", str);

    return 0;
}
void findAndReplace(char *str, const char *target, const char
*replacement) {
    char result[1000] = "";
    char *pos;
    int targetLen = strlen(target);
    int replaceLen = strlen(replacement);

    while ((pos = strstr(str, target)) != NULL) {

        strncat(result, str, pos - str);


        strcat(result, replacement);


        str = pos + targetLen;
    }


    strcat(result, str);


    strcpy(str, result);
}
```

==========
==========

Problem 4: Reverse Words in a Sentence

Problem Statement:

Write a program to reverse the words in a given sentence. Use strtok() to extract words and strcat() to rebuild the reversed string.

Example:

Input: "The quick brown fox"

Output: "fox brown quick The"

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

```c
#include <string.h>

int main() {
    char str[1000], reversedStr[1000] = "";


    printf("Enter a sentence: ");
    scanf("%[^\n]", str);


    char *token = strtok(str, " ");


    while (token != NULL) {

        char temp[1000];
        strcpy(temp, token);
        strcat(temp, " ");
        strcat(temp, reversedStr);
        strcpy(reversedStr, temp);

        token = strtok(NULL, " ");
    }
```

==========

Problem 5: Longest Repeating Substring

Problem Statement:

Write a program to find the longest substring that appears more than once in a given string. Use strncpy() to extract substrings and strcmp() to compare them.

Example:

Input: "banana"

Output: "ana"

```c
#include <stdio.h>
#include <string.h>
void findLong(char *str);
int main() {
    char str[1000];
    printf("Enter a string: ");
    scanf("%s", str);
    findLong(str);
```

```
    return 0;
}
void findLong(char *str) {
    int n = strlen(str);
    int maxLength = 0;
    char longestSubstring[1000];

    for (int len = 1; len <= n / 2; len++) {
        for (int i = 0; i <= n - len; i++) {
            char substring[1000];
            strncpy(substring, str + i, len);
            substring[len] = '\0';

            for (int j = i + 1; j <= n - len; j++) {
                char compareSubstring[1000];
                strncpy(compareSubstring, str + j, len);
                compareSubstring[len] = '\0';
                if (strcmp(substring, compareSubstring) == 0) {
                    if (len > maxLength) {
                        maxLength = len;
                        strcpy(longestSubstring, substring);
                    }
                    break;
                }
            }
        }
    }
    if (maxLength > 0) {
        printf("Longest repeating substring: %s\n", longestSubstring);
    } else {
        printf("No repeating substring found.\n");
    }
}
```

==========
==========