1.

```c
/*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"*/
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main() {
    char str[100], clean_str[100];
    printf("Enter a string: ");
    scanf("%[^\n]" , str);

    int flag = 0, k = 0;

    for (int i = 0; str[i] != '\0'; i++) {
        if (isalnum(str[i])) {
            clean_str[k++] = tolower(str[i]);
        }
    }
    clean_str[k] = '\0';

    for (int j = 0; j < (strlen(clean_str) / 2); j++) {
        if (clean_str[j] != clean_str[strlen(clean_str) - j- 1]) {
            flag = 1;
            break;
        }
    }



    if (flag == 0) {
        printf("Palindrome\n");
    } else {
        printf("Not a Palindrome\n");
    }
```

```
    return 0;
}
```

```
PS C:\Users\betti\Desktop\Training\Day11> ./task1_v2
Enter a string: A man, a plan, a canal, Panama
Palindrome
```

2.

```
/*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 */



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

int main() {
    char str[100];
    char words[50][50];
    int frequency[50] = {0};
    int wordCount = 0;

    printf("Enter a string of size less than 100: ");
    fgets(str, sizeof(str), stdin);

    char *token = strtok(str, " .,?\n");
    while (token != NULL) {

        for (int i = 0; token[i] != '\0'; i++) {
            token[i] = tolower(token[i]);
        }
```

```
        int found = 0;
        for (int i = 0; i < wordCount; i++) {
            if (strcmp(words[i], token) == 0) {
                frequency[i]++;
                found = 1;
                break;
            }
        }

        if (!found) {
            strcpy(words[wordCount], token);
            frequency[wordCount] = 1;
            wordCount++;
        }

        token = strtok(NULL, " .,?\n");
    }

    for (int i = 0; i < wordCount; i++) {
        printf("Word: %s, Frequency: %d\n", words[i], frequency[i]);
    }

    return 0;
}
```

```
PS C:\Users\betti\Desktop\Training\Day11> ./task2_v1
Enter a string of size less than 100: This is a test. This test is simple.
Word: this, Frequency: 2
Word: is, Frequency: 2
Word: a, Frequency: 1
Word: test, Frequency: 2
Word: simple, Frequency: 1
```

3.

```
/*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>

int main() {
    char text[100];
    char target[10];
    char replace[10];
    char result[200];

    printf("String: ");
    fgets(text, sizeof(text), stdin);
    text[strcspn(text, "\n")] = '\0';

    printf("Target: ");
    scanf("%s", target);
    printf("Replace: ");
    scanf("%s", replace);

    char *pFound = NULL, *currentPos = NULL;
    currentPos = text;
    result[0] = '\0';

    while((pFound = strstr(currentPos, target)) != NULL) {

        strncat(result, currentPos, pFound - currentPos);

        strcat(result, replace);

        currentPos = pFound + strlen(target);
    }

    strcat(result, currentPos);

    printf("Output: %s", result);
    return 0;
}
```

```
PS C:\Users\betti\Desktop\Training\Day11> ./task3
String: hello world, hello everyone
Target: hello
Replace: hi
Output: hi world, hi everyone
```

4.

```c
/*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>
#include <string.h>

int main() {
    char str[100], reverse_str[100];
    char *tokens[50];
    int token_count = 0;

    printf("Input: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    reverse_str[0] = '\0';

    char *token = strtok(str, " ");
    while(token!=NULL) {
        tokens[token_count++] = token;
        token = strtok(NULL, " ");
    }

    for (int i = token_count - 1; i >= 0; i--) {
        strcat(reverse_str, tokens[i]);
        if (i > 0) {
            strcat(reverse_str, " ");
        }
    }
```

```
    printf("Output: %s", reverse_str);
    return 0;
}
```

```
PS C:\Users\betti\Desktop\Training\Day11> ./task4
Input: The quick brown fox
Output: fox brown quick The
```

5.

```
/*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"*/

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

int main() {
    char str[100];
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);

    str[strcspn(str, "\n")] = '\0';

    int maxLen = 0;
    char longestSubstr[100] = "";

    int len = strlen(str);

    for (int start = 0; start < len; start++) {
        for (int end = start + 1; end <= len; end++) {
            char substr[100];
            strncpy(substr, &str[start], end - start);
            substr[end - start] = '\0';

            int count = 0;
            for (int i = 0; i <= len - (end - start); i++) {
```

```
                if (strncmp(&str[i], substr, end - start) == 0) {
                    count++;
                }
            }

            if (count > 1 && (end - start) > maxLen) {
                maxLen = end - start;
                strcpy(longestSubstr, substr);
            }
        }
    }

    if (maxLen > 0) {
        printf("Longest repeating substring: \"%s\"\n", longestSubstr);
    } else {
        printf("No repeating substring found.\n");
    }

    return 0;
}
```

```
PS C:\Users\betti\Desktop\Training\Day11> ./task5
Enter a string: banana
Longest repeating substring: "ana"
```