

Palindromic String Analyzer

Subject Name: C Programming

Submitted By: Avishek Kumar Jain

ERP id:Ru-25-10048

Semester / Section: 1st/A

College Name: Rungta International Skills
University



Introduction



String manipulation is an important concept in C programming.



A palindrome is a string that reads the same forward and backward.



This project analyze strings by:



Reversing them



Checking whether they are palindromes

Problem Statement

To design a menu-driven C program
that:

Accepts a string from the user

Reverses the string

**Checks whether the string is a
palindrome**





Objectives

- To understand string handling in C
- To implement user-defined functions
- To apply conditional statements and loops
- To create a menu-driven program

Tools & Technologies Used

Programming Language: C

Compiler: **GCC Compiler**

Header Files Used:

stdio.h

string.h

A hand with a ring on the middle finger is pointing its index finger towards a computer screen. The screen displays a block of C programming code. The code includes logic for mirroring objects based on user input ('MIRROR_X', 'MIRROR_Y', 'MIRROR_Z') and handles selection states for objects. It also includes a section for 'OPERATOR CLASSES' and a warning message about selecting exactly one object. The background of the slide features a blue-to-white gradient with a white curved line on the right side.

```
mirror_mod = modifier_obj
# Set mirror object to mirror
mirror_mod.mirror_object = selected_object
if operation == "MIRROR_X":
    mirror_mod.use_x = True
    mirror_mod.use_y = False
    mirror_mod.use_z = False
elif operation == "MIRROR_Y":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
elif operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

#selection at the end -add
#    ob.select= 1
#    user_ob.select=1
context.scene.objects.active = eval("Selected" + str(modifier))
mirror_ob.select = 0
bpy.context.selected_objects.append(mirror_ob)
data.objects[one.name].select = 1
print("please select exactly one object")
----- OPERATOR CLASSES -----
types.Operator:
    X mirror to the selected object.mirror_mirror_x"
    mirror X"
context:
    context.active_object is not None
```

Functions Used

reverseString()

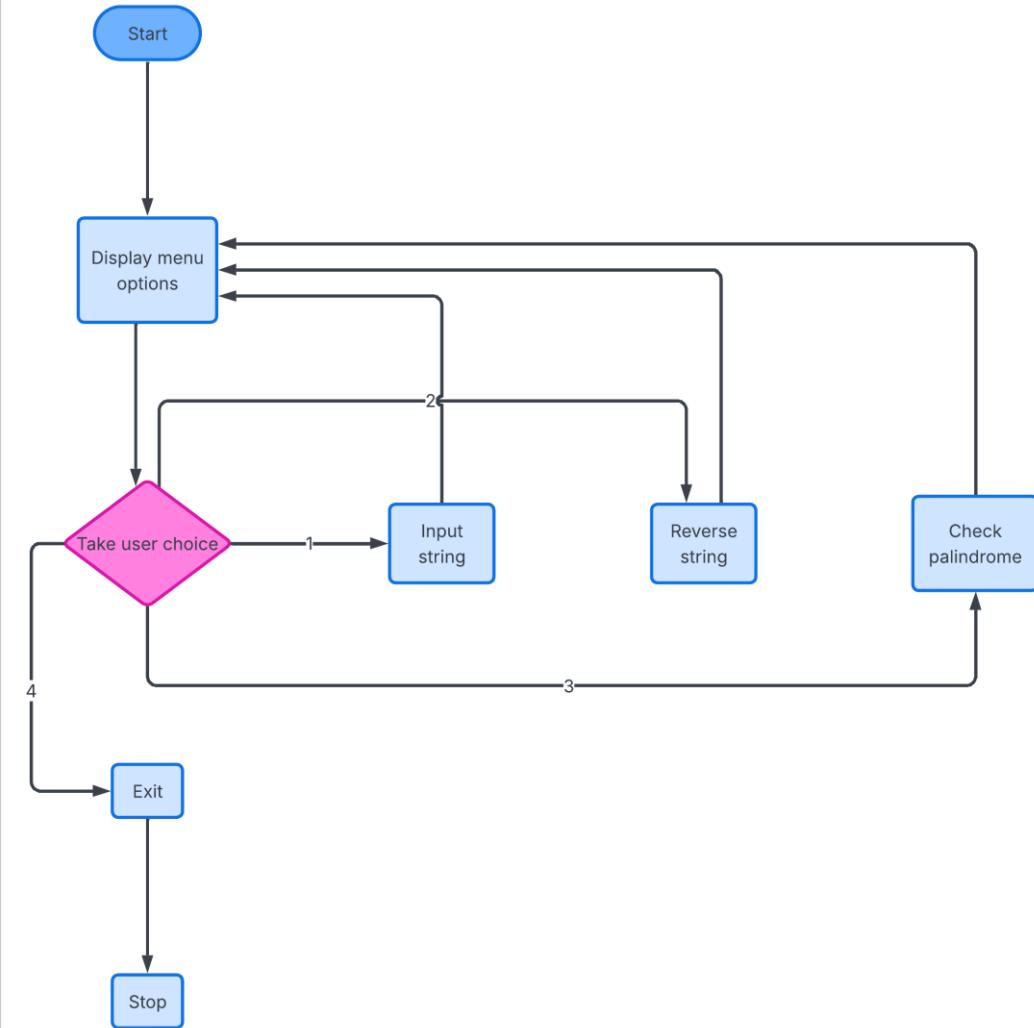
- Reverses the input string

isPalindrome()

- Checks if the string is palindrome

main()

- Controls menu and program flow



Algorithm

1. Start
2. Display menu options
3. Take user choice
4. If choice = 1 → Input string
5. If choice = 2 → Reverse string
6. If choice = 3 → Check palindrome
7. If choice = 4 → Exit
8. Stop

Program Working (Flow Explanation)

- User selects an option from the menu
- String is stored in an array
- Reverse function processes the string
- Palindrome function compares characters
- Result is displayed on screen



Source Code

C project.c X

```
C project.c > reverseString(char [], char [])
36         printf("Enter the string: ");
37         gets(str);
38         break;
39     case 2:
40         reverseString(str, rev);
41         printf("Reversed String: %s\n", rev);
42         break;
43     case 3:
44         if (isPalindrome(str))
45             printf("The string is a PALINDROME");
46         else
47             printf("The string is NOT a palindrome");
48         break;
49     case 4:
50         printf("Exiting program... \n");
51         break;
52     default:
53         printf("Invalid choice! Try again.\n");
54     }
55 } while (choice != 4);
56 return 0;
```

C project.c X

```
C project.c > reverseString(char [], char [])
1 #include <stdio.h>
2 #include <string.h>
3 void reverseString(char str[], char rev[]) {
4     int i, len;
5     len = strlen(str);
6     for (i = 0; i < len; i++) {
7         rev[i] = str[len - i - 1];
8     }
9     rev[len] = '\0';
10 }
11 int isPalindrome(char str[]) {
12     int i, len;
13     len = strlen(str);
14
15     for (i = 0; i < len / 2; i++) {
16         if (str[i] != str[len - i - 1]) {
17             return 0;
18         }
19     }
20     return 1;
21 }
22 int main() {
23     char str[100], rev[100];
24     int choice;
25     do {
26         printf("\n===== Palindromic String Analyzer =====\n");
27         printf("1. Enter a String\n");
28         printf("2. Reverse the String\n");
29         printf("3. Check Palindrome\n");
30         printf("4. Exit\n");
31         printf("Enter your choice: ");
```

Sample Output

Menu display

String input

Reversed
string output

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar listing various C files and a project folder named 'project.c'. The main editor area displays the following C code:

```
23     char str[100], rev[100];
24     int choice;
25     do {
26         printf("\n===== Palindromic String Analyzer =====\n");
27         printf("1. Enter a String\n");
28         printf("2. Reverse the String\n");
29         printf("3. Check Palindrome\n");
30         printf("4. Exit\n");
31         printf("Enter your choice: ");
32         scanf("%d", &choice);
33         getchar();
34         switch (choice) {
35             case 1:
36                 printf("Enter the string: ");
37                 gets(str);
38                 break;
39             case 2:

```

Below the code editor are several tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active, showing the command: "Executing task: C:/Windows/System32/cmd.exe /d /c .\build\Debug\outDebug.exe". The terminal output window shows the program's menu:

```
===== Palindromic String Analyzer =====
1. Enter a String
2. Reverse the String
3. Check Palindrome
4. Exit
Enter your choice:
```

Palindrome
check result

Applications



TEXT PROCESSING
SYSTEMS



DATA VALIDATION



LEARNING STRING
CONCEPTS



INTERVIEW AND
EXAM PRACTICE



BASIC COMPILER
PROJECTS

Advantages



SIMPLE AND USER-FRIENDLY



MODULAR USING FUNCTIONS



IMPROVES
UNDERSTANDING OF
STRINGS



MENU-DRIVEN APPROACH



Limitations

- Uses `gets()` (not secure)
- Limited string size
- Case-sensitive comparison
- Special characters not handled

Future Scope



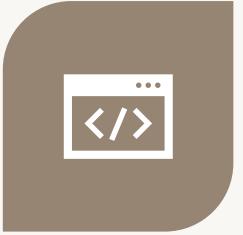
REPLACE GETS()
WITH FGETS()



CASE-INSENSITIVE
PALINDROME CHECK



SUPPORT FOR
SENTENCES



GUI-BASED VERSION

Conclusion

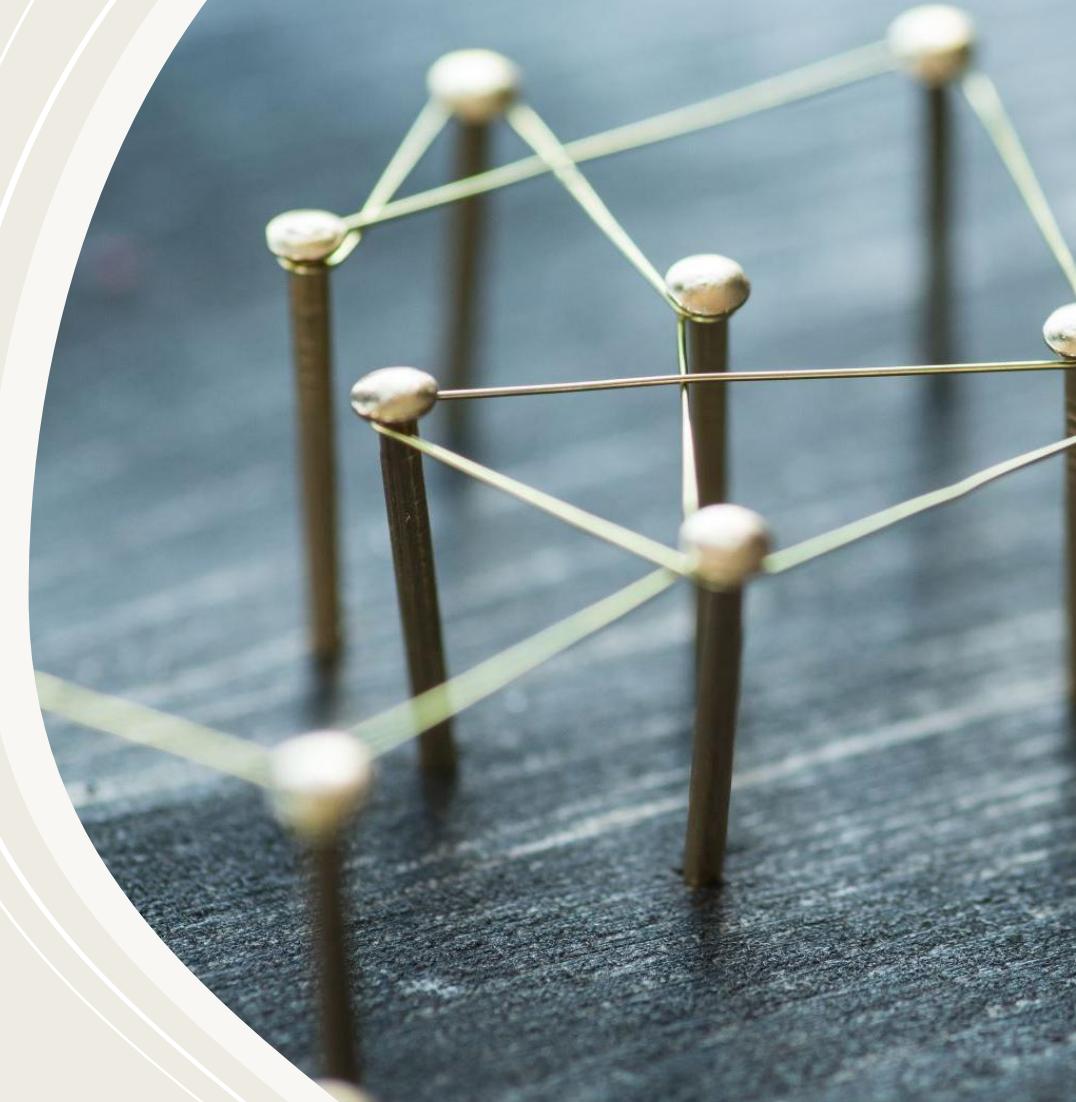
The project successfully demonstrates:

String manipulation

Function usage

Logical problem solving

It enhances understanding of core C concepts



Thank You



THANK YOU



ANY QUESTIONS?