**A Micro Project Report**

**on**

**Problem Solving using C Language**

Submitted by

#### VEMPARALA RAJESH (24475A0530)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET (AUTONOMOUS)**

**Accredited by NAAC with A+ Grade and NBA under Tier-1**

**NIRF rank in the band of 201-300 and is an ISO 9001:2015 certified Approved by AICTE, New Delhi, Permanently affiliated to JNTU Kakinada, Approved by AICTE, Accredited by NBA and accredited ’A+’ grade by NAAC Narasaraopet-522601, Palnadu(Dt.), Andhra Pradesh, India**

**2024-20****25**

**NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET**

**(AUTONOMOUS)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CERTIFICATE**

**This is to certify that VEMPARALA RAJESH rollno : 24475A0530, a Second Year Student of the Department of Computer Science and Engineering, has completed the Micro Project Satisfactorily in “Problem Solving using C Language" for the Academic Year 2024-2025.**.

Project Co-Ordinator HEAD OF THE DEPARTMENT

**Dr. Rama Krishna. Eluri, M.Tech., Ph.D.**  **Dr. S. N. Tirumala Rao,** **M.Tech., Ph.D. Asst. Professor Professor**

**INDEX**

|  |  |
| --- | --- |
| **S.No** | **Description** |
|  | C program to check string palindrome with out using string handling functions |
|  | Find shortest word from given sentence |
|  | String copy without strcpy() |
|  | compare string without strcpy() |
|  | Write a program that uses an array of pointers to strings str[]. Receive two strings str1 and str2 and check if str1 is embedded in any of the strings in str[]. If str1 is found, then replace it with str2.  Char\*str[]={“we will teach you how to…”,  “move a mountain”, “level a building”,  “Erase tha past”, “make a million”,  “…all through C!”};  For example if str1 contains “mountain” and str2 contains “car”, then the second string in str should get changed to “move a car”) |

**C programing codeing**

**AIM**: **C program to check string palindrome with out using string handling functions**

#include <stdio.h>

int main() {

char str[100], reverse[100];

int i = 0, j = 0, isPalindrome = 1;

printf("Enter a string: ");

fgets(str, sizeof(str), stdin);

while (str[i] != '\0') {

i++;

}

if (str[i - 1] == '\n') {

str[i - 1] = '\0';

}

i = 0;

while (str[i] != '\0') {

i++;

}

j = i - 1; // Set j to the last index of the string

for (int k = 0; k <= i / 2; k++) {

if (str[k] != str[j]) {

isPalindrome = 0;

break;

}

j--;

}

if (isPalindrome)

printf("The string is a palindrome.\n");

else

printf("The string is not a palindrome.\n");

return 0;

}

**Input 1: Palindrome string**

**Enter a string: madam**

**Output:**

**The string is a palindrome.**

**Input 2: Non-palindrome string**

**Enter a string: hello**

**Output:**

**The string is not a palindrome.**

**AIM 2:Find shortest word from given sentence**

#include <stdio.h>

#include <string.h>

#define MAX\_LEN 1000

int main() {

char sentence[MAX\_LEN], shortestWord[MAX\_LEN], tempWord[MAX\_LEN];

int minLength = MAX\_LEN;

printf("Enter a sentence: ");

fgets(sentence, sizeof(sentence), stdin);

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

int i = 0, j = 0;

while (1) {

if (sentence[i] != ' ' && sentence[i] != '\0') {

tempWord[j++] = sentence[i];

}

else {

if (j > 0) {

tempWord[j] = '\0';

if (strlen(tempWord) < minLength) {

minLength = strlen(tempWord);

strcpy(shortestWord, tempWord);

}

j = 0;

}

}

if (sentence[i] == '\0') {

break;

}

i++;

}

if (j > 0) {

tempWord[j] = '\0';

if (strlen(tempWord) < minLength) {

minLength = strlen(tempWord);

strcpy(shortestWord, tempWord);

}

}

printf("The shortest word is: %s\n", shortestWord);

return 0;

}

**input 1:**

**Simple sentence**

**Enter a sentence: Hello world this is a test**

**Output:**

**The shortest word is: a**

**Input 2: Sentence with multiple short words**

**Enter a sentence: I am a cat**

**Output:**

**The shortest word is: I**

**Aim 3:String copy without strcpy(**

#include <stdio.h>

int main() {

char source[100], destination[100];

int i = 0;

printf("Enter a string: ");

fgets(source, sizeof(source), stdin);

while (source[i] != '\0') {

destination[i] = source[i];

i++;

}

destination[i] = '\0';

printf("Copied string: %s\n", destination);

return 0;

}

**Input 1: Simple string**

**Enter a string:Hello World**

**Output:**

**Copied string: Hello World**

**Input 2: String with punctuation**

**Enter a string: Hello, World!**

**Output:**

**Copied string: Hello, World!**

**Aim 4:compare string without strcpy()**

#include <stdio.h>

int compareStrings(const char \*str1, const char \*str2) {

int i = 0;

while (str1[i] != '\0' && str2[i] != '\0') {

if (str1[i] != str2[i]) {

return 0;

}

i++;

}

if (str1[i] == '\0' && str2[i] == '\0') {

return 1;

}

return 0;

}

int main() {

char str1[100], str2[100];

printf("Enter the first string: ");

fgets(str1, sizeof(str1), stdin);

printf("Enter the second string: ");

fgets(str2, sizeof(str2), stdin);

str1[strcspn(str1, "\n")] = '\0';

str2[strcspn(str2, "\n")] = '\0';

if (compareStrings(str1, str2)) {

printf("The strings are equal.\n");

} else {

printf("The strings are not equal.\n");

}

return 0;

}

**Input 1:**

**Enter the first string: Hello**

**Enter the second string: Hello**

**Output:**

**The strings are equal.**

**Input 2:**

**Enter the first string: Hello**

**Enter the second string: World**

**Output:**

**Aim 5:Write a program that uses an array of pointers to strings str[]. Receive two strings str1 and str2 and check if str1 is embedded in any of the strings in str[]. If str1 is found, then replace it with str2.**

**Char\*str[]={“we will teach you how to…”,**

**“move a mountain”, “level a building”,**

**“Erase tha past”, “make a million”,**

**“…all through C!”};**

**For example if str1 contains “mountain” and str2 contains “car”, then the second string in str should get changed to “move a car”)**

#include <stdio.h>

#include <string.h>

#define MAX\_STR\_LEN 100

void replaceSubstring(char \*str, const char \*oldSubstr, const char \*newSubstr) {

char temp[MAX\_STR\_LEN];

char \*pos;

int oldLen = strlen(oldSubstr);

int newLen = strlen(newSubstr);

temp[0] = '\0';

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

strncat(temp, str, pos - str);

strcat(temp, newSubstr);

str = pos + oldLen;

}

strcat(temp, str);

strcpy(str, temp);

}

int main() {

char \*str[] = {

"we will teach you how to...",

"move a mountain",

"level a building",

"Erase the past",

"make a million",

"...all through C!"

};

int n = sizeof(str) / sizeof(str[0]);

char str1[MAX\_STR\_LEN], str2[MAX\_STR\_LEN];

printf("Enter the string to search for (str1): ");

fgets(str1, sizeof(str1), stdin);

str1[strcspn(str1, "\n")] = '\0';

printf("Enter the string to replace with (str2): ");

fgets(str2, sizeof(str2), stdin);

str2[strcspn(str2, "\n")] = '\0';

for (int i = 0; i < n; i++) {

if (strstr(str[i], str1) != NULL) { // If str1 is found in the current string

printf("Found '%s' in: \"%s\". Replacing it with '%s'.\n", str1, str[i], str2);

replaceSubstring(str[i], str1, str2);

}

}

printf("\nModified strings:\n");

for (int i = 0; i < n; i++) {

printf("%s\n", str[i]);

}

return 0;

}

**Input 1:**

**Enter the string to search for (str1): will**

**Enter the string to replace with (str2): can**

**Output:**

**Found 'will' in: "we will teach you how to...". Replacing it with 'can'.**

**Found 'will' in: "we will teach you how to...".**

**Input 2:**

**Enter the string to search for (str1): the**

**Enter the string to replace with (str2): a**

**Output:**

**Found 'the' in: "we will teach you how to...". Replacing it with 'a'.**

**Found 'the' in: "level a building".**

**Found 'the' in: "Erase the past".**

**The strings are not equal.**