Assignment 17

Strings

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int countVowels(char str[]) {

int count = 0;

for (int i = 0; str[i] != '\0'; i++) {

if (str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u' ||

str[i] == 'A' || str[i] == 'E' || str[i] == 'I' || str[i] == 'O' || str[i] == 'U') {

count++;

}

}

return count;

}

int isPalindrome(char str[]) {

int len = strlen(str);

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

if (str[i] != str[len - i - 1]) {

return 0;

}

}

return 1;

}

int isValidIP(char ip[]) {

char \*token = strtok(ip, ".");

int count = 0;

while (token != NULL) {

int num = atoi(token);

if (num < 0 || num > 255) {

return 0;

}

count++;

token = strtok(NULL, ".");

}

return (count == 4);

}

int minDistance(char \*s[], char word1[], char word2[], int n) {

int minDist = n;

int posWord1 = -1, posWord2 = -1;

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

if (strcmp(s[i], word1) == 0) {

posWord1 = i;

} else if (strcmp(s[i], word2) == 0) {

posWord2 = i;

}

if (posWord1 != -1 && posWord2 != -1) {

int dist = abs(posWord1 - posWord2);

if (dist < minDist) {

minDist = dist;}

}

}

return minDist;}

int factorial(int n) {

if (n == 0 || n == 1) {

return 1;

} else {

return n \* factorial(n - 1);

}

}

int authenticate(char username[], char password[]) {

char validUsername[] = "admin";

char validPassword[] = "admin123";

if (strcmp(username, validUsername) == 0 && strcmp(password, validPassword) == 0) {

return 1;

} else {

return 0;

}

}

int main() {

//1 find the number of vowels in each of the 5 strings stored in two dimensional arrays, taken from the user.

char strings[5][50];

for (int i = 0; i < 5; i++) {

printf("Enter string %d: ", i + 1);

gets(strings[i]);

}

printf("\nNumber of vowels in each string:\n");

for (int i = 0; i < 5; i++) {

printf("String %d: %d vowels\n", i + 1, countVowels(strings[i]));

}printf("\n");

//2 sort 10 city names stored in two dimensional arrays, taken from the user.

char cities[10][50];

for (int i = 0; i < 10; i++) {

printf("Enter city name %d: ", i + 1);

gets(cities[i]);

}

for (int i = 0; i < 9; i++) {

for (int j = i + 1; j < 10; j++) {

if (strcmp(cities[i], cities[j]) > 0) {

char temp[50];

strcpy(temp, cities[i]);

strcpy(cities[i], cities[j]);

strcpy(cities[j], temp);

}

}

}

printf("\nSorted city names:\n");

for (int i = 0; i < 10; i++) {

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

}printf("\n");

//3 read and display a 2D array of strings in C language.

printf("Enter 3 strings:\n");

for (int i = 0; i < 3; i++) {

printf("String %d: ", i + 1);

scanf("%s", strings[i]);

}

printf("\n2D Array of Strings:\n");

for (int i = 0; i < 3; i++) {

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

}printf("\n");

//4 search a string in the list of strings.

char search[50];

printf("Enter 5 strings:\n");

for (int i = 0; i < 5; i++) {

printf("String %d: ", i + 1);

gets(strings[i]);

}

printf("Enter string to search: ");

gets(search);

int found = 0;

for (int i = 0; i < 5; i++) {

if (strcmp(strings[i], search) == 0) {

found = 1;

break;

}

}

if (found) {

printf("String found in the list.\n\n");

} else {

printf("String not found in the list.\n\n");

}

//5 Suppose we have a list of email addresses, check whether all email addresses have '@'in it. Print the odd email out.

char emails[5][50];

printf("Enter 5 email addresses:\n");

for (int i = 0; i < 5; i++) {

printf("Email %d: ", i + 1);

gets(emails[i]);

if (strchr(emails[i], '@') == NULL) {

printf("Odd email out: %s\n", emails[i]);

}

}printf("\n");

//6 print the strings which are palindrome in the list of strings.

printf("Enter 5 strings:\n");

for (int i = 0; i < 5; i++) {

printf("String %d: ", i + 1);

gets(strings[i]);

}

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

for (int i = 0; i < 5; i++) {

if (isPalindrome(strings[i])) {

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

}

}printf("\n");

//7 From the list of IP addresses, check whether all ip addresses are valid.

char ips[3][20];

printf("Enter 3 IP addresses:\n");

for (int i = 0; i < 3; i++) {

printf("IP %d: ", i + 1);

gets(ips[i]);

if (!isValidIP(ips[i])) {

printf("Invalid IP address: %s\n", ips[i]);

}

}printf("\n");

//8 Given a list of words followed by two words, the task is to find the minimum distance between the given two words in the list of words.

char \*words[] = {"the", "quick", "brown", "fox", "quick"};

char word1[20], word2[20];

printf("Enter two words: ");

scanf("%s %s", word1, word2);

int distance = minDistance(words, word1, word2, 5);

printf("Minimum distance between '%s' and '%s' is: %d\n\n", word1, word2, distance);

//9 asks the user to enter a username. If the username entered is one of the names in the list then the user is allowed to calculate the factorial of a number.

//Otherwise, an error message is displayed

char usernames[][20] = {"user1", "user2", "user3"};

char inputUsername[20];

int n;

printf("Enter your username: ");

gets(inputUsername);

puts(inputUsername);

int validUser = 0;

for (int i = 0; i < 3; i++) {

if (strcmp(usernames[i], inputUsername) == 0) {

validUser = 1;

break;

}

}

if (validUser) {

printf("Enter a number to calculate its factorial: ");

scanf("%d", &n);

printf("Factorial of %d is: %d\n\n", n, factorial(n));

} else {

printf("Error: Invalid username.\n\n");

}

//10 Create an authentication system. It should be menu driven.

char username[20], password[20];

int choice;

while (choice != 2){

printf("\nMenu:\n");

printf("1. Login\n");

printf("2. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter username: ");

scanf("%s", username);

printf("Enter password: ");

scanf("%s", password);

if (authenticate(username, password)) {

printf("Authentication successful. Welcome, %s!\n", username);

} else {

printf("Authentication failed. Invalid username or password.\n");

}

break;

case 2:

printf("Exiting program.\n");

break;

default:

printf("Invalid choice. Please enter 1 or 2.\n");

}

}

return 0;

}