2347207

1 D integer array  
Write a C program using functions and pointers for the following  
1. Read and display n numbers  
2. Read and display odd positioned elements  
3. Display the even numbers from the set of integers  
4. Display maximum number from the set of integers  
5. Calculate the sum and average of n numbers

#include <stdio.h>

void read\_array(int \*arr, int n);

void display\_array(int \*arr, int n);

void display\_odd(int \*arr, int n);

void display\_even(int \*arr, int n);

int find\_max(int \*arr, int n);

void calculate\_sum\_avg(int \*arr, int n, int \*sum, float \*avg);

int main() {

    int n, sum, max;

    float avg;

    printf("Enter the number of elements in the array: ");

    scanf("%d", &n);

    int arr[n];

    read\_array(arr, n);

    printf("The array is: ");

    display\_array(arr, n);

    printf("The odd positioned elements are: ");

    display\_odd(arr, n);

    printf("The even numbers in the array are: ");

    display\_even(arr, n);

    max = find\_max(arr, n);

    printf("The greatest number in the array is: %d\n", max);

    calculate\_sum\_avg(arr, n, &sum, &avg);

    printf("The sum of the numbers in the array is: %d\n", sum);

    printf("The average of the numbers in the array is: %.2f\n", avg);

    return 0;

}

void read\_array(int \*arr, int n) {

    int i;

    printf("Enter %d numbers: ", n);

    for (i = 0; i < n; i++) {

        scanf("%d", &arr[i]);

    }

}

void display\_array(int \*arr, int n) {

    int i;

    for (i = 0; i < n; i++) {

        printf("%d ", arr[i]);

    }

    printf("\n");

}

void display\_odd(int \*arr, int n) {

    int i;

    for (i = 1; i < n; i += 2) {

        printf("%d ", arr[i]);

    }

    printf("\n");

}

void display\_even(int \*arr, int n) {

    int i;

    for (i = 0; i < n; i++) {

        if (arr[i] % 2 == 0) {

            printf("%d ", arr[i]);

        }

    }

    printf("\n");

}

int find\_max(int \*arr, int n) {

    int i;

    int max = arr[0];

    for (i = 1; i < n; i++) {

        if (arr[i] > max) {

            max = arr[i];

        }

    }

    return max;

}

void calculate\_sum\_avg(int \*arr, int n, int \*sum, float \*avg) {

    \*sum = 0;

    int i;

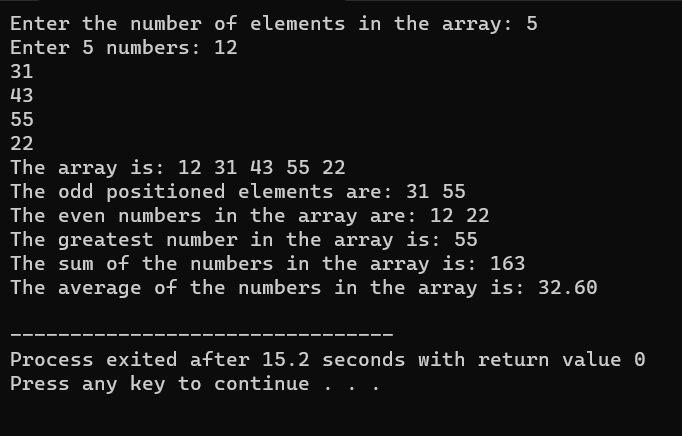
    for ( i = 0; i < n; i++) {

        \*sum += arr[i];

    }

    \*avg = (float)\*sum / n;

}



2 D integer array  
Write a C program using functions and pointers for the following  
1. Read and display n\*n matrix  
2. Calculate the row total of a given matrix  
3. Check whether the given matrix is identity matrix or not

#include <stdio.h>

#include <stdbool.h>

// Function to read n x n matrix

void readMatrix(int \*matrix, int n) {

    printf("Enter the elements of the %dx%d matrix:\n", n, n);

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++) {

            scanf("%d", &matrix[i \* n + j]);

        }

    }

}

// Function to display n x n matrix

void displayMatrix(int \*matrix, int n) {

    printf("Matrix:\n");

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++) {

            printf("%d\t", matrix[i \* n + j]);

        }

        printf("\n");

    }

}

// Function to calculate the row total of a given matrix

void calculateRowTotal(int \*matrix, int n, int rowIndex, int \*total) {

    \*total = 0;

    for (int j = 0; j < n; j++) {

        \*total += matrix[rowIndex \* n + j];

    }

}

// Function to check if the matrix is an identity matrix

bool isIdentityMatrix(int \*matrix, int n) {

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++) {

            if (i == j && matrix[i \* n + j] != 1) {

                return false;

            } else if (i != j && matrix[i \* n + j] != 0) {

                return false;

            }

        }

    }

    return true;

}

int main() {

    int n;

    printf("Enter the size of the square matrix: ");

    scanf("%d", &n);

    int matrix[n][n];

    readMatrix(&matrix[0][0], n);

    displayMatrix(&matrix[0][0], n);

    int rowIndex;

    printf("Enter the row index to calculate total: ");

    scanf("%d", &rowIndex);

    int total;

    calculateRowTotal(&matrix[0][0], n, rowIndex, &total);

    printf("Total of row %d: %d\n", rowIndex, total);

    if (isIdentityMatrix(&matrix[0][0], n)) {

        printf("The given matrix is an identity matrix.\n");

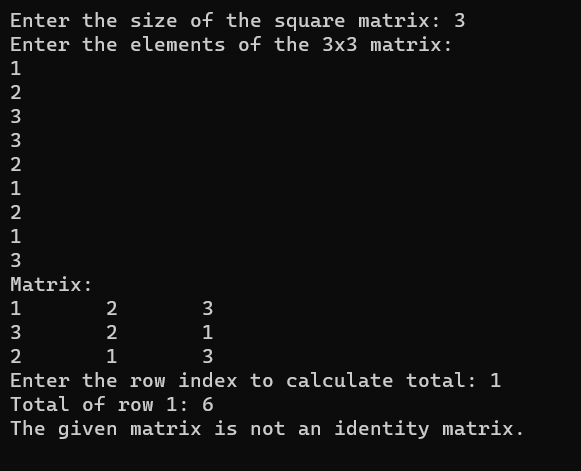
    } else {

        printf("The given matrix is not an identity matrix.\n");

    }

    return 0;

}



1 D Char array  
Write a C program using functions and pointers for the following  
1. Read and display a string  
2. Without using string builtin functions, calculate the string length  
3. Without using string builtin functions, reverse the string  
4. Without using string builtin functions, copy one string into other  
5. Read a string and check whether the given character is present or not. If present, count the number of times, it is repeated

#include <stdio.h>

#include <stdbool.h>

void readString(char \*str) {

printf("Enter a string: ");

scanf(" %s", str); .

int stringLength(char \*str) {

}

int length = 0;

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

length++;

}

return length;

}

void reverseString(char \*str) {

    int i,j;

int length = stringLength(str);

for ( i = 0, j = length - 1; i < j; i++, j--) {

char temp = str[i];

str[i] = str[j];

str[j] = temp;

}

}

void copyString(char \*src, char \*dest) {

int i = 0;

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

dest[i] = src[i];

i++;

}

dest[i] = '\0';

}

int countCharOccurrences(char \*str, char targetChar) {

int count = 0;

int i;

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

if (str[i] == targetChar) {

count++;

}

}

return count;

}

int main() {

char inputString[100];

readString(inputString);

printf("Entered String: %s\n", inputString);

int length = stringLength(inputString);

printf("String Length: %d\n", length);

reverseString(inputString);

printf("Reversed String: %s\n", inputString);

char copiedString[100];

copyString(inputString, copiedString);

printf("Copied String: %s\n", copiedString);

char targetChar;

printf("Enter a character to check its presence: ");

scanf(" %c", &targetChar); .

int charCount = countCharOccurrences(inputString, targetChar);

if (charCount > 0) {

printf("%c is present in the string, and it appears %d times.\n", targetChar, charCount);

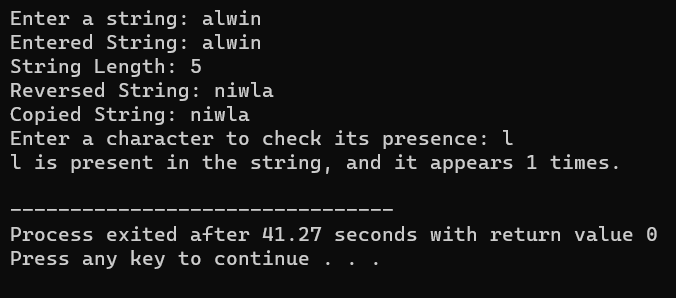
} else {

printf("%c is not present in the string.\n", targetChar);

}

return 0;

}



2 D Char array  
Write a C program using functions and pointers for the following  
1. Read and display n names  
2. Implement bubble sort for n names

#include <stdio.h>

#include <string.h>

#define MAX\_NAMES 100

#define MAX\_NAME\_LENGTH 50

void readNames(char names[][MAX\_NAME\_LENGTH], int n) {

    int i;

    printf("Enter %d names:\n", n);

    for ( i = 0; i < n; i++) {

        scanf("%s", names[i]);

    }

}

void displayNames(char names[][MAX\_NAME\_LENGTH], int n) {

    int i;

    printf("Names:\n");

    for ( i = 0; i < n; i++) {

        printf("%s\n", names[i]);

    }

}

void bubbleSortNames(char names[][MAX\_NAME\_LENGTH], int n) {

    int i,j;

    for ( i = 0; i < n - 1; i++) {

        for ( j = 0; j < n - i - 1; j++) {

            if (strcmp(names[j], names[j + 1]) > 0) {

                char temp[MAX\_NAME\_LENGTH];

                strcpy(temp, names[j]);

                strcpy(names[j], names[j + 1]);

                strcpy(names[j + 1], temp);

            }

        }

    }

}

int main() {

    int n;

    printf("Enter the number of names: ");

    scanf("%d", &n);

    if (n > MAX\_NAMES) {

        printf("Error: Exceeded the maximum number of names.\n");

        return 1;

    }

    char names[MAX\_NAMES][MAX\_NAME\_LENGTH];

    readNames(names, n);

    printf("Entered Names:\n");

    displayNames(names, n);

    bubbleSortNames(names, n);

    printf("Sorted Names:\n");

    displayNames(names, n);

    return 0;

}

