**Tutorial 1 :**   
  
Problem Statement : Write a C program to perform and display the results of all arithmetic operations

(+, -, \*, /, %) on two floating-point numbers provided by the user.

*#include* <stdio.h>

*// Function to calculate modulo for floating-point numbers*

float fmod\_custom(float *a*, float *b*) {

*return* *a* - (int)(*a* / *b*) \* *b*;

}

int main() {

float num1, num2;

float sum, difference, product, quotient, modulo;

*// Taking input from the user*

printf("Enter two floating-point numbers:\n");

printf("Number 1: ");

scanf("%f", &num1);

printf("Number 2: ");

scanf("%f", &num2);

*// Performing arithmetic operations*

sum = num1 + num2;

difference = num1 - num2;

product = num1 \* num2;

*// Check for division by zero*

*if* (num2 != 0) {

quotient = num1 / num2;

modulo = fmod\_custom(num1, num2);

} *else* {

printf("Error: Division or modulo by zero is not allowed.\n");

*return* 1; *// Exit the program if division by zero*

}

*// Displaying results*

printf("\nResults of Arithmetic Operations:\n");

printf("Sum: %.2f\n", sum);

printf("Difference: %.2f\n", difference);

printf("Product: %.2f\n", product);

printf("Quotient: %.2f\n", quotient);

printf("Modulo: %.2f\n", modulo);

*return* 0;

}

*/\* OUTPUT*

*Enter two floating-point numbers:*

*Number 1: 4*

*Number 2: 8*

*Results of Arithmetic Operations:*

*Sum: 12.00*

*Difference: -4.00*

*Product: 32.00*

*Quotient: 0.50*

*Modulo: 4.00*

*\*/*

**Tutorial 2 :**   
  
Problem Statement : Determine if a given year is a leap year or not using nested if-else statements.

*#include* <stdio.h>

int main() {

int year;

*// Input the year from the user*

printf("Enter a year: ");

scanf("%d", &year);

*// Check if the year is a leap year using nested if-else*

*if* (year % 4 == 0) {

*if* (year % 100 == 0) {

*if* (year % 400 == 0) {

printf("%d is a leap year.\n", year);

} *else* {

printf("%d is not a leap year.\n", year);

}

} *else* {

printf("%d is a leap year.\n", year);

}

} *else* {

printf("%d is not a leap year.\n", year);

}

*return* 0;

}

*/\* OUTPUT*

*Enter a year: 2019*

*2019 is not a*

*Enter a year: 2016*

*2016 is a le*

*\*/*

**Tutorial 3 :**   
  
Problem Statement : Print the first n prime numbers using a while loop and for loop.

*#include* <stdio.h>

int main() {

int n, count = 0, num = 2;

*// Input: Number of prime numbers to print*

printf("Enter the number of prime numbers to print: ");

scanf("%d", &n);

printf("The first %d prime numbers are:\n", n);

*// Use a while loop to find and print the first n prime numbers*

*while* (count < n) {

int isPrime = 1; *// Flag to indicate if a number is prime*

*// Check if the current number is prime*

*for* (int i = 2; i <= num / 2; i++) {

*if* (num % i == 0) {

isPrime = 0; *// Not a prime number*

*break*;

}

}

*// If the number is prime, print it and increase the count*

*if* (isPrime) {

printf("%d ", num);

count++;

}

num++; *// Check the next number*

}

printf("\n");

*return* 0;

}

}

*/\* OUTPUT*

*Enter the number of prime numbers to print: 5*

*The first 5 prime numbers are:*

*2 3 5*

*\*/*

**Tutorial 4**   
  
Problem Statement :Calculate the GCD and LCM of two numbers using both iterative and recursive

Functions.

*#include* <stdio.h>

*// Function prototypes*

int gcd\_iterative(int *a*, int *b*);

int gcd\_recursive(int *a*, int *b*);

int lcm(int *a*, int *b*);

int main() {

int num1, num2, gcd\_i, gcd\_r, lcm\_result;

*// Input two numbers from the user*

printf("Enter two numbers: ");

scanf("%d %d", &num1, &num2);

*// Compute GCD using iterative and recursive functions*

gcd\_i = gcd\_iterative(num1, num2);

gcd\_r = gcd\_recursive(num1, num2);

*// Compute LCM*

lcm\_result = lcm(num1, num2);

*// Display results*

printf("\nResults:\n");

printf("GCD (Iterative): %d\n", gcd\_i);

printf("GCD (Recursive): %d\n", gcd\_r);

printf("LCM: %d\n", lcm\_result);

*return* 0;

}

*// Iterative function to calculate GCD*

int gcd\_iterative(int *a*, int *b*) {

*while* (*b* != 0) {

int temp = *b*;

*b* = *a* % *b*;

*a* = temp;

}

*return* *a*;

}

*// Recursive function to calculate GCD*

int gcd\_recursive(int *a*, int *b*) {

*if* (*b* == 0)

*return* *a*;

*return* gcd\_recursive(*b*, *a* % *b*);

}

*// Function to calculate LCM*

int lcm(int *a*, int *b*) {

*return* (*a* \* *b*) / gcd\_iterative(*a*, *b*); *// Use the iterative GCD function*

}

*/\* OUTPUT*

*Enter two numbers: 20 50*

*Results:*

*GCD (Iterative): 10*

*GCD (Recursive): 10*

*LCM: 100*

*\*/*

**Tutorial 5**  
  
Problem Statement : Dynamically allocate memory for an array of integers, accept values from the user,

and find the sum and average of the elements.

*#include* <stdio.h>

*#include* <stdlib.h> *// For malloc and free*

int main() {

int n, \*arr;

int sum = 0;

float average;

*// Input the number of elements*

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

scanf("%d", &n);

*// Dynamically allocate memory for the array*

arr = (int \*)malloc(n \* sizeof(int));

*if* (arr == NULL) {

printf("Memory allocation failed.\n");

*return* 1; *// Exit the program if memory allocation fails*

}

*// Accept values from the user*

printf("Enter %d integers:\n", n);

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

scanf("%d", &arr[i]);

sum += arr[i]; *// Add to the sum*

}

*// Calculate the average*

average = (float)sum / n;

*// Display the sum and average*

printf("\nSum of the elements: %d\n", sum);

printf("Average of the elements: %.2f\n", average);

*// Free the dynamically allocated memory*

free(arr);

*return* 0;

}

*/\* OUTPUT*

*Enter the number of elements: 4*

*Enter 4 integers:*

*2 4 6 8*

*Sum of the elements: 20*

*Average of the ele*

*\*/*

**Tutorial 6 :**   
  
Problem Statement : In array do the following:

1. Find given element in array

2. Find Max element

3. Find Min element

4. Find frequency of given element in array

Find Average of elements in Array.

*#include* <stdio.h>

int main() {

int n, arr[100]; *// Static array with a maximum size of 100*

int element, frequency = 0;

int max, min, sum = 0;

float average;

*// Input the number of elements (up to 100)*

printf("Enter the number of elements in the array (max 100): ");

scanf("%d", &n);

*if* (n > 100) {

printf("Error: Number of elements exceeds the limit of 100.\n");

*return* 1; *// Exit if the input exceeds array size*

}

*// Accept array elements from the user*

printf("Enter %d elements:\n", n);

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

scanf("%d", &arr[i]);

}

*// Find max, min, sum, and average*

max = arr[0];

min = arr[0];

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

*if* (arr[i] > max) {

max = arr[i];

}

*if* (arr[i] < min) {

min = arr[i];

}

sum += arr[i];

}

average = (float)sum / n;

*// Find frequency of a given element*

printf("Enter the element to find in the array: ");

scanf("%d", &element);

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

*if* (arr[i] == element) {

frequency++;

}

}

*// Display results*

printf("\nResults:\n");

printf("Maximum element: %d\n", max);

printf("Minimum element: %d\n", min);

printf("Average of elements: %.2f\n", average);

*if* (frequency > 0) {

printf("The element %d is found in the array.\n", element);

} *else* {

printf("The element %d is not found in the array.\n", element);

}

printf("Frequency of %d: %d\n", element, frequency);

*return* 0;

}

*/\* OUTPUT*

*Enter the number of elements in the array (max 100): 4*

*Enter 4 elements:*

*1 3 5 7*

*Enter the element to find in the array: 3*

*Results:*

*Maximum element: 7*

*Minimum element: 1*

*Average of elements: 4.00*

*The element 3 is found in the array.*

*Frequency*

*\*/*

**Tutorial 7 :**

Problem Statement : Write a C program that takes a string as input and performs the following

operations:Reverse the string.

i) Check if the string is a palindrome.

ii) Count the frequency of each character in the string and display the results.

*#include* <stdio.h>

*#include* <string.h>

*#include* <ctype.h>

void reverseString(char *str*[], char *reversed*[]) {

int len = strlen(str);

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

reversed[i] = str[len - i - 1];

}

reversed[len] = '\0';

}

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; *// Not a palindrome*

}

}

*return* 1; *// Is a palindrome*

}

void countFrequency(char *str*[]) {

int freq[256] = {0}; *// ASCII characters array to store frequency*

*// Count the frequency of each character*

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

freq[(unsigned char)str[i]]++;

}

*// Display the frequencies*

printf("\nCharacter Frequencies:\n");

*for* (int i = 0; i < 256; i++) {

*if* (freq[i] > 0) {

printf("'%c': %d\n", i, freq[i]);

}

}

}

int main() {

char str[100], reversed[100];

*// Input the string*

printf("Enter a string: ");

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

str[strcspn(str, "\n")] = '\0'; *// Remove the trailing newline character*

*// Reverse the string*

reverseString(str, reversed);

printf("\nReversed String: %s\n", reversed);

*// Check if the string is a palindrome*

*if* (isPalindrome(str)) {

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

} *else* {

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

}

*// Count and display character frequencies*

countFrequency(str);

*return* 0;

}

*/\* OUTPUT*

*Enter a string: THIS IS GAURAV*

*Reversed String: VARUAG SI SIHT*

*The string is not a palindrome.*

*Character Frequencies:*

*' ': 2*

*'A': 2*

*'G': 1*

*'H': 1*

*'I': 2*

*'R': 1*

*'S': 2*

*'T': 1*

*'U': 1*

*'V'*

*Enter a string: RACECAR*

*Reversed String: RACECAR*

*The string is a palindrome.*

*Character Frequencies:*

*'A': 2*

*'C': 2*

*'E': 1*

*'R': 2*

*\*/*

**Tutorial 8 :**

Problem Statement : Display information of multiple students (name, age, grade) using an array of

Structures

*#include* <stdio.h>

*#include* <string.h>

*// Define the structure for a student*

struct Student {

char name[50];

int age;

char grade[5];

};

int main() {

int n;

*// Ask the user for the number of students*

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

scanf("%d", &n);

*// Declare an array of structures*

struct Student students[n];

*// Input student details*

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

printf("\nEnter details for student %d:\n", i + 1);

printf("Name: ");

scanf(" %[^\n]", students[i].name); *// %[^\n] to take string with spaces*

printf("Age: ");

scanf("%d", &students[i].age);

printf("Grade: ");

scanf(" %s", students[i].grade);

}

*// Display student details*

printf("\n--- Student Information ---\n");

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

printf("\nStudent %d:\n", i + 1);

printf("Name: %s\n", students[i].name);

printf("Age: %d\n", students[i].age);

printf("Grade: %s\n", students[i].grade);

}

*return* 0;

}

Output :

Enter the number of students: 4

Enter details *for* student 1:

Name: Shreyash

Age: 18

Grade: A

Enter details *for* student 2:

Name: Gaurav

Age: 18

Grade: A

Enter details *for* student 3:

Name: Vinit

Age: 18

Grade: A

Enter details *for* student 4:

Name: Sarthak

Age: 18

Grade: A

--- Student Information ---

Student 1:

Name: Shreyash

Age: 18

Grade: A

Student 2:

Name: Gaurav

Age: 18

Grade: A

Student 3:

Name: Vinit

Age: 18

Grade: A

Student 4:

Name: Sarthak

Age: 18

Grade: A

**Tutorial 9 :**

Problem Statement : Write a C program to store and retrieve employee records (name, id, salary) from a binary file

*#include* <stdio.h>

*#include* <stdlib.h>

struct Employee {

char name[50];

int id;

float salary;

};

int main() {

FILE \*file;

struct Employee emp;

int n, i;

printf("Enter number of employees: ");

scanf("%d", &n);

file = fopen("employee\_records.dat", "wb");

*if*(file == NULL) {

printf("Error opening file.\n");

*return* 1;

}

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

printf("\nEnter name of employee %d: ", i + 1);

scanf("%s", emp.name);

printf("Enter id of employee %d: ", i + 1);

scanf("%d", &emp.id);

printf("Enter salary of employee %d: ", i + 1);

scanf("%f", &emp.salary);

fwrite(&emp, sizeof(struct Employee), 1, file);

}

fclose(file);

file = fopen("employee\_records.dat", "rb");

*if*(file == NULL) {

printf("Error opening file.\n");

*return* 1;

}

printf("\nEmployee Records:\n");

*while*(fread(&emp, sizeof(struct Employee), 1, file)) {

printf("\nName: %s\n", emp.name);

printf("ID: %d\n", emp.id);

printf("Salary: %.2f\n", emp.salary);

}

fclose(file);

*return* 0;

}

Output:

Enter number of employees: 2

Enter name of employee 1: Akash

Enter id of employee 1: 101

Enter salary of employee 1: 50000

Enter name of employee 2: Rakesh

Enter id of employee 2: 102

Enter salary of employee 2: 55000

Employee Records:

Name: Akash

ID: 101

Salary: 50000.00

Name: Rakesh

ID: 102

Salary: 55000.00

**Tutorial 10 :**   
  
Problem Statement :Create a program that uses a union to store an integer, a floating-point number, or a string andDisplay the stored values.

*#include* <stdio.h>

union Data {

int i;

float f;

char str[20];

};

int main() {

union Data data;

data.i = 10;

printf("Integer value: %d\n", data.i);

data.f = 3.14;

printf("Float value: %.2f\n", data.f);

snprintf(data.str, sizeof(data.str), "Hello");

printf("String value: %s\n", data.str);

*return* 0;

}

Output:

Integer value: 10

Float value: 3.14

String value: Hello