1. Write a program in C to show the simple structure of a function.

#include <stdio.h>

int add(int, int);

int main() {

printf("Enter two numbers: ");

int a, b; scanf("%d %d", &a, &b);

printf("%d + %d = %d\n", a, b, add(a, b));

return 0;

}

int add(int num1, int num2) {

return num1 + num2;

}

Input: Enter two numbers:

1

2

Expected Output:

1+1=2

1. Write a program in C to find the square of any number using the function.

#include <stdio.h>

long square(int n) {

return n \* n;

}

int main() {

printf("Enter a number: ");

int n; scanf("%d", &n);

printf("The Square of %d is: %ld\n", n,square(n));

return 0;

}

Test Data:

Input a number: 20

Expected Output :

The square of 20 is : 400.00

1. Write a program in C to swap two numbers using function.

#include <stdio.h>

void swap(int \*a, int \*b) {

\*a = \*a + \*b;

\*b = \*a - \*b;

\*a = \*a - \*b;

}

int main() {

printf("Enter two numbers: ");

int a, b; scanf("%d %d", &a, &b);

printf("Original values:\n");

printf("First number: %d\nSecond number: %d\n", a, b);

swap(&a, &b);

printf("After swapping values:\n");

printf("First number: %d\nSecond number: %d\n", a, b);

return 0;

}

Test Data :

Enter two numbers :

2

4

Expected Output :

Original values: n1 = 2, n2 = 4

After swapping values: n1 = 4, n2 = 2

1. Write a program in C to check a given number is even or odd using the function.

#include <stdio.h>

int isOdd(int n) {

return n % 2;

}

int main() {

printf("Enter a number: ");

int n; scanf("%d", &n);

if(isOdd(n)) printf("%d is Odd\n", n);

else printf("%d is Even\n", n);

return 0;

}

Test Data :

Enter a number : 5

Expected Output :

5 is Odd

1. Write a program in C to find the sum of the series 1!/1+2!/2+3!/3+4!/4+5!/5 using the function.

#include <stdio.h>

long factorial(int n) {

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

return 1;

return n \* factorial(n-1);

}

long sum(int n) {

if(n == 1)

return 1;

return factorial(n-1) + sum(n-1);

}

int main() {

printf("Enter number of terms: ");

int n; scanf("%d", &n);

printf("Sum of first %d terms of the series: %ld\n", n, sum(n));

return 0;

}

Input:

Enter number of terms: 5

Expected Output :

The sum of the series is : 34

1. Write a program in C to convert decimal number to binary number using the function.

|  |
| --- |
| #include <stdio.h> |
|  |  |
|  | long dec2bin(int dec) { |
|  | if(dec == 0) |
|  | return 0; |
|  | return dec % 2 + 10 \* dec2bin(dec / 2); |
|  | } |
|  |  |
|  | int main() { |
|  | printf("Enter a number in decimal form: "); |
|  | int n; scanf("%d", &n); |
|  | printf("Binary form of %d: %ld\n", n, dec2bin(n)); |
|  |  |
|  | return 0; |
|  | } |

Test Data :

Input a number in decimal form: 65

Expected Output :

Binary form of 65 is : 1000001

1. Write a program in C to check whether a number is a prime number or not using the function.

|  |
| --- |
| #include <stdio.h> |
|  | #include <stdbool.h> |
|  |  |
|  | bool isPrime(int n) { |
|  | for(int i = 2; i < n; i++) |
|  | if(n % i == 0) |
|  | return false; |
|  | return true; |
|  | } |
|  |  |
|  | int main() { |
|  | printf("Enter a number: "); |
|  | int n; scanf("%d", &n); |
|  | if(isPrime(n)) |
|  | printf("%d is a Prime number\n", n); |
|  | else |
|  | printf("%d is a Composite number\n", n); |
|  |  |
|  | return 0; |
|  | } |

Test Data :

Enter a number : 5

Expected Output :

5 is a prime number.

1. Write a program in C to get the largest element of an array using the function.

|  |
| --- |
| #include <stdio.h> |
|  |  |
|  | int max(int a, int b) { |
|  | return a > b ? a : b; |
|  | } |
|  |  |
|  | int findMax(int arr[], int n) { |
|  | if(n == 1) |
|  | return arr[0]; |
|  | return max(arr[n-1], findMax(arr, n-1)); |
|  | } |
|  |  |
|  | int main() { |
|  | printf("Enter the size of the array: "); |
|  | int n; scanf("%d", &n); int arr[n]; |
|  |  |
|  | printf("Input %d elements in the array:\n", n); |
|  | for(int i = 0; i < n; i++) { |
|  | printf("element - %d: "); |
|  | scanf("%d", arr+i); |
|  | } |
|  | printf("Largest element in the array: %d\n", findMax(arr, n)); |
|  |  |
|  | return 0; |
|  | } |

Test Data :

Enter the size of the array :5

Input 5 elements in the array :

element - 0 : 1

element - 1 : 2

element - 2 : 3

element - 3 : 4

element - 4 : 5

Expected Output :

The largest element in the array is : 5

1. Write a program in C to check armstrong and perfect numbers using the function.

|  |
| --- |
| #include <stdio.h> |
|  | #include <math.h> |
|  | #include <stdbool.h> |
|  |  |
|  | bool isArmstrong(int num) { |
|  | int digits = log10(num) + 1; |
|  | int sum = 0; |
|  | for(int temp = num; temp > 0; temp /= 10) |
|  | sum += (int)pow(temp % 10, digits); |
|  | return sum == num; |
|  | } |
|  |  |
|  | bool isPerfect(int num) { |
|  | int sum = 0; |
|  | for(int i = 1; i < num; i++) |
|  | if(num % i == 0) |
|  | sum += i; |
|  | return sum == num; |
|  | } |
|  |  |
|  | int main() { |
|  | printf("Enter a number: "); |
|  | int n; scanf("%d", &n); |
|  |  |
|  | if(isArmstrong(n)) |
|  | printf("%d is an armstrong number\n", n); |
|  | else |
|  | printf("%d is not an armstrong number\n", n); |
|  |  |
|  | if(isPerfect(n)) |
|  | printf("%d is a perfect number\n", n); |
|  | else |
|  | printf("%d is not a perfect number\n", n); |
|  |  |
|  | return 0; |
|  | } |

Test Data :

Enter a number: 371

Expected Output :

371 is an Armstrong number.

371 is not a Perfect number.

1. Write a program in C to print all perfect numbers in given range using the function.

|  |
| --- |
| #include <stdio.h> |
|  | #include <stdbool.h> |
|  |  |
|  | bool isPerfect(int num) { |
|  | int sum = 0; |
|  | for(int i = 1; i < num; i++) |
|  | if(num % i == 0) |
|  | sum += i; |
|  | return num == sum; |
|  | } |
|  |  |
|  | void printPerfectNumbers(int lower, int upper) {  printf(“The perfect numbers between 1 to 100 are :”); |
|  | while(lower <= upper) { |
|  | if(isPerfect(lower)) |
|  | printf("%d ", lower); |
|  | lower++; |
|  | } |
|  | printf("\n"); |
|  | } |
|  |  |
|  | int main() { |
|  | printf("Enter lower limit: "); |
|  | int l; scanf("%d", &l); |
|  | printf("Enter upper limit: "); |
|  | int u; scanf("%d", &u); |
|  | printPerfectNumbers(l, u); |
|  | } |

Test Data :

Enter lower limit: 1

Enter upper limit: 100

Expected Output :

The perfect numbers between 1 to 100 are :

6 28

1. Write a program in C to check whether two given strings are an anagram.

|  |
| --- |
| #include <stdio.h> |
|  | #include <string.h> |
|  |  |
|  | #define MAX\_SIZE 100 |
|  |  |
|  | void swap(char \*a, char \*b) { |
|  | char temp = \*a; |
|  | \*a = \*b; |
|  | \*b = temp; |
|  | } |
|  |  |
|  | void sort(char str[], int n) { |
|  | for(int i = 0; i < n; i++) |
|  | for(int j = 1; j < n-i; j++) |
|  | if(str[j-1] > str[j]) |
|  | swap(str+j-1, str+j); |
|  | } |
|  |  |
|  | int compare(char str1[], char str2[]) { |
|  | sort(str1, strlen(str1)); |
|  | sort(str2, strlen(str2)); |
|  | return strcmp(str1, str2); |
|  | } |
|  |  |
|  | int main() { |
|  | char str1[MAX\_SIZE], str2[MAX\_SIZE], temp1[MAX\_SIZE], temp2[MAX\_SIZE]; |
|  | printf("Input the first string: "); |
|  | scanf("%s", str1); |
|  | printf("Input the second string: "); |
|  | scanf("%s", str2); |
|  |  |
|  | strcpy(temp1, str1); strcpy(temp2, str2); |
|  |  |
|  | if(compare(str1, str2) == 0) |
|  | printf("%s and %s are anagrams\n", temp1, temp2); |
|  | else |
|  | printf("%s and %s are not anagrams\n", temp1, temp2); |
|  |  |
|  | return 0; |
|  | } |

Test Data :

Input the first String : spare

Input the second String : pears

Expected Output :

spare and pears are anagrams.

1. Write a C programming to find out maximum and minimum of some values using function which will return an array.

|  |
| --- |
| #include <stdio.h> |
|  | #include <stdlib.h> |
|  |  |
|  | int \*extremes(int arr[], int n) { |
|  | int \*ans = malloc(2 \* sizeof(int)); |
|  | ans[0] = arr[0], ans[1] = arr[0]; |
|  | for(int i = 1; i < n; i++) { |
|  | if(arr[i] > ans[0]) |
|  | ans[0] = arr[i]; |
|  | if(arr[i] < ans[1]) |
|  | ans[1] = arr[i]; |
|  | } |
|  |  |
|  | return ans; |
|  | } |
|  |  |
|  | int main() { |
|  | printf("How many numbers you want to compare? "); |
|  | int n; scanf("%d", &n); int arr[n]; |
|  |  |
|  | printf("Input %d values:\n", n); |
|  | for(int i = 0; i < n; i++) |
|  | scanf("%d", arr+i); |
|  | int \*ans = extremes(arr, n); |
|  | printf("Minimum value: %d\n", \*(ans+1)); |
|  | printf("Maximum value: %d\n", \*ans); |
|  |  |
|  | free(ans); |
|  |  |
|  | return 0; |
|  | } |

Test Data :

How many numbers you want to compare? 5

Input 5 values

2

11

35

65

20

Expected Output :

Minimum value: 11

Maximum value: 65