**Program to Display "Hello, World!"**

#include <stdio.h>

int main()

{

// printf() displays the string inside quotation

printf("Hello, World!");

return 0;

}

**Output**

Hello, World!

## Program to Print an Integer

#include <stdio.h>

int main()

{

int number;

// printf() dislpays the formatted output

printf("Enter an integer: ");

// scanf() reads the formatted input and stores them

scanf("%d", &number);

// printf() displays the formatted output

printf("You entered: %d", number);

return 0;

}

**Output**

Enter a integer: 25

You entered: 25

## Program to Add Two Integers

#include <stdio.h>

int main()

{

int firstNumber, secondNumber, sumOfTwoNumbers;

printf("Enter two integers: ");

// Two integers entered by user is stored using scanf() function

scanf("%d %d", &firstNumber, &secondNumber);

// sum of two numbers in stored in variable sumOfTwoNumbers

sumOfTwoNumbers = firstNumber + secondNumber;

// Displays sum

printf("%d + %d = %d", firstNumber, secondNumber, sumOfTwoNumbers);

return 0;

}

**Output**

Enter two integers: 12

11

12 + 11 = 23

## Program to Multiply Two Numbers

#include <stdio.h>

int main()

{

double firstNumber, secondNumber, product;

printf("Enter two numbers: ");

// Stores two floating point numbers in variable firstNumber and secondNumber respectively

scanf("%lf %lf", &firstNumber, &secondNumber);

// Performs multiplication and stores the result in variable productOfTwoNumbers

product = firstNumber \* secondNumber;

// Result up to 2 decimal point is displayed using %.2lf

printf("Product = %.2lf", product);

return 0;

}

**Output**

Enter two numbers: 2.4

1.12

Product = 2.69

## Program to Print ASCII Value

#include <stdio.h>

int main()

{

char c;

printf("Enter a character: ");

// Reads character input from the user

scanf("%c", &c);

// %d displays the integer value of a character

// %c displays the actual character

printf("ASCII value of %c = %d", c, c);

return 0;

}

**Output**

Enter a character: G

ASCII value of G = 71

## Program to Compute Quotient and Remainder

#include <stdio.h>

int main(){

int dividend, divisor, quotient, remainder;

printf("Enter dividend: ");

scanf("%d", &dividend);

printf("Enter divisor: ");

scanf("%d", &divisor);

// Computes quotient

quotient = dividend / divisor;

// Computes remainder

remainder = dividend % divisor;

printf("Quotient = %d\n", quotient);

printf("Remainder = %d", remainder);

return 0;

}

**Output**

Enter dividend: 25

Enter divisor: 4

Quotient = 6

Remainder = 1

## Example: Program to Find the Size of a variable

#include <stdio.h>

int main()

{

int integerType;

float floatType;

double doubleType;

char charType;

// Sizeof operator is used to evaluate the size of a variable

printf("Size of int: %ld bytes\n",sizeof(integerType));

printf("Size of float: %ld bytes\n",sizeof(floatType));

printf("Size of double: %ld bytes\n",sizeof(doubleType));

printf("Size of char: %ld byte\n",sizeof(charType));

return 0;

}

**Output**

Size of int: 4 bytes

Size of float: 4 bytes

Size of double: 8 bytes

Size of char: 1 byte

## Program to Demonstrate the Working of long

#include <stdio.h>

int main()

{

int a;

long b;

long long c;

double e;

long double f;

printf("Size of int = %ld bytes \n", sizeof(a));

printf("Size of long = %ld bytes\n", sizeof(b));

printf("Size of long long = %ld bytes\n", sizeof(c));

printf("Size of double = %ld bytes\n", sizeof(e));

printf("Size of long double = %ld bytes\n", sizeof(f));

return 0;

}

**Output**

Size of int = 4 bytes

Size of long = 8 bytes

Size of long long = 8 bytes

Size of double = 8 bytes

Size of long double = 16 bytes

## Program to Swap Numbers Using Temporary Variable

#include <stdio.h>

int main()

{

double firstNumber, secondNumber, temporaryVariable;

printf("Enter first number: ");

scanf("%lf", &firstNumber);

printf("Enter second number: ");

scanf("%lf",&secondNumber);

// Value of firstNumber is assigned to temporaryVariable

temporaryVariable = firstNumber;

// Value of secondNumber is assigned to firstNumber

firstNumber = secondNumber;

// Value of temporaryVariable (which contains the initial value of firstNumber) is assigned to secondNumber

secondNumber = temporaryVariable;

printf("\nAfter swapping, firstNumber = %.2lf\n", firstNumber);

printf("After swapping, secondNumber = %.2lf", secondNumber);

return 0;

}

**Output**

Enter first number: 1.20

Enter second number: 2.45

After swapping, firstNumber = 2.45

After swapping, secondNumber = 1.20

## Program to Swap Number Without Using Temporary Variables

#include <stdio.h>

int main()

{

double firstNumber, secondNumber;

printf("Enter first number: ");

scanf("%lf", &firstNumber);

printf("Enter second number: ");

scanf("%lf",&secondNumber);

// Swapping process

firstNumber = firstNumber - secondNumber;

secondNumber = firstNumber + secondNumber;

firstNumber = secondNumber - firstNumber;

printf("\nAfter swapping, firstNumber = %.2lf\n", firstNumber);

printf("After swapping, secondNumber = %.2lf", secondNumber);

return 0;

}

**Output**

Enter first number: 10.25

Enter second number: -12.5

After swapping, firstNumber = -12.50

After swapping, secondNumber = 10.25

## Program to Check Even or Odd

#include <stdio.h>

int main()

{

int number;

printf("Enter an integer: ");

scanf("%d", &number);

// True if the number is perfectly divisible by 2

if(number % 2 == 0)

printf("%d is even.", number);

else

printf("%d is odd.", number);

return 0;

}

**Output**

Enter an integer: -7

-7 is odd.

## Program to Check Odd or Even Using Conditional Operator

#include <stdio.h>

int main()

{

int number;

printf("Enter an integer: ");

scanf("%d", &number);

(number % 2 == 0) ? printf("%d is even.", number) : printf("%d is odd.", number);

return 0;

}

## Program to Check Vowel or consonant

#include <stdio.h>

int main()

{

char c;

int isLowercaseVowel, isUppercaseVowel;

printf("Enter an alphabet: ");

scanf("%c",&c);

// evaluates to 1 (true) if c is a lowercase vowel

isLowercaseVowel = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');

// evaluates to 1 (true) if c is an uppercase vowel

isUppercaseVowel = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U');

// evaluates to 1 (true) if either isLowercaseVowel or isUppercaseVowel is true

if (isLowercaseVowel || isUppercaseVowel)

printf("%c is a vowel.", c);

else

printf("%c is a consonant.", c);

return 0;

}

**Output**

Enter an alphabet: G

G is a consonant

## Program to Check Vowel or consonant

#include <stdio.h>

#include <ctype.h>

int main()

{

char c;

int isLowercaseVowel, isUppercaseVowel;

do {

printf("Enter an alphabet: ");

scanf(" %c", &c);

}

// isalpha() returns 0 if the passed character is not an alphabet

while (!isalpha(c));

// evaluates to 1 (true) if c is a lowercase vowel

isLowercaseVowel = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');

// evaluates to 1 (true) if c is an uppercase vowel

isUppercaseVowel = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U');

// evaluates to 1 (true) if either isLowercaseVowel or isUppercaseVowel is true

if (isLowercaseVowel || isUppercaseVowel)

printf("%c is a vowel.", c);

else

printf("%c is a consonant.", c);

return 0;

}

# **C Program to Find the Largest Number Among Three Numbers**

## Example #1

#include <stdio.h>

int main()

{

double n1, n2, n3;

printf("Enter three different numbers: ");

scanf("%lf %lf %lf", &n1, &n2, &n3);

if( n1>=n2 && n1>=n3 )

printf("%.2f is the largest number.", n1);

if( n2>=n1 && n2>=n3 )

printf("%.2f is the largest number.", n2);

if( n3>=n1 && n3>=n2 )

printf("%.2f is the largest number.", n3);

return 0;

}

This program uses if...else statement to find the largest number.

## Example #2

#include <stdio.h>

int main()

{

double n1, n2, n3;

printf("Enter three numbers: ");

scanf("%lf %lf %lf", &n1, &n2, &n3);

if (n1>=n2)

{

if(n1>=n3)

printf("%.2lf is the largest number.", n1);

else

printf("%.2lf is the largest number.", n3);

}

else

{

if(n2>=n3)

printf("%.2lf is the largest number.", n2);

else

printf("%.2lf is the largest number.",n3);

}

return 0;

}

This program uses nested if...else statement to find the largest number.

## Example #3

#include <stdio.h>

int main()

{

double n1, n2, n3;

printf("Enter three numbers: ");

scanf("%lf %lf %lf", &n1, &n2, &n3);

if( n1>=n2 && n1>=n3)

printf("%.2lf is the largest number.", n1);

else if (n2>=n1 && n2>=n3)

printf("%.2lf is the largest number.", n2);

else

printf("%.2lf is the largest number.", n3);

return 0;

}

Though, the largest number among three numbers is found using multiple ways, the output of all these program will be same.

Enter three numbers: -4.5

3.9

5.6

5.60 is the largest number.

## Program to Find Roots of a Quadratic Equation

#include <stdio.h>

#include <math.h>

int main()

{

double a, b, c, determinant, root1,root2, realPart, imaginaryPart;

printf("Enter coefficients a, b and c: ");

scanf("%lf %lf %lf",&a, &b, &c);

determinant = b\*b-4\*a\*c;

// condition for real and different roots

if (determinant > 0)

{

// sqrt() function returns square root

root1 = (-b+sqrt(determinant))/(2\*a);

root2 = (-b-sqrt(determinant))/(2\*a);

printf("root1 = %.2lf and root2 = %.2lf",root1 , root2);

}

//condition for real and equal roots

else if (determinant == 0)

{

root1 = root2 = -b/(2\*a);

printf("root1 = root2 = %.2lf;", root1);

}

// if roots are not real

else

{

realPart = -b/(2\*a);

imaginaryPart = sqrt(-determinant)/(2\*a);

printf("root1 = %.2lf+%.2lfi and root2 = %.2f-%.2fi", realPart, imaginaryPart, realPart, imaginaryPart);

}

return 0;

}

**Output**

Enter coefficients a, b and c: 2.3

4

5.6

Roots are: -0.87+1.30i and -0.87-1.30i

## Program to Check Leap Year

#include <stdio.h>

int main()

{

int year;

printf("Enter a year: ");

scanf("%d",&year);

if(year%4 == 0)

{

if( year%100 == 0)

{

// year is divisible by 400, hence the year is a leap year

if ( year%400 == 0)

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

else

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

}

else

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

}

else

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

return 0;

}

**Output 1**

Enter a year: 1900

1900 is not a leap year.

**Output 2**

Enter a year: 2012

2012 is a leap year.

## Check if a Number is Positive or Negative Using if...else

#include <stdio.h>

int main()

{

double number;

printf("Enter a number: ");

scanf("%lf", &number);

if (number <= 0.0)

{

if (number == 0.0)

printf("You entered 0.");

else

printf("You entered a negative number.");

}

else

printf("You entered a positive number.");

return 0;

}

## Check if a Number is Positive or Negative Using Nested if...else

#include <stdio.h>

int main()

{

double number;

printf("Enter a number: ");

scanf("%lf", &number);

// true if number is less than 0

if (number < 0.0)

printf("You entered a negative number.");

// true if number is greater than 0

else if ( number > 0.0)

printf("You entered a positive number.");

// if both test expression is evaluated to false

else

printf("You entered 0.");

return 0;

}

**Output 1**

Enter a number: 12.3

You entered a positive number.

**Output 2**

Enter a number: 0

You entered 0.

## Program to Check Alphabet

#include <stdio.h>

int main()

{

char c;

printf("Enter a character: ");

scanf("%c",&c);

if( (c>='a' && c<='z') || (c>='A' && c<='Z'))

printf("%c is an alphabet.",c);

else

printf("%c is not an alphabet.",c);

return 0;

}

**Output**

Enter a character: \*

\* is not an alphabet

## Sum of Natural Numbers Using for Loop

#include <stdio.h>

int main()

{

int n, i, sum = 0;

printf("Enter a positive integer: ");

scanf("%d",&n);

for(i=1; i <= n; ++i)

{

sum += i; // sum = sum+i;

}

printf("Sum = %d",sum);

return 0;

}

## Sum of Natural Numbers Using while Loop

#include <stdio.h>

int main()

{

int n, i, sum = 0;

printf("Enter a positive integer: ");

scanf("%d",&n);

i = 1;

while ( i <=n )

{

sum += i;

++i;

}

printf("Sum = %d",sum);

return 0;

}

**Output**

Enter a positive integer: 100

Sum = 5050

## Program to Read Input Until User Enters a Positive Integer

#include <stdio.h>

int main()

{

int n, i, sum = 0;

do {

printf("Enter a positive integer: ");

scanf("%d",&n);

}

while (n <= 0);

for(i=1; i <= n; ++i)

{

sum += i; // sum = sum+i;

}

printf("Sum = %d",sum);

return 0;

}

## Factorial of a Number

#include <stdio.h>

int main()

{

int n, i;

unsigned long long factorial = 1;

printf("Enter an integer: ");

scanf("%d",&n);

// show error if the user enters a negative integer

if (n < 0)

printf("Error! Factorial of a negative number doesn't exist.");

else

{

for(i=1; i<=n; ++i)

{

factorial \*= i; // factorial = factorial\*i;

}

printf("Factorial of %d = %llu", n, factorial);

}

return 0;

}

**Output**

Enter an integer: 10

Factorial of 10 = 3628800

## Multiplication Table Up to 10

#include <stdio.h>

int main()

{

int n, i;

printf("Enter an integer: ");

scanf("%d",&n);

for(i=1; i<=10; ++i)

{

printf("%d \* %d = %d \n", n, i, n\*i);

}

return 0;

}

**Output**

Enter an integer: 9

9 \* 1 = 9

9 \* 2 = 18

9 \* 3 = 27

9 \* 4 = 36

9 \* 5 = 45

9 \* 6 = 54

9 \* 7 = 63

9 \* 8 = 72

9 \* 9 = 81

9 \* 10 = 90

## Multiplication Table Up to a range (entered by the user)

#include <stdio.h>

int main()

{

int n, i, range;

printf("Enter an integer: ");

scanf("%d",&n);

printf("Enter the range: ");

scanf("%d", &range);

for(i=1; i <= range; ++i)

{

printf("%d \* %d = %d \n", n, i, n\*i);

}

return 0;

}

**Output**

Enter an integer: 12

Enter the range: 8

12 \* 1 = 12

12 \* 2 = 24

12 \* 3 = 36

12 \* 4 = 48

12 \* 5 = 60

12 \* 6 = 72

12 \* 7 = 84

12 \* 8 = 96

## Fibonacci Series up to n number of terms

#include <stdio.h>

int main()

{

int i, n, t1 = 0, t2 = 1, nextTerm;

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

scanf("%d", &n);

printf("Fibonacci Series: ");

for (i = 1; i <= n; ++i)

{

printf("%d, ", t1);

nextTerm = t1 + t2;

t1 = t2;

t2 = nextTerm;

}

return 0;

}

**Output**

Enter the number of terms: 10

Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34,

## Fibonacci Sequence Up to a Certain Number

#include <stdio.h>

int main()

{

int t1 = 0, t2 = 1, nextTerm = 0, n;

printf("Enter a positive number: ");

scanf("%d", &n);

// displays the first two terms which is always 0 and 1

printf("Fibonacci Series: %d, %d, ", t1, t2);

nextTerm = t1 + t2;

while(nextTerm <= n)

{

printf("%d, ",nextTerm);

t1 = t2;

t2 = nextTerm;

nextTerm = t1 + t2;

}

return 0;

}

**Output**

Enter a positive integer: 100

Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89,

## GCD Using for loop and if Statement

#include <stdio.h>

int main()

{

int n1, n2, i, gcd;

printf("Enter two integers: ");

scanf("%d %d", &n1, &n2);

for(i=1; i <= n1 && i <= n2; ++i)

{

// Checks if i is factor of both integers

if(n1%i==0 && n2%i==0)

gcd = i;

}

printf("G.C.D of %d and %d is %d", n1, n2, gcd);

return 0;

}

## GCD Using while loop and if...else Statement

#include <stdio.h>

int main()

{

int n1, n2;

printf("Enter two positive integers: ");

scanf("%d %d",&n1,&n2);

while(n1!=n2)

{

if(n1 > n2)

n1 -= n2;

else

n2 -= n1;

}

printf("GCD = %d",n1);

return 0;

}

**Output**

Enter two positive integers: 81

153

GCD = 9

## GCD for both positive and negative numbers

#include <stdio.h>

int main()

{

int n1, n2;

printf("Enter two integers: ");

scanf("%d %d",&n1,&n2);

// if user enters negative number, sign of the number is changed to positive

n1 = ( n1 > 0) ? n1 : -n1;

n2 = ( n2 > 0) ? n2 : -n2;

while(n1!=n2)

{

if(n1 > n2)

n1 -= n2;

else

n2 -= n1;

}

printf("GCD = %d",n1);

return 0;

}

**Output**

Enter two integers: 81

-153

GCD = 9

## LCM using while Loop and if Statement

#include <stdio.h>

int main()

{

int n1, n2, minMultiple;

printf("Enter two positive integers: ");

scanf("%d %d", &n1, &n2);

// maximum number between n1 and n2 is stored in minMultiple

minMultiple = (n1>n2) ? n1 : n2;

// Always true

while(1)

{

if( minMultiple%n1==0 && minMultiple%n2==0 )

{

printf("The LCM of %d and %d is %d.", n1, n2,minMultiple);

break;

}

++minMultiple;

}

return 0;

}

**Output**

Enter two positive integers: 72

120

The LCM of 72 and 120 is 360.

## LCM Calculation by Finding GCD

#include <stdio.h>

int main()

{

int n1, n2, i, gcd, lcm;

printf("Enter two positive integers: ");

scanf("%d %d",&n1,&n2);

for(i=1; i <= n1 && i <= n2; ++i)

{

// Checks if i is factor of both integers

if(n1%i==0 && n2%i==0)

gcd = i;

}

lcm = (n1\*n2)/gcd;

printf("The LCM of two numbers %d and %d is %d.", n1, n2, lcm);

return 0;

}

## Program to Display English Alphabets

#include <stdio.h>

int main()

{

char c;

for(c = 'A'; c <= 'Z'; ++c)

printf("%c ", c);

return 0;

}

**Output**

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

## Program to Display English Alphabets in Uppercase and Lowercase

#include <stdio.h>

int main()

{

char c;

printf("Enter u to display alphabets in uppercase. And enter l to display alphabets in lowercase: ");

scanf("%c", &c);

if(c== 'U' || c== 'u')

{

for(c = 'A'; c <= 'Z'; ++c)

printf("%c ", c);

}

else if (c == 'L' || c == 'l')

{

for(c = 'a'; c <= 'z'; ++c)

printf("%c ", c);

}

else

printf("Error! You entered invalid character.");

return 0;

}

**Output**

Enter u to display alphabets in uppercase. And enter l to display alphabets in lowercase: l

a b c d e f g h i j k l m n o p q r s t u v w x y z

## Program to Count Number of Digits in an Integer

#include <stdio.h>

int main()

{

long long n;

int count = 0;

printf("Enter an integer: ");

scanf("%lld", &n);

while(n != 0)

{

// n = n/10

n /= 10;

++count;

}

printf("Number of digits: %d", count);

}

**Output**

Enter an integer: 3452

Number of digits: 4

## Reverse an Integer

#include <stdio.h>

int main()

{

int n, reversedNumber = 0, remainder;

printf("Enter an integer: ");

scanf("%d", &n);

while(n != 0)

{

remainder = n%10;

reversedNumber = reversedNumber\*10 + remainder;

n /= 10;

}

printf("Reversed Number = %d", reversedNumber);

return 0;

}

**Output**

Enter an integer: 2345

Reversed Number = 5432

## Power of a Number Using while Loop

#include <stdio.h>

int main()

{

int base, exponent;

long long result = 1;

printf("Enter a base number: ");

scanf("%d", &base);

printf("Enter an exponent: ");

scanf("%d", &exponent);

while (exponent != 0)

{

result \*= base;

--exponent;

}

printf("Answer = %lld", result);

return 0;

}

**Output**

Enter a base number: 3

Enter an exponent: 4

Answer = 81

## Power Using pow() Function

#include <stdio.h>

#include <math.h>

int main()

{

double base, exponent, result;

printf("Enter a base number: ");

scanf("%lf", &base);

printf("Enter an exponent: ");

scanf("%lf", &exponent);

// calculates the power

result = pow(base, exponent);

printf("%.1lf^%.1lf = %.2lf", base, exponent, result);

return 0;

}

**Output**

Enter a base number: 2.3

Enter an exponent: 4.5

2.3^4.5 = 42.44

## Program to Check Palindrome

#include <stdio.h>

int main()

{

int n, reversedInteger = 0, remainder, originalInteger;

printf("Enter an integer: ");

scanf("%d", &n);

originalInteger = n;

// reversed integer is stored in variable

while( n!=0 )

{

remainder = n%10;

reversedInteger = reversedInteger\*10 + remainder;

n /= 10;

}

// palindrome if orignalInteger and reversedInteger are equal

if (originalInteger == reversedInteger)

printf("%d is a palindrome.", originalInteger);

else

printf("%d is not a palindrome.", originalInteger);

return 0;

}

**Output**

Enter an integer: 1001

1001 is a palindrome.

## Program to Check Prime Number

#include <stdio.h>

int main()

{

int n, i, flag = 0;

printf("Enter a positive integer: ");

scanf("%d",&n);

for(i=2; i<=n/2; ++i)

{

// condition for nonprime number

if(n%i==0)

{

flag=1;

break;

}

}

if (flag==0)

printf("%d is a prime number.",n);

else

printf("%d is not a prime number.",n);

return 0;

}

**Output**

Enter a positive integer: 29

29 is a prime number.

## Display Prime Numbers Between two Intervals

#include <stdio.h>

int main()

{

int low, high, i, flag;

printf("Enter two numbers(intervals): ");

scanf("%d %d", &low, &high);

printf("Prime numbers between %d and %d are: ", low, high);

while (low < high)

{

flag = 0;

for(i = 2; i <= low/2; ++i)

{

if(low % i == 0)

{

flag = 1;

break;

}

}

if (flag == 0)

printf("%d ", low);

++low;

}

return 0;

}

**Output**

Enter two numbers(intervals): 20

50

Prime numbers between 20 and 50 are: 23 29 31 37 41 43 47

## Display Prime Numbers Between two Intervals When Larger Number is Entered first

#include <stdio.h>

int main()

{

int low, high, i, flag, temp;

printf("Enter two numbers(intevals): ");

scanf("%d %d", &low, &high);

//swapping numbers if low is greater than high

if (low > high) {

temp = low;

low = high;

high = temp;

}

printf("Prime numbers between %d and %d are: ", low, high);

while (low < high)

{

flag = 0;

for(i = 2; i <= low/2; ++i)

{

if(low % i == 0)

{

flag = 1;

break;

}

}

if (flag == 0)

printf("%d ", low);

++low;

}

return 0;

}

## Check Armstrong Number of three digits

#include <stdio.h>

int main()

{

int number, originalNumber, remainder, result = 0;

printf("Enter a three digit integer: ");

scanf("%d", &number);

originalNumber = number;

while (originalNumber != 0)

{

remainder = originalNumber%10;

result += remainder\*remainder\*remainder;

originalNumber /= 10;

}

if(result == number)

printf("%d is an Armstrong number.",number);

else

printf("%d is not an Armstrong number.",number);

return 0;

}

**Output**

Enter a three digit integer: 371

371 is an Armstrong number.

## Check Armstrong Number of n digits

#include <stdio.h>

#include <math.h>

int main()

{

int number, originalNumber, remainder, result = 0, n = 0 ;

printf("Enter an integer: ");

scanf("%d", &number);

originalNumber = number;

while (originalNumber != 0)

{

originalNumber /= 10;

++n;

}

originalNumber = number;

while (originalNumber != 0)

{

remainder = originalNumber%10;

result += pow(remainder, n);

originalNumber /= 10;

}

if(result == number)

printf("%d is an Armstrong number.", number);

else

printf("%d is not an Armstrong number.", number);

return 0;

}

**Output**

Enter an integer: 1634

1634 is an Armstrong number.

## Armstrong Numbers Between Two Integers

#include <stdio.h>

#include <math.h>

int main()

{

int low, high, i, temp1, temp2, remainder, n = 0, result = 0;

printf("Enter two numbers(intervals): ");

scanf("%d %d", &low, &high);

printf("Armstrong numbers between %d an %d are: ", low, high);

for(i = low + 1; i < high; ++i)

{

temp2 = i;

temp1 = i;

// number of digits calculation

while (temp1 != 0)

{

temp1 /= 10;

++n;

}

// result contains sum of nth power of its digits

while (temp2 != 0)

{

remainder = temp2 % 10;

result += pow(remainder, n);

temp2 /= 10;

}

// checks if number i is equal to the sum of nth power of its digits

if (result == i) {

printf("%d ", i);

}

// resetting the values to check Armstrong number for next iteration

n = 0;

result = 0;

}

return 0;

}

**Output**

Enter two numbers(intervals): 999

99999

Armstrong numbers between 999 an 99999 are: 1634 8208 9474 54748 92727 93084

## Factors of a Positive Integer

#include <stdio.h>

int main()

{

int number, i;

printf("Enter a positive integer: ");

scanf("%d",&number);

printf("Factors of %d are: ", number);

for(i=1; i <= number; ++i)

{

if (number%i == 0)

{

printf("%d ",i);

}

}

return 0;

}

**Output**

Enter a positive integer: 60

Factors of 60 are: 1 2 3 4 5 6 10 12 15 20 30 60

### Program to print half pyramid using \*

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

#include <stdio.h>

int main()

{

int i, j, rows;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=1; i<=rows; ++i)

{

for(j=1; j<=i; ++j)

{

printf("\* ");

}

printf("\n");

}

return 0;

}

### Program to print half pyramid a using numbers

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

**Source Code**

#include <stdio.h>

int main()

{

int i, j, rows;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=1; i<=rows; ++i)

{

for(j=1; j<=i; ++j)

{

printf("%d ",j);

}

printf("\n");

}

return 0;

}

### Program to print half pyramid using alphabets

A

B B

C C C

D D D D

E E E E E

**Source Code**

#include <stdio.h>

int main()

{

int i, j;

char input, alphabet = 'A';

printf("Enter the uppercase character you want to print in last row: ");

scanf("%c",&input);

for(i=1; i <= (input-'A'+1); ++i)

{

for(j=1;j<=i;++j)

{

printf("%c", alphabet);

}

++alphabet;

printf("\n");

}

return 0;

}

## Programs to print inverted half pyramid using \* and numbers

### Inverted half pyramid using \*

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

**Source Code**

#include <stdio.h>

int main()

{

int i, j, rows;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=rows; i>=1; --i)

{

for(j=1; j<=i; ++j)

{

printf("\* ");

}

printf("\n");

}

return 0;

}

### Inverted half pyramid using numbers

1 2 3 4 5

1 2 3 4

1 2 3

1 2

1

**Source Code**

#include <stdio.h>

int main()

{

int i, j, rows;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=rows; i>=1; --i)

{

for(j=1; j<=i; ++j)

{

printf("%d ",j);

}

printf("\n");

}

return 0;

}

## Programs to display pyramid and inverted pyramid using \* and digits

### Program to print full pyramid using \*

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

**Source Code**

#include <stdio.h>

int main()

{

int i, space, rows, k=0;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=1; i<=rows; ++i, k=0)

{

for(space=1; space<=rows-i; ++space)

{

printf(" ");

}

while(k != 2\*i-1)

{

printf("\* ");

++k;

}

printf("\n");

}

return 0;

}

### Program to print pyramid using numbers

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4

5 6 7 8 9 8 7 6 5

**Source Code**

#include <stdio.h>

int main()

{

int i, space, rows, k=0, count = 0, count1 = 0;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=1; i<=rows; ++i)

{

for(space=1; space <= rows-i; ++space)

{

printf(" ");

++count;

}

while(k != 2\*i-1)

{

if (count <= rows-1)

{

printf("%d ", i+k);

++count;

}

else

{

++count1;

printf("%d ", (i+k-2\*count1));

}

++k;

}

count1 = count = k = 0;

printf("\n");

}

return 0;

}

### Inverted full pyramid using \*

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \*

\* \* \*

\*

**Source Code**

#include<stdio.h>

int main()

{

int rows, i, j, space;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=rows; i>=1; --i)

{

for(space=0; space < rows-i; ++space)

printf(" ");

for(j=i; j <= 2\*i-1; ++j)

printf("\* ");

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

printf("\* ");

printf("\n");

}

return 0;

}

### Print Pascal's triangle

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

**Source Code**

#include <stdio.h>

int main()

{

int rows, coef = 1, space, i, j;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=0; i<rows; i++)

{

for(space=1; space <= rows-i; space++)

printf(" ");

for(j=0; j <= i; j++)

{

if (j==0 || i==0)

coef = 1;

else

coef = coef\*(i-j+1)/j;

printf("%4d", coef);

}

printf("\n");

}

return 0;

}

### Print Floyd's Triangle.

1

2 3

4 5 6

7 8 9 10

**Source Code**

#include <stdio.h>

int main()

{

int rows, i, j, number= 1;

printf("Enter number of rows: ");

scanf("%d",&rows);

for(i=1; i <= rows; i++)

{

for(j=1; j <= i; ++j)

{

printf("%d ", number);

++number;

}

printf("\n");

}

return 0;

}

## Simple Calculator using switch Statement

// Performs addition, subtraction, multiplication or division depending the input from user

# include <stdio.h>

int main() {

char operator;

double firstNumber,secondNumber;

printf("Enter an operator (+, -, \*,): ");

scanf("%c", &operator);

printf("Enter two operands: ");

scanf("%lf %lf",&firstNumber, &secondNumber);

switch(operator)

{

case '+':

printf("%.1lf + %.1lf = %.1lf",firstNumber, secondNumber, firstNumber + secondNumber);

break;

case '-':

printf("%.1lf - %.1lf = %.1lf",firstNumber, secondNumber, firstNumber - secondNumber);

break;

case '\*':

printf("%.1lf \* %.1lf = %.1lf",firstNumber, secondNumber, firstNumber \* secondNumber);

break;

case '/':

printf("%.1lf / %.1lf = %.1lf",firstNumber, secondNumber, firstNumber / secondNumber);

break;

// operator doesn't match any case constant (+, -, \*, /)

default:

printf("Error! operator is not correct");

}

return 0;

}

**Output**

Enter an operator (+, -, \*,): \*

Enter two operands: 1.5

4.5

1.5 \* 4.5 = 6.8

## Prime Numbers Between Two Integers

#include <stdio.h>

int checkPrimeNumber(int n);

int main()

{

int n1, n2, i, flag;

printf("Enter two positive integers: ");

scanf("%d %d", &n1, &n2);

printf("Prime numbers between %d and %d are: ", n1, n2);

for(i=n1+1; i<n2; ++i)

{

// i is a prime number, flag will be equal to 1

flag = checkPrimeNumber(i);

if(flag == 1)

printf("%d ",i);

}

return 0;

}

// user-defined function to check prime number

int checkPrimeNumber(int n)

{

int j, flag = 1;

for(j=2; j <= n/2; ++j)

{

if (n%j == 0)

{

flag =0;

break;

}

}

return flag;

}

**Output**

Enter two positive integers: 12

30

Prime numbers between 12 and 30 are: 13 17 19 23 29

## Check Prime and Armstrong Number

#include <stdio.h>

#include <math.h>

int checkPrimeNumber(int n);

int checkArmstrongNumber(int n);

int main()

{

int n, flag;

printf("Enter a positive integer: ");

scanf("%d", &n);

// Check prime number

flag = checkPrimeNumber(n);

if (flag == 1)

printf("%d is a prime number.\n", n);

else

printf("%d is not a prime number.\n", n);

// Check Armstrong number

flag = checkArmstrongNumber(n);

if (flag == 1)

printf("%d is an Armstrong number.", n);

else

printf("%d is not an Armstrong number.",n);

return 0;

}

int checkPrimeNumber(int n)

{

int i, flag = 1;

for(i=2; i<=n/2; ++i)

{

// condition for non-prime number

if(n%i == 0)

{

flag = 0;

break;

}

}

return flag;

}

int checkArmstrongNumber(int number)

{

int originalNumber, remainder, result = 0, n = 0, flag;

originalNumber = number;

while (originalNumber != 0)

{

originalNumber /= 10;

++n;

}

originalNumber = number;

while (originalNumber != 0)

{

remainder = originalNumber%10;

result += pow(remainder, n);

originalNumber /= 10;

}

// condition for Armstrong number

if(result == number)

flag = 1;

else

flag = 0;

return flag;

}

**Output**

Enter a positive integer: 407

407 is not a prime number.

407 is an Armstrong number.

## Integer as a Sum of Two Prime Numbers

#include <stdio.h>

int checkPrime(int n);

int main()

{

int n, i, flag = 0;

printf("Enter a positive integer: ");

scanf("%d", &n);

for(i = 2; i <= n/2; ++i)

{

// condition for i to be a prime number

if (checkPrime(i) == 1)

{

// condition for n-i to be a prime number

if (checkPrime(n-i) == 1)

{

// n = primeNumber1 + primeNumber2

printf("%d = %d + %d\n", n, i, n - i);

flag = 1;

}

}

}

if (flag == 0)

printf("%d cannot be expressed as the sum of two prime numbers.", n);

return 0;

}

// Function to check prime number

int checkPrime(int n)

{

int i, isPrime = 1;

for(i = 2; i <= n/2; ++i)

{

if(n % i == 0)

{

isPrime = 0;

break;

}

}

return isPrime;

}

**Output**

Enter a positive integer: 34

34 = 3 + 31

34 = 5 + 29

34 = 11 + 23

34 = 17 + 17

## Sum of Natural Numbers Using Recursion

#include <stdio.h>

int addNumbers(int n);

int main()

{

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

printf("Sum = %d",addNumbers(num));

return 0;

}

int addNumbers(int n)

{

if(n != 0)

return n + addNumbers(n-1);

else

return n;

}

**Output**

Enter a positive integer: 20

Sum = 210

## Factorial of a Number Using Recursion

#include <stdio.h>

long int multiplyNumbers(int n);

int main()

{

int n;

printf("Enter a positive integer: ");

scanf("%d", &n);

printf("Factorial of %d = %ld", n, multiplyNumbers(n));

return 0;

}

long int multiplyNumbers(int n)

{

if (n >= 1)

return n\*multiplyNumbers(n-1);

else

return 1;

}

**Output**

Enter a positive integer: 6

Factorial of 6 = 720

## GCD of Two Numbers using Recursion

#include <stdio.h>

int hcf(int n1, int n2);

int main()

{

int n1, n2;

printf("Enter two positive integers: ");

scanf("%d %d", &n1, &n2);

printf("G.C.D of %d and %d is %d.", n1, n2, hcf(n1,n2));

return 0;

}

int hcf(int n1, int n2)

{

if (n2 != 0)

return hcf(n2, n1%n2);

else

return n1;

}

**Output**

Enter two positive integers: 366

60

G.C.D of 366 and 60 is 6.

## Program to convert binary number to decimal

#include <stdio.h>

#include <math.h>

int convertBinaryToDecimal(long long n);

int main()

{

long long n;

printf("Enter a binary number: ");

scanf("%lld", &n);

printf("%lld in binary = %d in decimal", n, convertBinaryToDecimal(n));

return 0;

}

int convertBinaryToDecimal(long long n)

{

int decimalNumber = 0, i = 0, remainder;

while (n!=0)

{

remainder = n%10;

n /= 10;

decimalNumber += remainder\*pow(2,i);

++i;

}

return decimalNumber;

}

**Output**

Enter a binary number: 110110111

110110111 in binary = 439

## Program to convert decimal number to binary

#include <stdio.h>

#include <math.h>

long long convertDecimalToBinary(int n);

int main()

{

int n;

printf("Enter a decimal number: ");

scanf("%d", &n);

printf("%d in decimal = %lld in binary", n, convertDecimalToBinary(n));

return 0;

}

long long convertDecimalToBinary(int n)

{

long long binaryNumber = 0;

int remainder, i = 1, step = 1;

while (n!=0)

{

remainder = n%2;

printf("Step %d: %d/2, Remainder = %d, Quotient = %d\n", step++, n, remainder, n/2);

n /= 2;

binaryNumber += remainder\*i;

i \*= 10;

}

return binaryNumber;

}

**Output**

Enter a decimal number: 19

Step 1: 19/2, Remainder = 1, Quotient = 9

Step 2: 9/2, Remainder = 1, Quotient = 4

Step 3: 4/2, Remainder = 0, Quotient = 2

Step 4: 2/2, Remainder = 0, Quotient = 1

Step 5: 1/2, Remainder = 1, Quotient = 0

19 in decimal = 10011 in binary

## Program to Convert Decimal to Octal

#include <stdio.h>

#include <math.h>

int convertDecimalToOctal(int decimalNumber);

int main()

{

int decimalNumber;

printf("Enter a decimal number: ");

scanf("%d", &decimalNumber);

printf("%d in decimal = %d in octal", decimalNumber, convertDecimalToOctal(decimalNumber));

return 0;

}

int convertDecimalToOctal(int decimalNumber)

{

int octalNumber = 0, i = 1;

while (decimalNumber != 0)

{

octalNumber += (decimalNumber % 8) \* i;

decimalNumber /= 8;

i \*= 10;

}

return octalNumber;

}

**Output**

Enter a decimal number: 78

78 in decimal = 116 in octal

## Program to Convert Octal to Decimal

#include <stdio.h>

#include <math.h>

long long convertOctalToDecimal(int octalNumber);

int main()

{

int octalNumber;

printf("Enter an octal number: ");

scanf("%d", &octalNumber);

printf("%d in octal = %lld in decimal", octalNumber, convertOctalToDecimal(octalNumber));

return 0;

}

long long convertOctalToDecimal(int octalNumber)

{

int decimalNumber = 0, i = 0;

while(octalNumber != 0)

{

decimalNumber += (octalNumber%10) \* pow(8,i);

++i;

octalNumber/=10;

}

i = 1;

return decimalNumber;

}

**Output**

Enter an octal number: 116

116 in octal = 78 in decimal

## Program to Convert Binary to Octal

#include <stdio.h>

#include <math.h>

int convertBinarytoOctal(long long binaryNumber);

int main()

{

long long binaryNumber;

printf("Enter a binary number: ");

scanf("%lld", &binaryNumber);

printf("%lld in binary = %d in octal", binaryNumber, convertBinarytoOctal(binaryNumber));

return 0;

}

int convertBinarytoOctal(long long binaryNumber)

{

int octalNumber = 0, decimalNumber = 0, i = 0;

while(binaryNumber != 0)

{

decimalNumber += (binaryNumber%10) \* pow(2,i);

++i;

binaryNumber/=10;

}

i = 1;

while (decimalNumber != 0)

{

octalNumber += (decimalNumber % 8) \* i;

decimalNumber /= 8;

i \*= 10;

}

return octalNumber;

}

**Output**

Enter a binary number: 101001

101001 in binary = 51 in octal

## Program to Convert Octal to Binary

#include <stdio.h>

#include <math.h>

long long convertOctalToBinary(int octalNumber);

int main()

{

int octalNumber;

printf("Enter an octal number: ");

scanf("%d", &octalNumber);

printf("%d in octal = %lld in binary", octalNumber, convertOctalToBinary(octalNumber));

return 0;

}

long long convertOctalToBinary(int octalNumber)

{

int decimalNumber = 0, i = 0;

long long binaryNumber = 0;

while(octalNumber != 0)

{

decimalNumber += (octalNumber%10) \* pow(8,i);

++i;

octalNumber/=10;

}

i = 1;

while (decimalNumber != 0)

{

binaryNumber += (decimalNumber % 2) \* i;

decimalNumber /= 2;

i \*= 10;

}

return binaryNumber;

}

**Output**

Enter an octal number: 67

67 in octal = 110111 in binary

## Reverse a sentence using recursion

/\* Example to reverse a sentence entered by user without using strings. \*/

#include <stdio.h>

void reverseSentence();

int main()

{

printf("Enter a sentence: ");

reverseSentence();

return 0;

}

void reverseSentence()

{

char c;

scanf("%c", &c);

if( c != '\n')

{

reverseSentence();

printf("%c",c);

}

}

**Output**

Enter a sentence: margorp emosewa

awesome program

## Program to calculate power using recursion

#include <stdio.h>

int power(int n1, int n2);

int main()

{

int base, powerRaised, result;

printf("Enter base number: ");

scanf("%d",&base);

printf("Enter power number(positive integer): ");

scanf("%d",&powerRaised);

result = power(base, powerRaised);

printf("%d^%d = %d", base, powerRaised, result);

return 0;

}

int power(int base, int powerRaised)

{

if (powerRaised != 0)

return (base\*power(base, powerRaised-1));

else

return 1;

}

**Output**

Enter base number: 3

Enter power number(positive integer): 4

3^4 = 81

## Source Code to Calculate Average Using Arrays

#include <stdio.h>

int main()

{

int n, i;

float num[100], sum = 0.0, average;

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

scanf("%d", &n);

while (n > 100 || n <= 0)

{

printf("Error! number should in range of (1 to 100).\n");

printf("Enter the number again: ");

scanf("%d", &n);

}

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

{

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

scanf("%f", &num[i]);

sum += num[i];

}

average = sum / n;

printf("Average = %.2f", average);

return 0;

}

**Output**

Enter the numbers of elements: 6

1. Enter number: 45.3

2. Enter number: 67.5

3. Enter number: -45.6

4. Enter number: 20.34

5. Enter number: 33

6. Enter number: 45.6

Average = 27.69

## Display Largest Element of an array

#include <stdio.h>

int main()

{

int i, n;

float arr[100];

printf("Enter total number of elements(1 to 100): ");

scanf("%d", &n);

printf("\n");

// Stores number entered by the user

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

{

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

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

}

// Loop to store largest number to arr[0]

for(i = 1; i < n; ++i)

{

// Change < to > if you want to find the smallest element

if(arr[0] < arr[i])

arr[0] = arr[i];

}

printf("Largest element = %.2f", arr[0]);

return 0;

}

**Output**

Enter total number of elements(1 to 100): 8

Enter Number 1: 23.4

Enter Number 2: -34.5

Enter Number 3: 50

Enter Number 4: 33.5

Enter Number 5: 55.5

Enter Number 6: 43.7

Enter Number 7: 5.7

Enter Number 8: -66.5

## Program to Calculate Standard Deviation

#include <stdio.h>

#include <math.h>

float calculateSD(float data[]);

int main()

{

int i;

float data[10];

printf("Enter 10 elements: ");

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

scanf("%f", &data[i]);

printf("\nStandard Deviation = %.6f", calculateSD(data));

return 0;

}

float calculateSD(float data[])

{

float sum = 0.0, mean, standardDeviation = 0.0;

int i;

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

{

sum += data[i];

}

mean = sum/10;

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

standardDeviation += pow(data[i] - mean, 2);

return sqrt(standardDeviation/10);

}

**Output**

Enter 10 elements: 1

2

3

4

5

6

7

8

9

10

Standard Deviation = 2.872281

## Program to Add Two Matrices

#include <stdio.h>

int main(){

int r, c, a[100][100], b[100][100], sum[100][100], i, j;

printf("Enter number of rows (between 1 and 100): ");

scanf("%d", &r);

printf("Enter number of columns (between 1 and 100): ");

scanf("%d", &c);

printf("\nEnter elements of 1st matrix:\n");

for(i=0; i<r; ++i)

for(j=0; j<c; ++j)

{

printf("Enter element a%d%d: ",i+1,j+1);

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

}

printf("Enter elements of 2nd matrix:\n");

for(i=0; i<r; ++i)

for(j=0; j<c; ++j)

{

printf("Enter element a%d%d: ",i+1, j+1);

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

}

// Adding Two matrices

for(i=0;i<r;++i)

for(j=0;j<c;++j)

{

sum[i][j]=a[i][j]+b[i][j];

}

// Displaying the result

printf("\nSum of two matrix is: \n\n");

for(i=0;i<r;++i)

for(j=0;j<c;++j)

{

printf("%d ",sum[i][j]);

if(j==c-1)

{

printf("\n\n");

}

}

return 0;

}

**Output**

Enter number of rows (between 1 and 100): 2

Enter number of columns (between 1 and 100): 3

Enter elements of 1st matrix:

Enter element a11: 2

Enter element a12: 3

Enter element a13: 4

Enter element a21: 5

Enter element a22: 2

Enter element a23: 3

Enter elements of 2nd matrix:

Enter element a11: -4

Enter element a12: 5

Enter element a13: 3

Enter element a21: 5

Enter element a22: 6

Enter element a23: 3

Sum of two matrix is:

-2 8 7

10 8 6

## Program to Multiply Two Matrices

#include <stdio.h>

int main()

{

int a[10][10], b[10][10], result[10][10], r1, c1, r2, c2, i, j, k;

printf("Enter rows and column for first matrix: ");

scanf("%d %d", &r1, &c1);

printf("Enter rows and column for second matrix: ");

scanf("%d %d",&r2, &c2);

// Column of first matrix should be equal to column of second matrix and

while (c1 != r2)

{

printf("Error! column of first matrix not equal to row of second.\n\n");

printf("Enter rows and column for first matrix: ");

scanf("%d %d", &r1, &c1);

printf("Enter rows and column for second matrix: ");

scanf("%d %d",&r2, &c2);

}

// Storing elements of first matrix.

printf("\nEnter elements of matrix 1:\n");

for(i=0; i<r1; ++i)

for(j=0; j<c1; ++j)

{

printf("Enter elements a%d%d: ",i+1, j+1);

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

}

// Storing elements of second matrix.

printf("\nEnter elements of matrix 2:\n");

for(i=0; i<r2; ++i)

for(j=0; j<c2; ++j)

{

printf("Enter elements b%d%d: ",i+1, j+1);

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

}

// Initializing all elements of result matrix to 0

for(i=0; i<r1; ++i)

for(j=0; j<c2; ++j)

{

result[i][j] = 0;

}

// Multiplying matrices a and b and

// storing result in result matrix

for(i=0; i<r1; ++i)

for(j=0; j<c2; ++j)

for(k=0; k<c1; ++k)

{

result[i][j]+=a[i][k]\*b[k][j];

}

// Displaying the result

printf("\nOutput Matrix:\n");

for(i=0; i<r1; ++i)

for(j=0; j<c2; ++j)

{

printf("%d ", result[i][j]);

if(j == c2-1)

printf("\n\n");

}

return 0;

}

**Output**

Enter rows and column for first matrix: 3

2

Enter rows and column for second matrix: 3

2

Error! column of first matrix not equal to row of second.

Enter rows and column for first matrix: 2

3

Enter rows and column for second matrix: 3

2

Enter elements of matrix 1:

Enter elements a11: 3

Enter elements a12: -2

Enter elements a13: 5

Enter elements a21: 3

Enter elements a22: 0

Enter elements a23: 4

Enter elements of matrix 2:

Enter elements b11: 2

Enter elements b12: 3

Enter elements b21: -9

Enter elements b22: 0

Enter elements b31: 0

Enter elements b32: 4

Output Matrix:

24 29

6 25

## Program to Find Transpose of a Matrix

#include <stdio.h>

int main()

{

int a[10][10], transpose[10][10], r, c, i, j;

printf("Enter rows and columns of matrix: ");

scanf("%d %d", &r, &c);

// Storing elements of the matrix

printf("\nEnter elements of matrix:\n");

for(i=0; i<r; ++i)

for(j=0; j<c; ++j)

{

printf("Enter element a%d%d: ",i+1, j+1);

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

}

// Displaying the matrix a[][] \*/

printf("\nEntered Matrix: \n");

for(i=0; i<r; ++i)

for(j=0; j<c; ++j)

{

printf("%d ", a[i][j]);

if (j == c-1)

printf("\n\n");

}

// Finding the transpose of matrix a

for(i=0; i<r; ++i)

for(j=0; j<c; ++j)

{

transpose[j][i] = a[i][j];

}

// Displaying the transpose of matrix a

printf("\nTranspose of Matrix:\n");

for(i=0; i<c; ++i)

for(j=0; j<r; ++j)

{

printf("%d ",transpose[i][j]);

if(j==r-1)

printf("\n\n");

}

return 0;

}

**Output**

Enter rows and columns of matrix: 2

3

Enter element of matrix:

Enter element a11: 2

Enter element a12: 3

Enter element a13: 4

Enter element a21: 5

Enter element a22: 6

Enter element a23: 4

Entered Matrix:

2 3 4

5 6 4

Transpose of Matrix:

2 5

3 6

4 4

### Multiply Matrices by Passing it to a Function

#include <stdio.h>

void enterData(int firstMatrix[][10], int secondMatrix[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond);

void multiplyMatrices(int firstMatrix[][10], int secondMatrix[][10], int multResult[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond);

void display(int mult[][10], int rowFirst, int columnSecond);

int main()

{

int firstMatrix[10][10], secondMatrix[10][10], mult[10][10], rowFirst, columnFirst, rowSecond, columnSecond, i, j, k;

printf("Enter rows and column for first matrix: ");

scanf("%d %d", &rowFirst, &columnFirst);

printf("Enter rows and column for second matrix: ");

scanf("%d %d", &rowSecond, &columnSecond);

// If colum of first matrix in not equal to row of second matrix, asking user to enter the size of matrix again.

while (columnFirst != rowSecond)

{

printf("Error! column of first matrix not equal to row of second.\n");

printf("Enter rows and column for first matrix: ");

scanf("%d%d", &rowFirst, &columnFirst);

printf("Enter rows and column for second matrix: ");

scanf("%d%d", &rowSecond, &columnSecond);

}

// Function to take matrices data

enterData(firstMatrix, secondMatrix, rowFirst, columnFirst, rowSecond, columnSecond);

// Function to multiply two matrices.

multiplyMatrices(firstMatrix, secondMatrix, mult, rowFirst, columnFirst, rowSecond, columnSecond);

// Function to display resultant matrix after multiplication.

display(mult, rowFirst, columnSecond);

return 0;

}

void enterData(int firstMatrix[][10], int secondMatrix[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond)

{

int i, j;

printf("\nEnter elements of matrix 1:\n");

for(i = 0; i < rowFirst; ++i)

{

for(j = 0; j < columnFirst; ++j)

{

printf("Enter elements a%d%d: ", i + 1, j + 1);

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

}

}

printf("\nEnter elements of matrix 2:\n");

for(i = 0; i < rowSecond; ++i)

{

for(j = 0; j < columnSecond; ++j)

{

printf("Enter elements b%d%d: ", i + 1, j + 1);

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

}

}

}

void multiplyMatrices(int firstMatrix[][10], int secondMatrix[][10], int mult[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond)

{

int i, j, k;

// Initializing elements of matrix mult to 0.

for(i = 0; i < rowFirst; ++i)

{

for(j = 0; j < columnSecond; ++j)

{

mult[i][j] = 0;

}

}

// Multiplying matrix firstMatrix and secondMatrix and storing in array mult.

for(i = 0; i < rowFirst; ++i)

{

for(j = 0; j < columnSecond; ++j)

{

for(k=0; k<columnFirst; ++k)

{

mult[i][j] += firstMatrix[i][k] \* secondMatrix[k][j];

}

}

}

}

void display(int mult[][10], int rowFirst, int columnSecond)

{

int i, j;

printf("\nOutput Matrix:\n");

for(i = 0; i < rowFirst; ++i)

{

for(j = 0; j < columnSecond; ++j)

{

printf("%d ", mult[i][j]);

if(j == columnSecond - 1)

printf("\n\n");

}

}

}

**Output**

Enter rows and column for first matrix: 3

2

Enter rows and column for second matrix: 3

2

Error! column of first matrix not equal to row of second.

Enter rows and column for first matrix: 2

3

Enter rows and column for second matrix: 3

2

Enter elements of matrix 1:

Enter elements a11: 3

Enter elements a12: -2

Enter elements a13: 5

Enter elements a21: 3

Enter elements a22: 0

Enter elements a23: 4

Enter elements of matrix 2:

Enter elements b11: 2

Enter elements b12: 3

Enter elements b21: -9

Enter elements b22: 0

Enter elements b31: 0

Enter elements b32: 4

Output Matrix:

24 29

6 25

## Access Array Elements Using Pointers

#include <stdio.h>

int main()

{

int data[5], i;

printf("Enter elements: ");

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

scanf("%d", data + i);

printf("You entered: \n");

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

printf("%d\n", \*(data + i));

return 0;

}

**Output**

Enter elements: 1

2

3

5

4

You entered:

1

2

3

5

4

## Program to Swap Elements Using Call by Reference

#include<stdio.h>

void cyclicSwap(int \*a,int \*b,int \*c);

int main()

{

int a, b, c;

printf("Enter a, b and c respectively: ");

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

printf("Value before swapping:\n");

printf("a = %d \nb = %d \nc = %d\n",a,b,c);

cyclicSwap(&a, &b, &c);

printf("Value after swapping:\n");

printf("a = %d \nb = %d \nc = %d",a, b, c);

return 0;

}

void cyclicSwap(int \*a,int \*b,int \*c)

{

int temp;

// swapping in cyclic order

temp = \*b;

\*b = \*a;

\*a = \*c;

\*c = temp;

}

**Output**

Enter a, b and c respectively: 1

2

3

Value before swapping:

a = 1

b = 2

c = 3

Value after swapping:

a = 3

b = 1

c = 2

## Find Largest Element Using Dynamic Memory Allocation - calloc()

#include <stdio.h>

#include <stdlib.h>

int main()

{

int i, num;

float \*data;

printf("Enter total number of elements(1 to 100): ");

scanf("%d", &num);

// Allocates the memory for 'num' elements.

data = (float\*) calloc(num, sizeof(float));

if(data == NULL)

{

printf("Error!!! memory not allocated.");

exit(0);

}

printf("\n");

// Stores the number entered by the user.

for(i = 0; i < num; ++i)

{

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

scanf("%f", data + i);

}

// Loop to store largest number at address data

for(i = 1; i < num; ++i)

{

// Change < to > if you want to find the smallest number

if(\*data < \*(data + i))

\*data = \*(data + i);

}

printf("Largest element = %.2f", \*data);

return 0;

}

**Output**

Enter total number of elements(1 to 100): 10

Enter Number 1: 2.34

Enter Number 2: 3.43

Enter Number 3: 6.78

Enter Number 4: 2.45

Enter Number 5: 7.64

Enter Number 6: 9.05

Enter Number 7: -3.45

Enter Number 8: -9.99

Enter Number 9: 5.67

Enter Number 10: 34.95

Largest element: 34.95

## Find the Frequency of Characters

#include <stdio.h>

int main()

{

char str[1000], ch;

int i, frequency = 0;

printf("Enter a string: ");

gets(str);

printf("Enter a character to find the frequency: ");

scanf("%c",&ch);

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

{

if(ch == str[i])

++frequency;

}

printf("Frequency of %c = %d", ch, frequency);

return 0;

}

**Output**

Enter a string: This website is awesome.

Enter a character to find the frequency: e

Frequency of e = 4

## Program to count vowels, consonants etc.

#include <stdio.h>

int main()

{

char line[150];

int i, vowels, consonants, digits, spaces;

vowels = consonants = digits = spaces = 0;

printf("Enter a line of string: ");

scanf("%[^\n]", line);

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

{

if(line[i]=='a' || line[i]=='e' || line[i]=='i' ||

line[i]=='o' || line[i]=='u' || line[i]=='A' ||

line[i]=='E' || line[i]=='I' || line[i]=='O' ||

line[i]=='U')

{

++vowels;

}

else if((line[i]>='a'&& line[i]<='z') || (line[i]>='A'&& line[i]<='Z'))

{

++consonants;

}

else if(line[i]>='0' && line[i]<='9')

{

++digits;

}

else if (line[i]==' ')

{

++spaces;

}

}

printf("Vowels: %d",vowels);

printf("\nConsonants: %d",consonants);

printf("\nDigits: %d",digits);

printf("\nWhite spaces: %d", spaces);

return 0;

}

**Output**

Enter a line of string: adfslkj34 34lkj343 34lk

Vowels: 1

Consonants: 11

Digits: 9

White spaces: 2

## Remove Characters in String Except Alphabets

#include<stdio.h>

int main()

{

char line[150];

int i, j;

printf("Enter a string: ");

gets(line);

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

{

while (!( (line[i] >= 'a' && line[i] <= 'z') || (line[i] >= 'A' && line[i] <= 'Z') || line[i] == '\0') )

{

for(j = i; line[j] != '\0'; ++j)

{

line[j] = line[j+1];

}

line[j] = '\0';

}

}

printf("Output String: ");

puts(line);

return 0;

}

**Output**

Enter a string: p2'r-o@gram84iz./

Output String: programiz

## Calculate Length of String without Using strlen() Function

#include <stdio.h>

int main()

{

char s[1000], i;

printf("Enter a string: ");

scanf("%s", s);

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

printf("Length of string: %d", i);

return 0;

}

**Output**

Enter a string: Programiz

Length of string: 9

## Concatenate Two Strings Without Using strcat()

#include <stdio.h>

int main()

{

char s1[100], s2[100], i, j;

printf("Enter first string: ");

scanf("%s", s1);

printf("Enter second string: ");

scanf("%s", s2);

// calculate the length of string s1

// and store it in i

for(i = 0; s1[i] != '\0'; ++i);

for(j = 0; s2[j] != '\0'; ++j, ++i)

{

s1[i] = s2[j];

}

s1[i] = '\0';

printf("After concatenation: %s", s1);

return 0;

}

**Output**

Enter first string: lol

Enter second string: :)

After concatenation: lol:)

## Copy String Manually Without Using strcpy()

#include <stdio.h>

int main()

{

char s1[100], s2[100], i;

printf("Enter string s1: ");

scanf("%s",s1);

for(i = 0; s1[i] != '\0'; ++i)

{

s2[i] = s1[i];

}

s2[i] = '\0';

printf("String s2: %s", s2);

return 0;

}

**Output**

Enter String s1: programiz

String s2: programiz

## Program to Sort Strings in Dictionary Order

#include<stdio.h>

#include <string.h>

int main()

{

int i, j;

char str[10][50], temp[50];

printf("Enter 10 words:\n");

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

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

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

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

{

if(strcmp(str[i], str[j])>0)

{

strcpy(temp, str[i]);

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

strcpy(str[j], temp);

}

}

printf("\nIn lexicographical order: \n");

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

{

puts(str[i]);

}

return 0;

}

**Output**

Enter 10 words:

C

C++

Java

PHP

Python

Perl

Ruby

R

JavaScript

PHP

In lexicographical order:

C

C++

Java

JavaScript

PHP

PHP

Perl

Python

R

Ruby

## Store Information and Display it Using Structure

#include <stdio.h>

struct student

{

char name[50];

int roll;

float marks;

} s;

int main()

{

printf("Enter information:\n");

printf("Enter name: ");

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

printf("Enter roll number: ");

scanf("%d", &s.roll);

printf("Enter marks: ");

scanf("%f", &s.marks);

printf("Displaying Information:\n");

printf("Name: ");

puts(s.name);

printf("Roll number: %d\n",s.roll);

printf("Marks: %.1f\n", s.marks);

return 0;

}

**Output**

Enter information:

Enter name: Jack

Enter roll number: 23

Enter marks: 34.5

Displaying Information:

Name: Jack

Roll number: 23

Marks: 34.5

## Program to add two distances in inch-feet system

#include <stdio.h>

struct Distance

{

int feet;

float inch;

} d1, d2, sumOfDistances;

int main()

{

printf("Enter information for 1st distance\n");

printf("Enter feet: ");

scanf("%d", &d1.feet);

printf("Enter inch: ");

scanf("%f", &d1.inch);

printf("\nEnter information for 2nd distance\n");

printf("Enter feet: ");

scanf("%d", &d2.feet);

printf("Enter inch: ");

scanf("%f", &d2.inch);

sumOfDistances.feet = d1.feet+d2.feet;

sumOfDistances.inch = d1.inch+d2.inch;

// If inch is greater than 12, changing it to feet.

if (sumOfDistances.inch>12.0)

{

sumOfDistances.inch = sumOfDistances.inch-12.0;

++sumOfDistances.feet;

}

printf("\nSum of distances = %d\'-%.1f\"",sumOfDistances.feet, sumOfDistances.inch);

return 0;

}

**Output**

Enter information for 1st distance

Enter feet: 23

Enter inch: 8.6

Enter information for 2nd distance

Enter feet: 34

Enter inch: 2.4

Sum of distances = 57'-11.0"

## Add Two Complex Numbers

#include <stdio.h>

typedef struct complex

{

float real;

float imag;

} complex;

complex add(complex n1,complex n2);

int main()

{

complex n1, n2, temp;

printf("For 1st complex number \n");

printf("Enter real and imaginary part respectively:\n");

scanf("%f %f", &n1.real, &n1.imag);

printf("\nFor 2nd complex number \n");

printf("Enter real and imaginary part respectively:\n");

scanf("%f %f", &n2.real, &n2.imag);

temp = add(n1, n2);

printf("Sum = %.1f + %.1fi", temp.real, temp.imag);

return 0;

}

complex add(complex n1, complex n2)

{

complex temp;

temp.real = n1.real + n2.real;

temp.imag = n1.imag + n2.imag;

return(temp);

}

**Output**

For 1st complex number

Enter real and imaginary part respectively: 2.3

4.5

For 2nd complex number

Enter real and imaginary part respectively: 3.4

5

Sum = 5.7 + 9.5i

## Calculate Difference Between Two Time Periods

#include <stdio.h>

struct TIME

{

int seconds;

int minutes;

int hours;

};

void differenceBetweenTimePeriod(struct TIME t1, struct TIME t2, struct TIME \*diff);

int main()

{

struct TIME startTime, stopTime, diff;

printf("Enter start time: \n");

printf("Enter hours, minutes and seconds respectively: ");

scanf("%d %d %d", &startTime.hours, &startTime.minutes, &startTime.seconds);

printf("Enter stop time: \n");

printf("Enter hours, minutes and seconds respectively: ");

scanf("%d %d %d", &stopTime.hours, &stopTime.minutes, &stopTime.seconds);

// Calculate the difference between the start and stop time period.

differenceBetweenTimePeriod(startTime, stopTime, &diff);

printf("\nTIME DIFFERENCE: %d:%d:%d - ", startTime.hours, startTime.minutes, startTime.seconds);

printf("%d:%d:%d ", stopTime.hours, stopTime.minutes, stopTime.seconds);

printf("= %d:%d:%d\n", diff.hours, diff.minutes, diff.seconds);

return 0;

}

void differenceBetweenTimePeriod(struct TIME start, struct TIME stop, struct TIME \*diff)

{

if(stop.seconds > start.seconds){

--start.minutes;

start.seconds += 60;

}

diff->seconds = start.seconds - stop.seconds;

if(stop.minutes > start.minutes){

--start.hours;

start.minutes += 60;

}

diff->minutes = start.minutes - stop.minutes;

diff->hours = start.hours - stop.hours;

}

**Output**

Enter start time:

Enter hours, minutes and seconds respectively: 12

34

55

Enter stop time:

Enter hours, minutes and seconds respectively:8

12

15

TIME DIFFERENCE: 12:34:55 - 8:12:15 = 4:22:40

## Store Information in Structure and Display it

#include <stdio.h>

struct student

{

char name[50];

int roll;

float marks;

} s[10];

int main()

{

int i;

printf("Enter information of students:\n");

// storing information

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

{

s[i].roll = i+1;

printf("\nFor roll number%d,\n",s[i].roll);

printf("Enter name: ");

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

printf("Enter marks: ");

scanf("%f",&s[i].marks);

printf("\n");

}

printf("Displaying Information:\n\n");

// displaying information

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

{

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

printf("Name: ");

puts(s[i].name);

printf("Marks: %.1f",s[i].marks);

printf("\n");

}

return 0;

}

**Output**

Enter information of students:

For roll number1,

Enter name: Tom

Enter marks: 98

For roll number2,

Enter name: Jerry

Enter marks: 89

.

.

.

Displaying Information:

Roll number: 1

Name: Tom

Marks: 98

.

.

.

## Demonstrate the Dynamic Memory Allocation for Structure

#include <stdio.h>

#include<stdlib.h>

struct course

{

int marks;

char subject[30];

};

int main()

{

struct course \*ptr;

int i, noOfRecords;

printf("Enter number of records: ");

scanf("%d", &noOfRecords);

// Allocates the memory for noOfRecords structures with pointer ptr pointing to the base address.

ptr = (struct course\*) malloc (noOfRecords \* sizeof(struct course));

for(i = 0; i < noOfRecords; ++i)

{

printf("Enter name of the subject and marks respectively:\n");

scanf("%s %d", &(ptr+i)->subject, &(ptr+i)->marks);

}

printf("Displaying Information:\n");

for(i = 0; i < noOfRecords ; ++i)

printf("%s\t%d\n", (ptr+i)->subject, (ptr+i)->marks);

return 0;

}

**Output**

Enter number of records: 2

Enter name of the subject and marks respectively:

Programming

22

Enter name of the subject and marks respectively:

Structure

33

Displaying Information:

Programming 22

Structure 33

# **C Program to Write a Sentence to a File**

#include <stdio.h>

#include <stdlib.h> /\* For exit() function \*/

int main()

{

char sentence[1000];

FILE \*fptr;

fptr = fopen("program.txt", "w");

if(fptr == NULL)

{

printf("Error!");

exit(1);

}

printf("Enter a sentence:\n");

gets(sentence);

fprintf(fptr,"%s", sentence);

fclose(fptr);

return 0;

}

**Output**

Enter sentence:

I am awesome and so are files.

## Program to read text from a file

#include <stdio.h>

#include <stdlib.h> // For exit() function

int main()

{

char c[1000];

FILE \*fptr;

if ((fptr = fopen("program.txt", "r")) == NULL)

{

printf("Error! opening file");

// Program exits if file pointer returns NULL.

exit(1);

}

// reads text until newline

fscanf(fptr,"%[^\n]", c);

printf("Data from the file:\n%s", c);

fclose(fptr);

return 0;

}

the program.txt file contains following text.

C programming is awesome.

I love C programming.

How are you doing?

The output of the program will be:

Data from the file: C programming is awesome.

A predefined macro \_\_FILE\_\_ contains the location of a C programming file, it is working on. For example:

#include <stdio.h>

int main(){

printf("%s",\_\_FILE\_\_);

}

The output of this program is the location of this C programming file.

### C program to display its own source code using \_\_FILE\_\_

#include <stdio.h>

int main() {

FILE \*fp;

char c;

fp = fopen(\_\_FILE\_\_,"r");

do {

c = getc(fp);

putchar(c);

}

while(c != EOF);

fclose(fp);

return 0;

}

this program displays the content of this particular C programming file(source code) because \_\_FILE\_\_ contains the location of this C programming file in a string.