RECURSION

1. Write a c program to find 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: 23

Factorial of 23 = 8128291617894825984

1. Write a c program to find 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: 23 12

G.C.D of 23 and 12 is 1.

1. Write a c program to find out sum of digits using recursion

#include <stdio.h>

int sum (int a);

int main()

{

int num, result;

printf("Enter the number: ");

scanf("%d", &num);

result = sum(num);

printf("Sum of digits in %d is %d\n", num, result);

}

int sum (int num)

{

if (num != 0)

{

return (num % 10 + sum (num / 10));

}

else

{

return 0;

}

}

Output

Enter the number: 234345

Sum of digits in 234345 is 21

1. Write a c program to find power of a number using recursion

#include <stdio.h>

int power1(int n1, int n2);

int main()

{

int base, power, result;

printf("Enter base number: ");

scanf("%d",&base);

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

scanf("%d",&power);

result = power1(base, power);

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

return 0;

}

int power1(int base, int power)

{

if (power != 0)

{

return (base\*power1(base, power-1));

}

else

{

return 1;

}

}

Output

Enter base number: 2

Enter power number(positive integer): 5

2^5 = 32

1. Write a c program to reverse a number using recursion

#include<stdio.h>

int main(){

int num,reverse\_number;

printf("\nEnter any number:");

scanf("%d",&num);

reverse\_number=reverse\_function(num);

printf("\nAfter reversing the no is :%d",reverse\_number);

return 0;

}

int sum=0,rem;

reverse\_function(int num){

if(num){

rem=num%10;

sum=sum\*10+rem;

reverse\_function(num/10);

}

else

return sum;

return sum;

}

Output

Enter any number:34532

After reversing the no is :23543

1. Write a c program to reverse a string using recursion

#include <stdio.h>

#include <string.h>

int main()

{

char arr[100];

printf("Enter a string to reverse\n");

gets(arr);

strrev(arr);

printf("Reverse of the string is \n%s\n", arr);

}

Output

Enter a string to reverse hello

Reverse of the string is olleh

1. Write a c program to print 1 to 100 without using a loop

#include <stdio.h>

int main()

{

int i = 0;

start:

i = i + 1;

printf("%d\t", i);

if (i < 100)

goto start;

}

1 2 3 4 5 6 7 8 9 10

11 12 13 14 15 16 17 18 19 20

21 22 23 24 25 26 27 28 29 30

31 32 33 34 35 36 37 38 39 40

41 42 43 44 45 46 47 48 49 50

51 52 53 54 55 56 57 58 59 60

61 62 63 64 65 66 67 68 69 70

71 72 73 74 75 76 77 78 79 80

81 82 83 84 85 86 87 88 89 90

91 92 93 94 95 96 97 98 99 100

SIZE OF DATA TYPES

1. Write a c program to find the size of int without using size of operator

#include<stdio.h>

int main()

{

int \*ptr = 0;

ptr++;

printf("Size of int data type: %d",ptr);

}

Output

Size of int data type: 4

1. Write a c program to find the size of double without using size of operator

#include<stdio.h>

int main()

{

double \*ptr = 0;

ptr++;

printf("Size of int data type: %d”,ptr);

}

Output

Size of int data type: 8l

1. Write a c program to find the size of structure without using size of operator

#include <stdio.h>

#include <stdlib.h>

typedef struct

{

char Name[12];

int Age;

float Weight;

int RollNumber;

}sStudentInfo;

int main(int argc, char \*argv[])

{

sStudentInfo RamInfo[2] = {0};

sStudentInfo \*psInfo = NULL;

int iSizeofStructure = 0;

psInfo = RamInfo;

iSizeofStructure = (char\*)(psInfo + 1) - (char\*)(psInfo);

printf("Size of structure = %d\n\n",iSizeofStructure);

}

Output

Size of structure = 24

1. Write a c program to find the size of union without using size of operator

#include<stdio.h>

union student{

int roll;

char name[100];

float marks;

};

int main()

{

union student \*ptr = 0;

ptr++;

printf("Size of the union student: %d",ptr);

}

Output

Size of the union student: 100

C PROGRAM WITH VERY LARGE NUMBERS

1. Write a c program to find the factorial of 100 or very large number

#include<stdio.h>

int main()

{

int a[200],n,counter,temp,i;

a[0]=1;

counter=0;

printf("Enter the number to Find Factorial: ");

scanf("%d",&n);

for(; n>=2; n--)

{

temp=0;

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

{

temp=(a[i]\*n)+temp;

a[i]=temp%10;

temp=temp/10;

}

while(temp>0)

{

a[++counter]=temp%10;

temp=temp/10;

}

}

for(i=counter; i>=0; i--)

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

return 0;

}

Output

Enter the number to Find Factorial: 120

6689502913449127057588118054090372586752746333138029810295671352301633557244962989366874165271984981308157637893214090552534408589408121859898481114389650005964960521256960000000000000000000000000000

1. Write a c program to multiply the two very large number (larger than long int)

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

#include<string.h>

#define MAX 10000

char \* multiply(char [],char[]);

int main(){

char a[MAX];

char b[MAX];

char \*c;

int la,lb;

int i;

printf("Enter the first number : ");

scanf("%s",a);

printf("Enter the second number : ");

scanf("%s",b);

printf("Multiplication of two numbers : ");

c = multiply(a,b);

printf("%s",c);

return 0;

}

char \* multiply(char a[],char b[]){

static char mul[MAX];

char c[MAX];

char temp[MAX];

int la,lb;

int i,j,k=0,x=0,y;

long int r=0;

long sum = 0;

la=strlen(a)-1;

lb=strlen(b)-1;

for(i=0;i<=la;i++){

a[i] = a[i] - 48;

}

for(i=0;i<=lb;i++){

b[i] = b[i] - 48;

}

for(i=lb;i>=0;i--){

r=0;

for(j=la;j>=0;j--){

temp[k++] = (b[i]\*a[j] + r)%10;

r = (b[i]\*a[j]+r)/10;

}

temp[k++] = r;

x++;

for(y = 0;y<x;y++){

temp[k++] = 0;

}

}

k=0;

r=0;

for(i=0;i<la+lb+2;i++){

sum =0;

y=0;

for(j=1;j<=lb+1;j++){

if(i <= la+j){

sum = sum + temp[y+i];

}

y += j + la + 1;

}

c[k++] = (sum+r) %10;

r = (sum+r)/10;

}

c[k] = r;

j=0;

for(i=k-1;i>=0;i--){

mul[j++]=c[i] + 48;

}

mul[j]='\0';

return mul;

}

Output

Enter the first number : 67878997987653354

Enter the second number : 788576

Multiplication of two numbers : 53527748717111731283904

1. Write a c program for the division of large number (larger than long int)

#include<stdio.h>

#include<string.h>

#define MAX 10000

int validate(char []);

char \* division(char[],long);

long int remainder;

int main(){

char dividend[MAX],\*quotient;

long int divisor;

printf("Enter dividend: ");

scanf("%s",dividend);

if(validate(dividend))

return 0;

printf("Enter divisor: ");

scanf("%ld",&divisor);

quotient = division(dividend,divisor);

while(\*quotient)

if(\*quotient ==48)

quotient++;

else

break;

printf("Quotient: %s / %ld = %s",dividend,divisor,quotient);

printf ("\nRemainder: %ld",remainder);

return 0;

}

int validate(char num[]){

int i=0;

while(num[i]){

if(num[i] < 48 || num[i]> 57){

printf("Invalid positive integer: %s",num);

return 1;

}

i++;

}

return 0;

}

char \* division(char dividend[],long divisor){

static char quotient[MAX];

long temp=0;

int i=0,j=0;

while(dividend[i]){

temp = temp\*10 + (dividend[i] -48);

if(temp<divisor){

quotient[j++] = 48;

}

else{

quotient[j++] = (temp / divisor) + 48;;

temp = temp % divisor;

}

i++;

}

quotient[j] = '\0';

remainder = temp;

return quotient;

}

Output

Enter dividend: 789

Enter divisor: 45567

Quotient: 789 / 45567 = 0

Remainder: 789

1. Write a c program for the modular division of large number

#include<stdio.h>

#include<string.h>

#define MAX 10000

int validate(char []);

char \* division(char[],long);

long int remainder;

int main(){

char dividend[MAX],\*quotient;

long int divisor;

printf("Enter dividend: ");

scanf("%s",dividend);

if(validate(dividend))

return 0;

printf("Enter divisor: ");

scanf("%ld",&divisor);

quotient = division(dividend,divisor);

while(\*quotient)

if(\*quotient ==48)

quotient++;

else

break;

printf("Quotient: %s / %ld = %s",dividend,divisor,quotient);

printf ("\nRemainder: %ld",remainder);

return 0;

}

int validate(char num[]){

int i=0;

while(num[i]){

if(num[i] < 48 || num[i]> 57){

printf("Invalid positive integer: %s",num);

return 1;

}

i++;

}

return 0;

}

char \* division(char dividend[],long divisor){

static char quotient[MAX];

long temp=0;

int i=0,j=0;

while(dividend[i]){

temp = temp\*10 + (dividend[i] -48);

if(temp<divisor){

quotient[j++] = 48;

}

else{

quotient[j++] = (temp %divisor) + 48;;

temp = temp % divisor;

}

i++;

}

quotient[j] = '\0';

remainder = temp;

return quotient;

}

1. Write a c program for the division of large number

#include<stdio.h>

#include<string.h>

#define MAX 10000

int validate(char []);

char \* division(char[],long);

long int remainder;

int main(){

char dividend[MAX],\*quotient;

long int divisor;

printf("Enter dividend: ");

scanf("%s",dividend);

if(validate(dividend))

return 0;

printf("Enter divisor: ");

scanf("%ld",&divisor);

quotient = division(dividend,divisor);

while(\*quotient)

if(\*quotient ==48)

quotient++;

else

break;

printf("Quotient: %s / %ld = %s",dividend,divisor,quotient);

printf ("\nRemainder: %ld",remainder);

return 0;

}

int validate(char num[]){

int i=0;

while(num[i]){

if(num[i] < 48 || num[i]> 57){

printf("Invalid positive integer: %s",num);

return 1;

}

i++;

}

return 0;

}

char \* division(char dividend[],long divisor){

static char quotient[MAX];

long temp=0;

int i=0,j=0;

while(dividend[i]){

temp = temp\*10 + (dividend[i] -48);

if(temp<divisor){

quotient[j++] = 48;

}

else{

quotient[j++] = (temp / divisor) + 48;;

temp = temp % divisor;

}

i++;

}

quotient[j] = '\0';

remainder = temp;

return quotient;

}

Output

Enter dividend: 789

Enter divisor: 45567

Quotient: 789 / 45567 = 0

Remainder: 789

1. Write a c program for the power of large number

#include <stdio.h>

int power1(int n1, int n2);

int main()

{

int base, power, result;

printf("Enter base number: ");

scanf("%d",&base);

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

scanf("%d",&power);

result = power1(base, power);

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

return 0;

}

int power1(int base, int power)

{

if (power != 0)

{

return (base\*power1(base, power-1));

}

else

{

return 1;

}

}

Output

Enter base number: 2

Enter power number(positive integer): 5

2^5 = 32

STRUCTURES

1. Write a c program to store information (name,roll and marks )of a student using structures

#include<stdio.h>

struct student

{

char name[40];

int rollno;

float totalmarks;

}f1;

void main()

{

printf("enter students name");

scanf("%c",f1.name);

printf("enter students roll no.");

scanf("%d",f1.rollno);

printf("enter students marks");

scanf("%f",f1.totalmarks);

}

Ouput

enter students name sushi

enter students roll no.2311

enter students marks 456.4

1. Write a c program to add two distances in inch-feet system using structures

#include <stdio.h>

struct distance

{

int inch;

int feet;

}f1,f2;

void main()

{

printf("enter the value of inches");

scanf("%d",&f1.inch);

printf("enter the value of feet");

scanf("%d",&f1.feet);

printf("enter the value of inches");

scanf("%d",&f2.inch);

printf("enter the value of feet");

scanf("%d",&f2.feet);

int inch1=f1.inch+f2.inch;

int feet1=f1.feet+f2.feet;

if(feet1>=100)

{

int a=feet1%100;

int b=(feet1-a)/100;

inch1=inch1+b;

printf("answer is %d.%d",inch1,a);

}

else

{

printf("answer is %d.%d",inch1,feet1);

}

}

Output

enter the value of inches23

enter the value of feet67

enter the value of inches234

enter the value of feet76

answer is 258.43

1. Write a c program to add two complex numbers

#include <stdio.h>

struct distance

{

int real;

int img;

}f1,f2;

void main()

{

printf("enter the value of real");

scanf("%d",&f1.real);

printf("enter the value of img");

scanf("%d",&f1.img);

printf("enter the value of real");

scanf("%d",&f2.real);

printf("enter the value of img");

scanf("%d",&f2.img);

int real1=f1.real+f2.real;

int img1=f1.img+f2.img;

printf("answer is %d+i%d",real1,img1);

}

Output

enter the value of real23

enter the value of img12

enter the value of real3

enter the value of img12

answer is 26+i24

1. Write a c program to figure whether system is little endian or big endian

#include <stdio.h>

int main ()

{

unsigned int x = 0x76543210;

char \*c = (char\*) &x;

printf ("\*c is: 0x%x\n", \*c);

if (\*c == 0x10)

{

printf ("Underlying architecture is little endian. \n");

}

else

{

printf ("Underlying architecture is big endian. \n");

}

return 0;

}

Output

\*c is: 0x10

Underlying architecture is little endian.

OTHER PROGRAMS

1. Write a c program to print hello world without using semicolon

#include<stdio.h>

int main()

{

if(printf("hello world"))

{

}

}

Output :

Hello world

1. Write a c program to solve the quadratic equation

#include<stdio.h>

#include<math.h>

void main()

{

float a,b,c;

printf("enter the values of a,b,c");

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

float x1=(-b+(sqrt((b\*b)-(4\*a\*c))))/2\*a;

float x2=(-b-(sqrt((b\*b)-(4\*a\*c))))/2\*a;

printf("the roots are %f and %f",x1,x2);

}

Output

enter the values of a,b,c1 0 -9

the roots are 3.000000 and -3.000000

1. Write a c program to check whether a number is a perfect number or not

#include<stdio.h>

#include<math.h>

void main()

{

int a;

printf("enter the value of a");

scanf("%d",&a);

int sum=0;

int i;

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

{

if(a%i==0)

{

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

sum=sum+i;

}

}

if(sum==a)

{

printf("perfect number");

}

else

{

printf("not a perfect number");

}

}

Output

enter the value of a8128

1

2

4

8

16

32

64

127

254

508

1016

2032

4064

perfect number

1. Write a c program to check whether the given number is Armstrong number or not

#include<stdio.h>

void main()

{

int n;

printf("enter the no. of elements in the number");

scanf("%d",&n);

int a[n];

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

{

printf("enter the value");

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

}

int sum;

int product;

/\*only for 3-digit numbers\*/

sum=(a[0]\*a[0]\*a[0])+(a[1]\*a[1]\*a[1])+(a[2]\*a[2]\*a[2]);

product=(a[0]\*100)+(a[1]\*10)+(a[2]\*1);

if(sum==product)

{

printf("armstrong number");

}

else

{

printf("not an armstrong number");

}

}

Output

enter the no. of elements in the number3

enter the value1

enter the value5

enter the value3

armstrong number

1. Write a c program to check given number is strong number or not

#include<stdio.h>

void main()

{

/\*applicable for 3 digits\*/

int a;

printf("enter a 3-digit number");

scanf("%d",&a);

int b=a%100;

int c=(a-b)/100;

int d=b%10;

int e=(b-d)/10;

printf("the digits are %d\t%d\t%d",c,e,d);

int sum1=1,sum2=1,sum3=1;

int i=1;

while(i<=c)

{

sum1=sum1\*i;

i=i+1;

}

int j=1;

while(j<=e)

{

sum2=sum2\*j;

j=j+1;

}

int k=1;

while(k<=d)

{

sum3=sum3\*k;

k=k+1;

}

int sum;

sum=sum1+sum2+sum3;

if(sum==a)

{

printf("\n");

printf("strong number ");

}

else

{

printf("not a strong number");

}

}

Output

enter a 3-digit number145

the digits are 1 4 5

strong number

1. Write a c program to find out generic root of any number

#include<stdio.h>

void main()

{

int a;

printf("enter a 3-digit number");

scanf("%d",&a);

int b=a%100;

int c=(a-b)/100;

int d=b%10;

int e=(b-d)/10;

printf("the digits are %d\t%d\t%d",c,e,d);

int sum=c+e+d;

if(sum>=10)

{

int f=sum;

int g=f%10;

int h=(f-g)/10;

int sum1=g+h;

printf("the generic root is %d",sum1);

}

else

{

printf("the generic root is %d",sum);

}

}

Output

enter a 3-digit number456

the digits are 4 5 6

the generic root is 6

1. Write a c program for atm transaction

#include<stdio.h>

void main()

{

int a=1360;

float c=1344552.30;

int b;

float a1;

printf("swipe a card");

printf("enter the pin");

scanf("%d",&b);

if(b==a)

{

printf("how much cash would you like to withdraw?");

scanf("%f",&a1);

printf("submitted");

printf("please collect the money and the bill");

sum=c-a1;

printf("the balance amout in the card is %f",sum);

printf("thank you");

}

else

{

printf("please try again-pin does not match");

}

}

Output

swipe a card

enter the pin1234

please try again-pin does not match

1. Write a c program to take password from the user

#include<stdio.h>

void main()

{

char a[10]="askfj2";

char b[10];

printf("enter the password");

scanf("%s",b);

int i,j;

int sum=0;

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

{

if(a[i]==b[i])

{

sum=sum+1;

}

}

int a1=6;

if(sum==a1)

{

printf("correct password");

}

else

{

printf("wrong password");

}

}

Output

enter the password sdasdsdf

wrong password

1. Write a sccanf function in c which accepts sentence from the user

#include<stdio.h>

#define MAX 500

int main()

{

char arr[MAX];

printf("Enter any sentence which can include spaces.\n");

printf("To exit press enter key.\n");

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

printf("You had entered: \n");

printf("%s",arr);

}

Output

Enter any sentence which can include spaces.

To exit press, enter key.

my name is sathyasri.

You had entered:

my name is sathyasri.

1. Write a scanf function which accepts a paragraph from the user

#include<stdio.h>

#define MAX 500

int main()

{

char arr[MAX];

printf("Enter any paragraph which can include spaces.\n");

printf("To exit press enter key.\n");

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

printf("You had entered: \n");

printf("%s",arr);

}

Output

Enter any paragraph which can include spaces.

To exit press, enter key.

Your city has everything but trees. The trees are essential for clean environment. Thus, clean environment needs green environment. Each citizen should plant a tree in the empty space before his house. He should look after it and let it grow up. The greenness will make your city worth living.

You had entered

Your city has everything but trees. The trees are essential for clean environment. Thus, clean environment needs green environment. Each citizen should plant a tree in the empty space before his house. He should look after it and let it grow up. The greenness will make your city worth living.

1. Write a c program to print all the prime numbers between 1 to 300

#include<stdio.h>

int main()

{

int num,div;

printf("2\n");

for(num=2;num<=300;num++)

{

for(div=2;div<num;div++)

{

if((num%div==0))

{

break;

}

else if(div==num-1)

{

printf ("\n%d",num);

}

}

}

return(0);

}

Output

2

3

5

7

11

13

17

19

23

29

31

37

41

43

47

53

59

61

67

71

73

79

83

89

97

101

103

107

109

113

127

131

137

139

149

151

157

163

167

173

179

181

191

193

197

199

211

223

227

229

233

239

241

251

257

263

269

271

277

281

283

293

1. Write a c program to pass a structure to a function

#include<stdio.h>

struct fraction

{

int numerator;

int denominator;

}f1;

void main()

{

printf("numerator");

scanf("%d",&f1.numerator);

printf("denominator");

scanf("%d",&f1.denominator);

printf("fraction is %d/%d",f1.numerator,f1.denominator);

}

Output

numerator22

denominator

3

fraction is 22/3

1. Write a c program without main function

#include<stdio.h>

#define start main

void start()

{

printf("Hello");

}

Output

Hello

1. Write a c program to make a simple calculator using switch case

#include<stdio.h>

void main()

{

int a,b;

printf("enter the value of a");

scanf("%d",&a);

printf("enter the value of b");

scanf("%d",&b);

char c;

printf("enter the operator");

scanf("%c",&c);

switch(c)

{

case'+':printf("addition %d\n",a+b);

break;

case'-':printf("subtraction %d\n",a-b);

break;

case'\*':printf("multiplication %d\n",a\*b);

break;

case'/':printf("division %d\n",a/b);

break;

case'%':printf("modulo-division %d\n",a%b);

break;

}

}

Output

enter the value of a6

enter the value of b2

enter the operator +

addition 8

1. Write a c program to display prime numbers between intervals using functions

#include<stdio.h>

void prime();

int main()

{

prime();

}

void prime()

{

int num,div;

printf("2\n");

for(num=2;num<=300;num++)

{

for(div=2;div<num;div++)

{

if((num%div==0))

{

break;

}

else if(div==num-1)

{

printf ("\n%d",num);

}

}

}

}

Ouput

2

3

5

7

11

13

17

19

23

29

31

37

41

43

47

53

59

61

67

71

73

79

83

89

97

101

103

107

109

113

127

131

137

139

149

151

157

163

167

173

179

181

191

193

197

199

211

223

227

229

233

239

241

251

257

263

269

271

277

281

283

293

1. Write a c program to check whether the given number can be expressed as the 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)

{

if (checkPrime(i) == 1)

{

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

{

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;

}

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: 23

23 cannot be expressed as the sum of two prime numbers.

1. Write a c 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

1. Write a c program which produces its own source code as its output

#include <stdio.h>

int main() {

FILE \*fp;

int c;

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

do {

c = getc(fp);

putchar(c);

}

while(c != EOF);

fclose(fp);

return 0;

}

Output

#include <stdio.h>

int main() {

FILE \*fp;

int c;

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

do {

c = getc(fp);

putchar(c);

}

while(c != EOF);

fclose(fp);

return 0;

}�

1. Write a c program to sort elements in lexicographical order

#include <stdio.h>

#include <string.h>

int main() {

char str[5][50], temp[50];

printf("Enter 5 words: ");

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

fgets(str[i], sizeof(str[i]), stdin);

}

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

for(int j = i+1; j < 5 ; ++j) {

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

strcpy(temp, str[i]);

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

strcpy(str[j], temp);

}

}

}

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

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

fputs(str[i], stdout);

}

return 0;

}

Output

Enter 5 words: hello

life

corpse

dark

rainbow

In the lexicographical order:

corpse

dark

hello

life

rainbow

1. Write a c program to calculate the difference between two time zones

#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);

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

1. Write a c program to generate multiplication table

#include<stdio.h>

void main()

{

int n,i,product;

printf("the value of n ");

scanf("%d",&n);

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

{

product=i\*n;

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

}

}

Output

the value of n 9

9X1=9

9X2=18

9X3=27

9X4=36

9X5=45

9X6=54

9X7=63

9X8=72

9X9=81

9X10=90

1. Write a c program to print ASCII values for all characters

#include <stdio.h>

int main()

{

int i;

for(i=0; i<=255; i++)

{

printf("ASCII value of character %c = %d\n", i, i);

}

}