SEM Practice programs

Join Halfs:

#include <stdio.h>

#include<math.h>

int joinHalfs(int m, int n)

{

int cm=0,cn=0,p1,p2,fh,sh,result,r1,r2,om,on;

om=m;

on=n;

while(m!=0)

{

cm=cm+1;

m=m/10;

}

while(n!=0)

{

cn=cn+1;

n=n/10;

}

p1=cm/2;

p2=cn/2;

r1=pow(10,p1);

r2=pow(10,p2);

fh=om/r1;

sh=on%r2;

result=(fh\*r2)+sh;

return result;

}

int main()

{

int m, n;

scanf("%d", &m);

scanf("%d", &n);

printf("%d\n", joinHalfs(m, n));

return 0;

}

Middle digit:

#include <stdio.h>

#include<math.h>

int middleDigit(int n)

{

int c=0,p,r,res,mid,on;

on=n;

while(n!=0)

{

c=c+1;

n=n/10;

}

r=c/2;

p=pow(10,r);

res=on/p ;

mid=res%10;

return mid;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Number of digits:

#include <stdio.h>

int numberOfDigits(int n)

{

int c=0;

if(n==0)

{

return 1;

}

while(n!=0)

{

c=c+1;

n=n/10;

}

return c;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Number of prime digits:

#include <stdio.h>

#define TRUE 1

#define FALSE 0

int isPrime(int digit)

{

switch (digit)

{

case 2:

case 3:

case 5:

case 7:

return TRUE;

default:

return FALSE;

}

}

int numberOfPrimeDigits(int n)

{

int res=0,digit;

while(n!=0)

{

digit=n%10;

res=res+isPrime(digit);

n=n/10;

}

return res;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Number of prime factors less than 10:

#include <stdio.h>

int numberOfPrimeFactors(int n)

{

int c=0;

if((n%2)==0)

{

c=c+1;

}

if((n%3)==0)

{

c=c+1;

}

if((n%5)==0)

{

c=c+1;

}

if((n%7)==0)

{

c=c+1;

}

return c;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Name and CGPA <5:

#include <stdio.h>

struct details

{

char name[50];

float cgpa;

}d[10];

int main()

{

int n,i;

scanf("%d",&n);

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

{

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

scanf("%f",&d[i].cgpa);

}

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

{

if(d[i].cgpa<5)

{

printf("%s\n",d[i].name);

printf("%.1f\n",d[i].cgpa);

}

}

}

Name and salary <10000:

#include <stdio.h>

struct details

{

char name[50];

int salary;

}d[10];

int main()

{

int n,i;

scanf("%d",&n);

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

{

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

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

}

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

{

if(d[i].salary<10000)

{

printf("%s\n",d[i].name);

printf("%d\n",d[i].salary);

}

}

}

Name and age >60:

#include <stdio.h>

struct details

{

char name[50];

int age;

}d[10];

int main()

{

int n,i;

scanf("%d",&n);

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

{

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

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

}

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

{

if(d[i].age>60)

{

printf("%s\n",d[i].name);

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

}

}

}

B and L alternative:

#include<stdio.h>

int main()

{

char ch,toBePrinted;

int i,j,r,c;

//printf("Please enter the chair:\n");

scanf("%c",&ch);

//printf("Please enter the number of rows:\n");

scanf("%d",&r);

//printf("Please enter the number of cols:\n");

scanf("%d",&c);

toBePrinted = ch;

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

{

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

{

printf("%c ",toBePrinted);

if(toBePrinted == 'B')

{

toBePrinted = 'L';

}

else if(toBePrinted == 'L')

{

toBePrinted = 'B';

}

}

printf("\n");

}

}

Position of max element in a matrix:

#include<stdio.h>

int main()

{

float a[10][10],max;

int i,j,rows,cols;

scanf("%d%d",&rows,&cols);

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

{

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

{

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

}

}

max=a[0][0];

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

{

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

{

if(a[i][j]>max)

{

max = a[i][j];

}

}

}

//printf("%.2f\n",max);

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

{

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

{

if(a[i][j]==max)

{

//printf("Match found\n");

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

}

}

}

}

Min element in a column:

#include<stdio.h>

int main()

{

int a[5][5],i,j,rows,cols,c,min;

scanf("%d%d%d",&rows,&cols,&c);

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

{

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

{

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

}

}

min=a[0][c];

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

{

if(a[i][c]<min)

{

min = a[i][c];

}

}

printf("%d",min);

}

Max element in a column:

#include<stdio.h>

int main()

{

int a[5][5],i,j,rows,cols,c,max;

scanf("%d%d%d",&rows,&cols,&c);

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

{

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

{

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

}

}

max=a[0][c];

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

{

if(a[i][c]>max)

{

max = a[i][c];

}

}

printf("%d",max);

}

Swap i th column with j th column:

#include<stdio.h>

int main()

{

int a[10][10],i,j,rows,cols,c1,c2,temp=0;

scanf("%d%d%d%d",&rows,&cols,&c1,&c2);

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

{

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

{

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

}

}

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

{

temp=a[i][c1];

//printf("Value of temp:%d\n",temp);

a[i][c1]=a[i][c2];

a[i][c2] = temp;

}

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

{

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

{

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

}

printf("\n");

}

}

Factorial of a number:

#include<stdio.h>

int fact(int);

int main()

{

int n,f=1,i;

scanf("%d",&n);

if(n==0)

{

printf("1");

}

else if(n<0)

{

printf("Invalid Input");

}

else

{

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

{

f=f\*i;

}

printf("%d",f);

}

}

Palindrome or not

#include <stdio.h>

int main()

{

int n, reversed = 0, remainder, original;

scanf("%d", &n);

original = n;

while (n != 0) {

remainder = n % 10;

reversed = reversed \* 10 + remainder;

n /= 10;

}

if(original<0)

{

printf("Invalid Input");

}

else if (original == reversed)

{

printf("Palindrome");

}

else

{

printf("Not a palindrome");

}

return 0;

}

Perfect number:

#include<stdio.h>

void perfect(int);

int main()

{

int num;

scanf("%d",&num);

perfect(num);

}

void perfect(int num)

{

int i,result=0;

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

{

if(num%i==0)

{

result=result+i;

}

}

if(num<=0)

{

printf("Invalid Input");

}

else if(result==num)

{

printf("Perfect Number");

}

else

{

printf("Not a perfect number");

}

}

Print \* pattern:

#include<stdio.h>

int main()

{

int rows,i,j;

scanf("%d",&rows);

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

{

printf("\n");

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

{

printf("\* ");

}

printf("\n");

}

return 0;

}

Print ABC pattern:

#include<stdio.h>

int main()

{

int rows,i,j,n=65;

scanf("%d",&rows);

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

{

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

{

printf("%c ",n);

}

++n;

printf("\n");

}

return 0;

}

Print 123 pattern:

#include<stdio.h>

int main()

{

int rows,i,j,n=1;

scanf("%d",&rows);

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

{

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

{

printf("%d ",n);

++n;

}

printf("\n");

}

return 0;

}

Prime number:

#include <stdio.h>

void prime(int);

int main()

{

int num;

scanf("%d",&num);

prime(num);

}

void prime(int num)

{

int i,count=0;

for(i=2;i<= num-1;i++)

{

if (num%i==0)

{

count++;

}

}

if(count==0)

{

printf("Prime number");

}

else

{

printf("Not a prime number");

}

}

GCD of two numbers:

#include <stdio.h>

int gcd(int,int);

int main()

{

int a,b;

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

printf("%d",gcd(a,b));

}

int gcd(int a, int b)

{

if (b != 0)

{

return gcd(b, a % b);

}

else

{

return a;

}

}

Count of a character in a string:

#include <string.h>

#include <stdio.h>

int main()

{

char s[1000],c,ch;

int i,count=0;

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

scanf(" %c", &c);

if (c>96 && c<123)

{

ch=c-32;

}

else if(c>=65&&c<91)

{

ch=c+32;

}

for(i=0;i<=strlen(s);i++)

{

if(s[i]==c || s[i]==ch)

{

count++;

}

}

printf("%d",count);

return 0;

}

No. of words in a string:

#include <stdio.h>

#include <string.h>

int main()

{

char s[200];

int count = 0, i;

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

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

{

if (s[i] == ' ' && s[i+1] != ' ')

count++;

}

printf("%d\n", count + 1);

}

Reverse last three significant digits:

#include <stdio.h>

int reverseLast3(int n)

{

int on,l1,l2,l3,h,fh,sh,nn;

on=n;

l1=n%10;

l2=(n/10)%10;

l3=(n/100)%10;

h=on/1000;

fh=h\*1000;

if(l3==0)

{

if(l2==0)

{

sh=l1;

}

else

{

sh=(l2\*1)+(l1\*10);

}

}

else

{

sh=(l3\*1)+(l2\*10)+(l1\*100);

}

nn=fh+sh;

return nn;

}

int main() {

int n;

scanf("%d", &n);

printf("%d\n", reverseLast3(n));

return 0;

}

Same sign or not:

#include <stdio.h>

#define TRUE 1

#define FALSE 0

// return TRUE or FALSE

int sameSign(int m, int n)

{

if(((m>0)&&(n>0))||((m<0)&&(n<0)))

{

return 1;

}

else

return 0;

}

int main() {

int m, n, result;

scanf("%d", &m);

scanf("%d", &n);

result = sameSign(m, n);

printf("%s\n", result ? "TRUE" : "FALSE");

return 0;

}

Sum of cubes of LSD and MSD:

#include <stdio.h>

#include<math.h>

int sumOfCubesOfLSDAndMSD(int n)

{

int ld,fd,result;

ld = n %10;

while(n> 10)

{

n = n/10;

}

fd = n;

result= pow(ld,3) + pow(fd,3);

return result;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Sum of LSD and MSD:

#include <stdio.h>

int sumOfLSDAndMSD(int n) {

// Write your code here

int ld,fd,sum;

ld = n %10;

while(n> 10)

{

n = n/10;

}

fd = n;

sum= ld + fd;

return sum;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Sum of MSD of two numbers:

#include <stdio.h>

int sumOfMSD(int num1, int num2) {

// Write your code here

int fd,fd2,sum;

while(num1> 10)

{

num1 = num1/10;

}

fd = num1;

while(num2> 10)

{

num2 = num2/10;

}

fd2 = num2;

sum= fd + fd2;

return sum;

}

int main() {

int m, n;

scanf("%d", &m);

scanf("%d", &n);

printf("%d\n", sumOfMSD(m, n));

return 0;

}

Sum of squares of LSD and MSD:

#include <stdio.h>

// Explain your logic here as comments

//find last digit

//find first digit

// square the digits and add

// return value

int sumOfSquareOfLSDAndMSD(int n) {

// Write your code here

int ld,fd,squares;

ld = n %10;

while(n> 10)

{

n = n/10;

}

fd = n;

squares= (ld \* ld) + (fd \* fd);

return squares;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Sum of squares of MSD of three numbers:

#include <stdio.h>

// Explain your logic here as comments

//

//

//

//

//

// return most significant digit of n

int msd(int n) {

while (n >= 10) {

n /= 10;

}

return n;

}

int sumOfSquaresOfMSDs(int num1, int num2, int num3) {

// Write your code here

// use the above function to compute this result

int squares;

int msd1= msd(num1);

int msd2= msd(num2);

int msd3= msd(num3);

squares=(msd1\*msd1)+(msd2\*msd2)+(msd3\*msd3);

return squares;

}

int main() {

int l, m, n;

scanf("%d", &l);

scanf("%d", &m);

scanf("%d", &n);

printf("%d\n", sumOfSquaresOfMSDs(l, m, n));

return 0;

}

Binary palindrome:

#include <stdio.h>

#define TRUE 1

#define FALSE 0

// if n is palindrome in binary format

// return TRUE

// otherwise return FALSE

int palindromeInBinary(int n)

{

int r,sum=0,temp=0;

while(n>0)

{

r=n%2;

sum=(sum\*10)+r;

n=n/2;

}

n=sum;

while(n>0)

{

r=n%10;

temp=(temp\*10)+r;

n=n/10;

}

if(temp==sum)

return TRUE;

else

return FALSE;

}

int main() {

int n, result;

scanf("%d", &n);

result = palindromeInBinary(n);

printf("%s\n", result ? "TRUE" : "FALSE");

return 0;

}

Sum of factorial of digits:

#include <stdio.h>

int fact(int digit)

{

if((digit==0)||(digit==1))

{

return 1;

}

else

{

return fact(digit-1)\*digit;

}

}

int sumOfFactorialOfDigits(int n)

{

int f=0,digit;

if(n==0)

{

return 1;

}

while(n!=0)

{

digit=n%10;

n=n/10;

f=f+fact(digit);

}

return f;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Repeated sum of digits:

#include <stdio.h>

int repeatedSumOfDigits(int n) {

int sum=0;

while(n>0 || sum>9)

{

if(n==0)

{

n=sum;

sum=0;

}

sum+=n%10;

n/=10;

}

return sum;

}

int main() {

int n;

scanf("%d", &n);

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

return 0;

}

Sum of middle digits:

#include <stdio.h>

int sumOfMiddleDigits(int num1, int num2)

{

int i,t1,t2,c1=0,c2=0,sum=0;

t1=num1;

t2=num2;

while(num1>0)

{

num1/=10;

c1++;

}

num1=t1;

c1=c1/2;

while(num2>0)

{

num2/=10;

c2++;

}

num2=t2;

c2=c2/2;

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

{

num1=num1/10;

}

num1=num1%10;

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

{

num2=num2/10;

}

num2=num2%10;

sum=num1+num2;

return sum;

}

int main()

{

int m, n;

scanf("%d", &m);

scanf("%d", &n);

printf("%d\n", sumOfMiddleDigits(m, n));

return 0;

}

Power raised to the position of the digit:

#include <stdio.h>

#include <math.h>

int main(){

long num=0, temp=0, digit=0, sum=0;

int countDigits=0;

scanf("%ld", &num);

temp=num;

while(temp>0){

countDigits = countDigits + 1;

temp = temp/10;

}

while(num>0){

digit = num%10;

sum += pow(digit, countDigits--);

num/=10;

}

printf("%ld", sum);

}

Remove common digits:

#include <stdio.h>

int removeCommonDigits(int m, int n)

{

int result = 0;

int unique = 0;

int l1 = 0, l2 = 0;

int i, j;

int temp = m;

while (temp != 0)

{

l1++;

temp = temp / 10;

}

temp = n;

while (temp != 0)

{

l2++;

temp = temp / 10;

}

for (i = 0; i < l1; i++)

{

int flag = 0;

int num1 = m % 10;

temp = n;

for (j = 0; j < l2; j++)

{

int num2 = temp % 10;

if (num1 == num2)

{

flag = 1;

break;

}

temp = temp / 10;

}

m = m / 10;

if (flag == 0)

{

unique = (unique \* 10) + num1;

}

}

while (unique != 0)

{

int r = unique % 10;

result = (result \* 10) + r;

unique = unique / 10;

}

return result;

}

int main()

{

int m, n;

scanf("%d", &m);

scanf("%d", &n);

printf("%d\n", removeCommonDigits(m, n));

return 0;

}

Remove repeated digits:

#include<stdio.h>

#include<string.h>

int main()

{

long n;

scanf("%ld",&n);

long res=0;

long set=0;

long rev=0;

while(n>0)

{

rev=(rev\*10)+(n%10);

n=n/10;

}

while(rev>0)

{

long mod=rev%10;

long mask=1<<mod;

if((set&mask)==0)

{

res=(res\*10)+mod;

set|=mask;

}

rev=rev/10;

}

printf("%ld",res);

return 0;

}