

/*PROGRAMA QUE EVALÚA LA ECUACIÓN CUADRÁTICA. PRUEBA SI EXISTE LA ECUACIÓN CUADRÁTICA, CALCULA RAÍCES REALES O RAÍCES COMPLEJAS */

```
#include<math.h>
#include<stdio.h>
#include<conio.h>

void error(void);
void imprime_real(float,float);
int cuadratica(float *,float *,float);
void imprime_complejo(float,float);
void datos_cuadratica(float *,float *,float *);

void main(void)
{
float a,b,c;
clrscr();
datos_cuadratica(&a,&b,&c);
switch(cuadratica(&a,&b,c)){
case 0 :error();          break;
case 1 :imprime_real(a,b); break;
case -1:imprime_complejo(a,b);
}
getch();
}

int cuadratica(float *a,float *b,float c)
{
float d;
if(*a==0)return(0);
d>(*b)*(*b)-4.0*(*a)*c;
c=1.0/(2.0*(*a));
if(d>=0.0){
*a=(-*b)+sqrt(d)*c;
*b=(-*b)-sqrt(d)*c;
return(1);
}
*a=-(*b)*c;
*b=sqrt(fabs(d))*c;
return(-1);
}

void error(void)
{ puts("\nNo existe la ecuación cuadrática."); }
```

```
void imprime_real(float x1,float x2)
{ printf("\nRAICES REALES.\n\nx1=%f\nx2=%f",x1,x2); }
```

```
void imprime_complejo(float r,float i)
{ printf("\nRAICES COMPLEJAS.\n\nx1=%.2f%.2fi\nx2=%.2f%.2fi",r,i,r,-i); }
```

```
void datos_cuadratica(float *a,float *b,float *c)
{
puts("Da los valores de a,b,c: ");
scanf("%f%f%f",a,b,c);
}
```

/*EJEMPLO DE UNA FUNCIÓN QUE GENERA N NÚMEROS ALEATORIOS EN UN RANGO DE LINF A LSUP Y ALMACENA ESTOS NÚMEROS EN UN VECTOR*/

```
#include<time.h>
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
```

```
void muestra_vector(int,int *);
int datos_correctos(int,int,int);
void aleatorios(int,int,int,int *);
void datos_aleatorio(int *,int *,int *);
```

```
void main(void)
{
int num[100],linf,lsup,n;
clrscr();
srand(time(NULL));
datos_aleatorio(&n,&linf,&lsup);
if(datos_correctos(n,linf,lsup))aleatorios(n,linf,lsup,num);
else{
puts("\nError de lógica en los datos");
exit(0);
}
puts("\nLos números generados son:");
muestra_vector(n,num);
}
```

```
void aleatorios(int n,int linf,int lsup,int *num)
{
    int i;
    for(i=0;i<n;i++)num[i]=linf+rand()%(lsup-linf+1);
}
```

```
void datos_aleatorio(int *n,int *linf,int *lsup)
{
    printf("Cuantos números quieres obtener. (Máximo 100): ");
    scanf("%d",n);
    printf("\nDa el limite inferior: ");
    scanf("%d",linf);
    printf("\nDa el limite superior: ");
    scanf("%d",lsup);
}
```

```
int datos_correctos(int n,int linf,int lsup)
{
    if(n<2 || (linf>lsup) || linf==lsup)return(0);
    return(1);
}
```

```
void muestra_vector(int n,int *num)
{
    int i;
    for(i=0;i<n;i++)printf("%8d",num[i]);
}
```

/*MÁXIMO COMÚN DIVISOR DE DOS NÚMEROS Y DE N NÚMEROS */

```
#include<stdio.h>
#include<conio.h>
#include<limits.h>
```

```
int mcd(int,int);
char menu_mcd(void);
int mcdn(int,int *,int); /*tercer argumento es el minimo encontrado*/
void datos_mcd(int *,int *);
int datos_mcdn(int *,int *); /*retorna el minimo encontrado*/
```

```
char *mensaje[]={ "MENU PARA EL MAXIMO COMUN DIVISOR",
                  "DATOS PARA EL MCD DE DOS NUMEROS\n\n",
                  "\nDATOS PARA EL MCD DE n NUMEROS\n\n"};
```

```

void main(void)
{
int num1,num2,num[80],n,min;
for(;;){
switch(menu_mcd()){
case '1':datos_mcd(&num1,&num2);
printf("\nEl MCD=%d",mcd(num1,num2));
break;
case '2':min=datos_mcdn(num,&n);
printf("\nEl MCD = %d",mcdn(min,num,n));
break;
case '3':exit(0);
}
getch();
}
}

```

```

char menu_mcd(void)
{
clrscr();
puts(mensaje[0]);
puts("\n1. Máximo común divisor de dos números.");
puts("\n2. Máximo común divisor de n números.");
puts("\n3. Terminar.");
printf("\nElige una opción: ");
return (getch());
}

```

```

void datos_mcd(int *num2,int *num1)
{
clrscr();
puts(mensaje[1]);
printf("\nIntroduce el primer número: ");
scanf("%d",num1);
printf("\nIntroduce el segundo número: ");
scanf("%d",num2);
}

```

```

int datos_mcdn(int *num,int *n)
{
int i,min=INT_MAX;
clrscr();
puts(mensaje[2]);
printf("\nCuantos son los enteros para encontrar su MCD: ");
scanf("%d",n);
for(i=1;i<=*n;i++){
printf("\nIntroduce el número %d: ",i);

```

```

scanf("%d",&num[i]);
if(num[i]<min)min=num[i];
}
return (min);
}

```

```

int mcd(int num,int MCD)
{
int min;
for(min=MCD;num%MCD || min%MCD;MCD--);
return (MCD);
}

```

```

int mcdn(int min,int *num,int n)
{
int i,MCD;
for(MCD=min,i=1;i<=n;i++)          /*función reutilizada*/
    MCD=mcd(num[i],MCD);          /*Compara cada número con el menor*/
return(MCD);
}

```

/*PROGRAMA QUE SUMA DOS MATRICES*/

```

#include<conio.h>
#include<stdlib.h>
#include<stdio.h>

```

```

#define C 5
#define R 5

```

```

void tam_mat(int *,int *);
void imprime_summat(int(*)[C],int,int);
void captura_mat(int(*)[C],int,int,char *);
void suma_mat(int(*)[C],int(*)[C],int,int);

```

```

void main(void)
{
int a[R][C],b[R][C],m,n;
clrscr();
tam_mat(&m,&n);
captura_mat(a,m,n,"\n captura la primer matriz:");
captura_mat(b,m,n,"\n captura la segunda matriz:");
suma_mat(a,b,m,n);
puts("\n el resultado es:");
imprime_summat(a,m,n);
getch(); }

```

```
void captura_mat(int(*x)[C],int m,int n,char *msg)
{
    int i,j;
    puts(msg);
    for(i=0;i<m;i++)
        for(j=0;j<n;j++)
            {
                printf("mat[%d][%d]=",i,j);
                scanf("%d",&x[i][j]);
            }
}
```

```
void tam_mat(int *m,int*n)
{
    puts("\n Da el numero de renglones:");
    scanf("%d",m);
    puts("\n Da el numero de columnas:");
    scanf("%d",n);
}
```

```
void suma_mat(int(*x)[C],int(*y)[C],int m,int n)
{
    int i,j;
    for(i=0;i<m;i++)
        for(j=0;j<n;j++)
            x[i][j]=x[i][j]+y[i][j];
}
```

```
void imprime_summat(int(*x)[C],int m,int n)
{
    int i,j;
    for(i=0;i<m;i++)
        {
            for(j=0;j<n;j++)
                printf("[%d][%d]=%d\t",i,j,x[i][j]);
            printf("\n");
        }
}
```