Grupo: 2BV3
Practica No: 5°
Fecha de realización: 20-Marzo-07
Fecha de entrega: 27-Marzo-07

ALGORITMO:

Algoritmo de eliminación gaussiana con pivoteo parcial:

Para resolver el sistema lineal:

$$E_1 = a_{11}x_1 + a_{12}x_2 + ...a_{1n}x_{1n} = a_{1,n+1}$$
$$E_2 = a_{21}x_1 + a_{22}x_2 + ...a_{2n}x_{2n} = a_{2,n+1}$$
$$.$$
$$.$$
$$.$$
$$E_n = a_{n1}x_1 + a_{n2}x_2 + ...a_{nn}x_{nn} = a_{n,n+1}$$

Entrada
Número de incógnitas y ecuaciones n; matriz aumentada $A=(a_{ij})$ donde $1\leq i\leq n$ y $1\leq j\leq n+1$.

Salida
Solución $x_1,\ldots,xn$ o mensaje de que el sistema lineal no tiene una solución única.

Paso 1 Para i=1,…,n tome NROW(i)=i. (Inicializar apuntador de renglón)

Paso 2 Para i=1,n-1 haga pasos 3-6 (Proceso de eliminación)

Paso 3 Sea p el entero mas pequeño con $i\leq p\leq n$ y
|a(NROW(p),i| =máx $_{i\leq p\leq n}$|a(NROW(j),i)|
Notacion::a(NROW(i),j) ≡a_{NROWij})

Paso 4 Si a(NROW(p),i)=0 entonces SALIDA (No existe una solución única´);
PARAR

Paso 5 Si NROW(i)≠NROW(p) entonces tome NCOPY = NROW(i)
NROW(i) = NROW(P)
NROW(p) = NCOPY

(Intercambio de renglones simulado.)

Paso 6 Para j=i+1,...,n haga los pasos 7 y 8

Paso 7 Tome m(NROW(j),i) = a(NROW(i),i)

Paso 8 Realice ($E_{NROW((j)}$ – m(NROW(j),i) * $E_{NROW(i)}$ => ($E_{NROW(j)}$)

Paso 9 Si a(NROW(n),n) = 0 entonces SALIDA (No existe una solución única´);
PARAR

Paso 10 Tome x_n= a(NROW(n),n+1)/a(NROW(n),n).
(Comience la sustitución hacia atrás.)

Paso 11 Para i=n-1,…,1

$$\text{Tome } x_i = \frac{a(NROW(i), n+1) - \sum_{j=i+1}^{n} a(NROW(i), j) * x_j}{a(NROW(i), i)}$$

Paso 12 SALIDA (x_1,…,xn);(Procedimiento terminado exitosamente.)
PARAR.

CÓDIGO FUENTE:

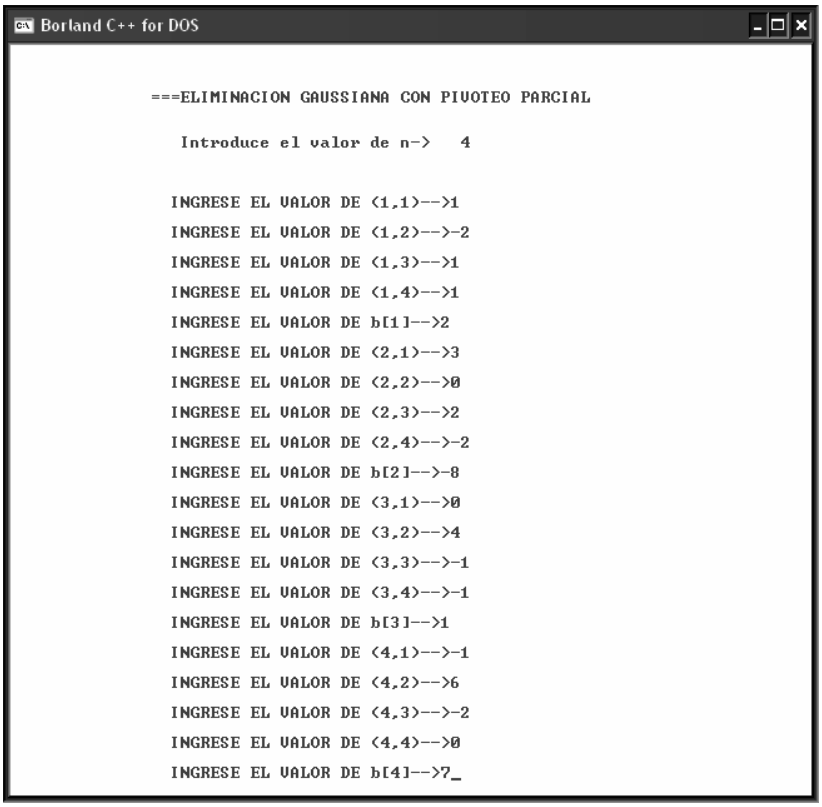
```
#include<stdlib.h>
#include<dos.h>
#include<stdio.h>
#include <conio.h>
#include<math.h>

void sustitucion(int);
void diagonal(int);
void rellenar(int);
void soluciones (int);
float x[20],a[20][20],b[30],nrow[20],d[20];

void main()
{
int n,o,t;
clrscr();
gotoxy (15,4);printf ("===ELIMINACION GAUSSIANA CON PIVOTEO PARCIAL");
gotoxy(18,7); printf ("Introduce el valor de n->\t");flushall();
scanf ("%d", & n);
rellenar(n);
diagonal(n);
sustitucion(n);
soluciones(n);
t=20-(n*2);
for (o=1;o<=n;o++)
   {
   gotoxy (t,22);printf ("x[%d]->%.2f",o,x[o]);
   t=t+12;
   }
getch();
}

void diagonal(int n)
{
int i,j,o,p,max,ncopy,c,g,r,h,l;//max guarda el maximo y
double factor;              //d me mantienes renglones ordenados y con nrow me guarda a cual apunta
for (i=1;i<=n;i++)
  {
  nrow[i]=i;
  d[i]=i;
```

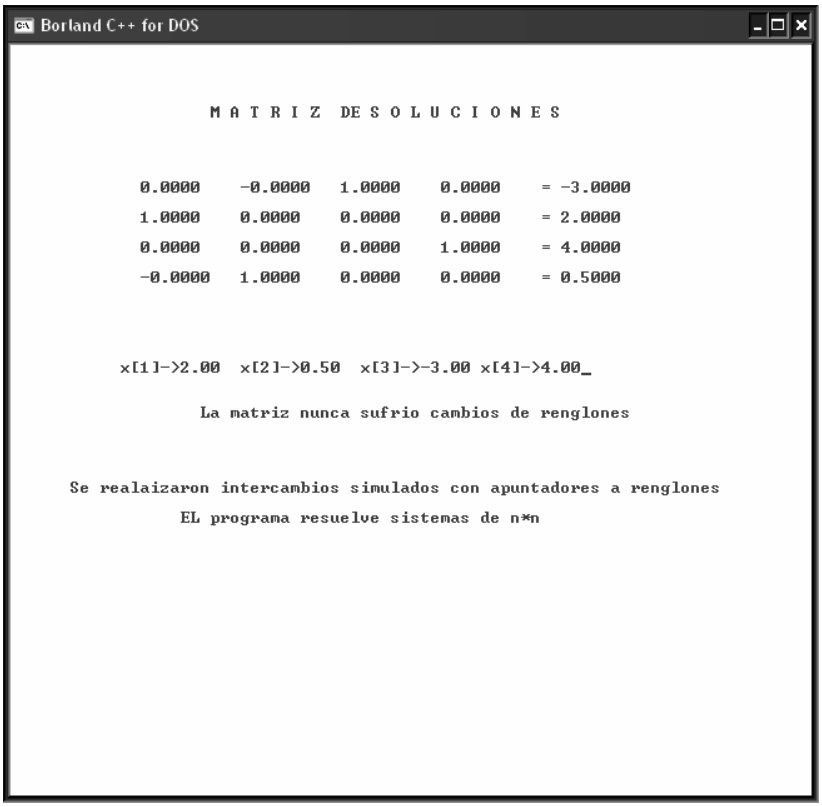
```
  }
for (i=1;i<=n-1;i++) //i es el renglon para el pivote
   {
   max=0;
   c=i;
   for (p=1;p<=n;p++)
     {
     if (nrow[p]>i)
            {
            if (sqrt(pow((a[nrow[p]][i]),2))>max)
              {
              max=sqrt(pow((a[nrow[p]][i]),2));
              c=p;
              }
            }
     }
   if (max==0)
     {
     clrscr();
     gotoxy (10,10); printf ("\n\tEl sistema no tiene solucion unica");
     getch ();
     exit(n);
     }
   if (c!=i)
     {
     for (p=1;p<=n;p++)
            {
            if (nrow[i]==d[p])
              g=p;
            if (nrow[c]==d[p])
              h=p;
            }
     ncopy=d[g];
     d[g]=d[h];
     d[h]=ncopy;
     }
   for (j=i+1;j<=n;j++) //j representa los renglones abajo del pivote
     {
     for (r=1;r<=n;r++)
             {
             if (d[r]==j)
               g=r;
             if (d[r]==nrow[i])
               h=r;
             }
     factor=a[g][i]/a[h][i];
     a[g][i]=(a[g][i]-(factor*(a[h][i])));
     for (o=i+1;o<=n;o++)
             {
             a[g][o]=a[g][o]-factor*a[h][o]; //cambio de la fila por la multimpliacion por factor y resta con
renglon i
             }
     b[g]=b[g]-(factor*(b[h])); // se cambia el valor de los resultados tambien
     }
   }
}
```

```
void sustitucion(int n)
{
double sum;
int i,j,g,u;
for (i=1;i<=n;i++)
   {
   if (d[i]==n)
     g=i;
   }
if (a[g][n]==0)
  {
  clrscr();
  gotoxy (10,10); printf ("El sistema no tiene solucion unica");
  getch();
  exit;
  }
x[n]=b[g]/a[g][n];
a[g][n]=1;
for (i=n-1;i>=1;i--)
   {
   for (u=1;u<=n;u++)
      {
      if (d[u]==i)
            g=u;
      }
   sum=0;
   for (j=i+1;j<=n;j++)
      {
      sum=sum+(a[g][j]*x[j]);
      a[g][j]=0;
      }
   x[i]=(b[g]-sum)/a[g][i];
   a[g][i]=1;
   }
}

void soluciones (int n)
{
int y,r,m,l,u;
clrscr();
gotoxy(20,5);printf(" M A T R I Z  DE S O L U C I O N E S ");
gotoxy(20,25); printf("La matriz nunca sufrio cambios de renglones");
gotoxy(7,30);printf ("Se realaizaron intercambios simulados con apuntadores a renglones");
gotoxy(18,32);printf ("EL programa resuelve sistemas de n*n");
y=10;
u=1;
for (m=1;m<=n;m++)
   {
   r=22-(2*n);
   for (l=1;l<=n;l++)
      {
      gotoxy(r,y);printf("%.4f",a[m][l]);
      r=r+10;
      }
   gotoxy(r,y);printf("= %.4f", x[d[u]]);
```

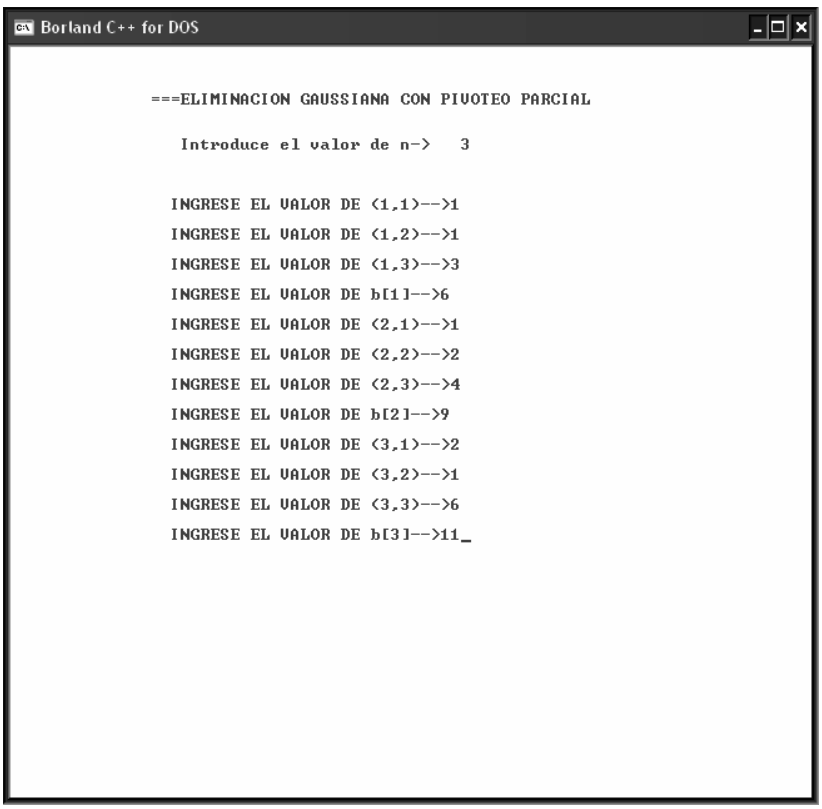
```
  y=y+2;
  u++;
  }
}

void rellenar(int n)
{
int i,j;
printf("\n\n");
for (i=1;i<=n;i++)
  {
  for (j=1;j<=n;j++)
    {
    printf("\n\t\tINGRESE EL VALOR DE (%d,%d)-->",i,j);
    scanf("%f",&a[i][j]);
    }
  printf("\n\t\tINGRESE EL VALOR DE b[%d]-->",i);
  scanf("%f",&b[i]);
  }
}
```

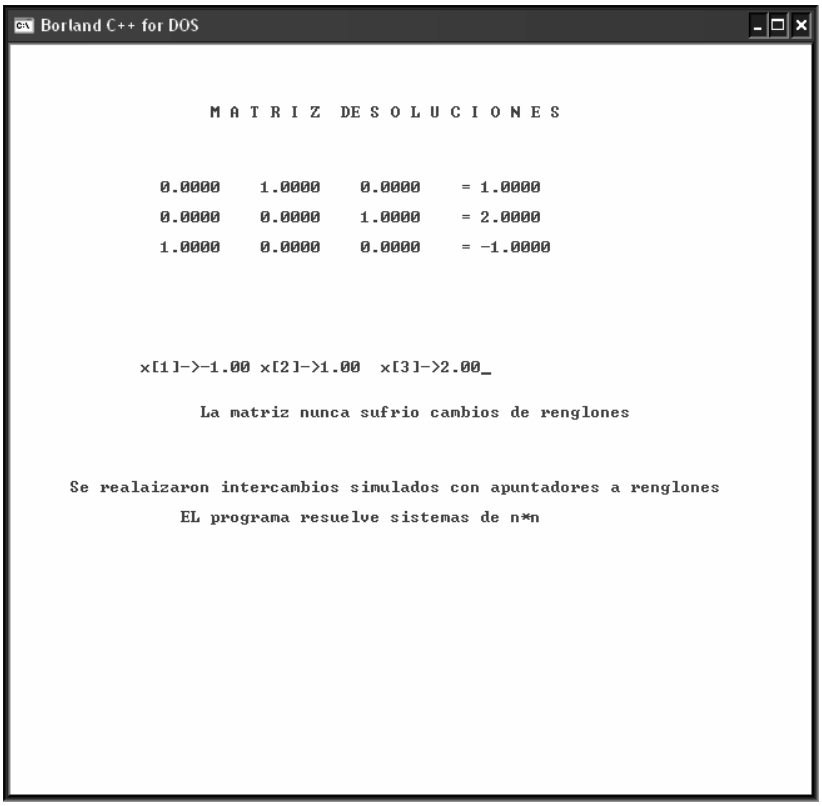
CORRIDA:




Borland C++ for DOS
M A T R I Z DE S O L U C I O N E S
0.0000 -0.0000 1.0000 0.0000 = -3.0000
1.0000 0.0000 0.0000 0.0000 = 2.0000
0.0000 0.0000 0.0000 1.0000 = 4.0000
-0.0000 1.0000 0.0000 0.0000 = 0.5000
x[1]->2.00 x[2]->0.50 x[3]->-3.00 x[4]->4.00_
La matriz nunca sufrio cambios de renglones
Se realaizaron intercambios simulados con apuntadores a renglones
EL programa resuelve sistemas de n*n

b)


Borland C++ for DOS
===ELIMINACION GAUSSIANA CON PIVOTEO PARCIAL
Introduce el valor de n-> 3
INGRESE EL VALOR DE (1,1)-->1
INGRESE EL VALOR DE (1,2)-->1
INGRESE EL VALOR DE (1,3)-->3
INGRESE EL VALOR DE b[1]-->6
INGRESE EL VALOR DE (2,1)-->1
INGRESE EL VALOR DE (2,2)-->2
INGRESE EL VALOR DE (2,3)-->4
INGRESE EL VALOR DE b[2]-->9
INGRESE EL VALOR DE (3,1)-->2
INGRESE EL VALOR DE (3,2)-->1
INGRESE EL VALOR DE (3,3)-->6
INGRESE EL VALOR DE b[3]-->11_


Borland C++ for DOS
M A T R I Z DE S O L U C I O N E S
0.0000 1.0000 0.0000 = 1.0000
0.0000 0.0000 1.0000 = 2.0000
1.0000 0.0000 0.0000 = -1.0000
x[1]->-1.00 x[2]->1.00 x[3]->2.00_
La matriz nunca sufrio cambios de renglones
Se realaizaron intercambios simulados con apuntadores a renglones
EL programa resuelve sistemas de n*n