

En esta práctica se pide la implementación de un conjunto de funciones y un programa principal para el manejo de matrices de 8x8 enteros. En concreto, las funciones a implementar son:

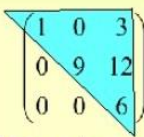
- rellenar una matriz pasada como parámetro con enteros en el rango [-20,20] usando la función aleatorio de la práctica anterior.
- imprimir una matriz por pantalla.
- multiplicar dos matrices que se pasan por parámetro dejando el resultado en la primera.
- convertir una matriz en triangular superior.
- rellenar una matriz como triangular inferior con valores en el rango [-20,20].
- rellenar una matriz identidad.

Ejemplos de Matriz Identidad / Unidad

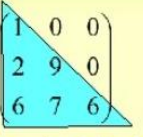
$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Matriz triangular

Matriz cuadrada en la que los elementos por debajo o por encima de la diagonal principal son ceros.



Triangular superior



Triangular inferior

El programa principal deberá definir una constante para el tamaño de la matriz y realizar las siguientes operaciones:

1. **Rellenar la matriz 1.**
2. Imprimir la matriz 1.
3. **Rellenar la matriz 2.**
4. Imprimir la matriz 2.
5. **Multiplicar la matriz 1 por la matriz 2.**
6. Imprimir la matriz 1.
7. **Convertir la matriz 1 en triangular superior.**
8. Imprimir la matriz 1.
9. **Convertir la matriz 2 en triangular superior.**
10. Imprimir la matriz 2.
11. **Multiplicar la matriz 2 por la matriz 1.**
12. Imprimir la matriz 2.
13. **Rellenar como triangular inferior la matriz 1.**
14. Imprimir la matriz 1.
15. **Multiplicar la matriz 2 por la matriz 1.**
16. Imprimir la matriz 2.
17. **Rellenar como identidad la matriz 2.**
18. Imprimir la matriz 2.
19. **Multiplicar la matriz 1 por la matriz 2.**
20. Imprimir la matriz 1.

Una posible ejecución del programa sería:

```

-19  11  -4  -2  -7  -7  -16  -15
 6    6    9  17  -7  19    2    1
 4   -19   6  -3    7    1   17  -12
-15  -11  -20  -6  -16  -17  -6   17
18   -3   -7  -15  17   16   -9    0
13   18    1  -12  -14   6   11  -17
19   12   -9  -5  -14   -9   -7    0
 7   -1   -2   15  -5    9  -15  -14

14    7   -9  -15   16  -19   -9   -6
15   -5   12  -16  -11  -19  -11  -11
17   13  -11  20   -4  -17   -6  -19
-20  11  -15  16   19    1   18  -14
17  -19  -10   4   -8    4    5  -15
 8   18    1  -20   -6  -17    6   18
-10  -4  -12   15  -13  18   -9   11
-11  -16  -15   10    5   15   18   18

21    49   857  -281  -216  -204  -165  -370
-11   767  -286  -102  238  -664  211  -24
22   177  -378  459  -151  317  -245   -3
-1130 -626  273  261  240  1200  446  849
895  -114   58  -989  -51  -547  -95  202
596  484  537  -836  -374  -833  -976    6
153   33  362  -718  370  -470  -400  -34
 40  734  171  -454  527  -718  142  -383

21    49   857  -281  -216  -204  -165  -370
 0   767  -286  -102  238  -664  211  -24
 0    0  -378  459  -151  317  -245   -3
 0    0    0  261  240  1200  446  849
 0    0    0    0  -51  -547  -95  202
 0    0    0    0    0  -833  -976    6
 0    0    0    0    0    0  -400  -34
 0    0    0    0    0    0    0  -383

14    7   -9  -15   16  -19   -9   -6
 0   -5   12  -16  -11  -19  -11  -11
 0    0  -11  20   -4  -17   -6  -19
 0    0    0  16   19    1   18  -14
 0    0    0    0   -8    4    5  -15
 0    0    0    0    0  -17    6   18
 0    0    0    0    0    0   -9   11
 0    0    0    0    0    0    0   18

294  6055  13398 -12694 -4415 -21282  15306 -12334
 0  -3035  -3106  1042  -6281  9760  12858 -11249
 0    0  -4158  171  6665  36862  30987  23584
 0    0    0  4176  2871  7974  -2845  22178
 0    0    0    0  408  1044  -5144  3983
 0    0    0    0    0  14161  14192  -7200
 0    0    0    0    0    0  3600  -3907
 0    0    0    0    0    0    0  -6894

-20    0    0    0    0    0    0    0
-13   14    0    0    0    0    0    0
 7   -6    0    0    0    0    0    0
11  -12    9    7    0    0    0    0
12   -8   -9  -12   11    0    0    0
-15  -18  -12   19  -19   -4    0    0
 9   -8   20   15  -18   -8    0    0
15    6  -10   20   11   12   -9   17

80551 370654 610333-457326 -55389-105328 111006-209670
-226530-353092 325541 241748-609866-276924 101241-191233
190600-850228-1168901558080-925405-112336-212256 400928
267043 -60784-362623 547171 175243 257000-199602 377026
2685 42994-158910 17440 121057 84772 -35847 67711
-192687-411634 185908 337939-603715-256580 64800-122400
-26205 -52242 111070 -24140-107777 -75684 35163 -66419
-103410 -41364 68940-137880 -75834 -82728 62046-117198

 1    0    0    0    0    0    0    0
 0    1    0    0    0    0    0    0
 0    0    1    0    0    0    0    0
 0    0    0    1    0    0    0    0
 0    0    0    0    1    0    0    0
 0    0    0    0    0    1    0    0
 0    0    0    0    0    0    1    0
 0    0    0    0    0    0    0    1

-20    0    0    0    0    0    0    0
-13   14    0    0    0    0    0    0
 7   -6    0    0    0    0    0    0
11  -12    9    7    0    0    0    0
12   -8   -9  -12   11    0    0    0
-15  -18  -12   19  -19   -4    0    0
 9   -8   20   15  -18   -8    0    0
15    6  -10   20   11   12   -9   17

Process returned 10 (0xA)   execution time : 0.460 s
Press any key to continue.

```

SOLUCIÓN:

```
// includes y defines
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define SIZE 8

// prototipos de las funciones
int aleatorio (int, int);
void rellenar(int [SIZE][SIZE]);
void imprimir(int [SIZE][SIZE]);
void mult(int [SIZE][SIZE], int [SIZE][SIZE]);
void triangularSup(int [SIZE][SIZE])
void triangularInf(int [SIZE][SIZE])
void identidad(int [SIZE][SIZE])

// main
int main()
{
    srand(time(NULL));
    int matA[SIZE][SIZE], matB[SIZE][SIZE];
    //1
    rellenar(matA);
    imprimir(matA);
    printf("\n\n");
    //3
    rellenar(matB);
    imprimir(matB);
    printf("\n\n");
    //5
    mult(matA, matB);
    imprimir(matA);
    printf("\n\n");
    //7
    triangularSup(matA);
    imprimir(matA);
    printf("\n\n");
    //9
    triangularSup(matB);
    imprimir(matB);
    printf("\n\n");
    //11
    mult(matB, matA);
    imprimir(matB);
    printf("\n\n");
    //13
    rellenar(matA);
```

```

    triangularInf(matA);
    imprimir(matA);
    printf("\n\n");
    //15
    mult(matB, matA);
    imprimir(matB);
    printf("\n\n");
    //17
    identidad(matB);
    imprimir(matB);
    printf("\n\n");
    //19
    mult(matA, matB);
    imprimir(matA);
    printf("\n\n");
}

```

```

// Implementación de las funciones
int aleatorio (int inf, int sup)
{
    int aux = inf;
    if (inf > sup)
    { inf = sup;
      sup = aux;
    }
    return (rand()%(sup-inf+1)+inf);
}

```

```

void rellenar(int mat[SIZE][SIZE])
{
    int i, j;
    for (i = 0; i < SIZE; i++)
    for (j = 0; j < SIZE; j++)
        mat[i][j] = aleatorio(-20, 20);
}

```

```

void imprimir(int mat[SIZE][SIZE])
{
    int i, j;
    for (i = 0; i < SIZE; i++)
    {
        printf("\n");
        for (j = 0; j < SIZE; j++)
            printf("%7d ", mat[i][j]);
    }
    printf("\n");
}

```

```
void mult(int matA [SIZE][SIZE], int matB [SIZE][SIZE])
{
    int matAux [SIZE][SIZE], i, j, k, sum;
    for (i = 0; i < SIZE ; i++)
        for (j = 0; j < SIZE; j++)
            {
                for (k = sum = 0; k < SIZE; k++)
                    sum += matA[i][k] * matB[k][j];
                matAux[i][j] = sum;
            }

    for (i = 0; i < SIZE; i++)
        for (j = 0; j < SIZE; j++)
            matA[i][j] = matAux[i][j];
}
```

```
void triangularSup(int mat[SIZE][SIZE])
{
    int i, j;
    for (i = 1; i < SIZE; i++)
        for (j = 0; j < i; j++)
            mat[i][j] = 0;
}
```

```
void triangularInf(int mat[SIZE][SIZE])
{
    int i, j;
    for (i = 0; i < SIZE - 1; i++)
        for (j = 0; j < SIZE; j++)
            if (j > i)
                mat[i][j] = 0;
}
```

```
void identidad(int mat[SIZE][SIZE])
{
    int i, j;
    for (i = 0; i < SIZE; i++)
        for (j = 0; j < SIZE; j++)
            mat[i][j] = i == j;
}
```