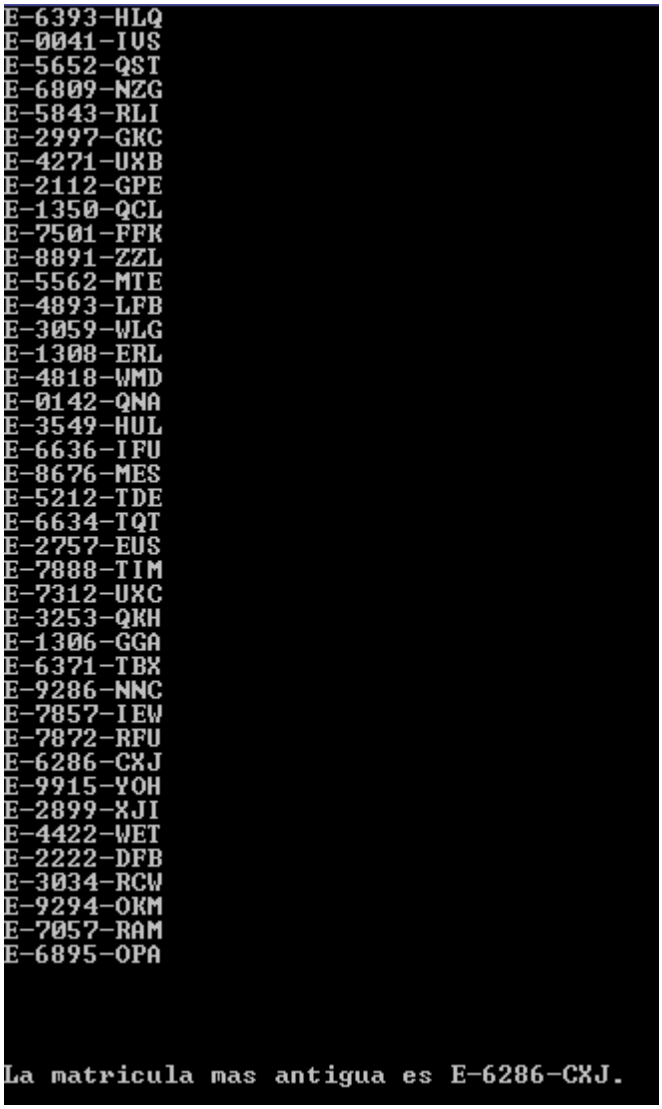
En esta práctica se pide la implementación de un conjunto de funciones y un programa principal para el manejo de cadenas de caracteres junto con matrices. Las cadenas van a ser matrículas de tráfico con el formato E-DDDD-LLL, donde D es un dígito y L una letra del alfabeto (sin incluir la Ñ), siendo válidas todas las letras aunque en la realidad haya algunas letras que no se usen. Por ejemplo, E-2166-BRB o E-8882-FYW serían matrículas válidas. Las funciones que se pide implementar son:

- **rellenar** una matriz de 40 matrículas válidas usando la función aleatorio de las prácticas anteriores.
- **imprimir** el contenido de la matriz.
- **comparar** dos matrículas. Esta función devolverá un ***valor negativo*** si la primera matrícula es más antigua que la segunda, un ***valor positivo*** si la primera matrícula es más reciente que la segunda y ***0*** si son iguales.
- **mas_antigua,** que devuelve la matrícula más antigua de las almacenadas en la matriz.

El programa principal deberá rellenar la matriz con las 40 matrículas, imprimir la matriz y también la matrícula más antigua.

Una posible ejecución de la misma es la siguiente:



SOLUCIÓN:

```
// includes y defines

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

#define MATSIZE 40
#define STRSIZE 10+1

// prototipos

int random(int, int);
void rellenar(char [MATSIZE][STRSIZE]);
void imprimir(char [MATSIZE][STRSIZE]);
int comparar(char * , char * );
char * mas_antigua(char [MATSIZE][STRSIZE]);

// main

int main()
{
   char matriz[MATSIZE][STRSIZE];
   srand(time(NULL));
   rellenar(matriz);
   imprimir(matriz);
   printf("\nMAS ANTIGUA: %s\n", mas_antigua(matriz));
   return 0;
}


// definición de funciones

void rellenar(char mat[MATSIZE][STRSIZE])
{
   int i, j;
   for (i = 0; i < MATSIZE; i++)
   {
      mat[i][0] = 'E';
      mat[i][1] = '-';
      for (j = 2; j <= 5 ; j++)
         mat[i][j] = random('0','9');
      mat[i][6] = '-';
      for (j = 7; j < STRSIZE-1 ; j++)
         mat[i][j] = random('A','Z');
      mat[i][j] = '\0';
   }
}

void imprimir(char mat[MATSIZE][STRSIZE])
```

```
{
    int i;
    for (i = 0; i < MATSIZE; i++)
        printf("%s\n", mat[i]);
}

int comparar(char * matricA, char * matricB)
{
    int n = strcmp(matricA + 7, matricB+7);
    return n ? n : strncmp(matricA + 2, matricB + 2, 4);
}

char * mas_antigua(char mat[MATSIZE][STRSIZE])
{
    int i, n;
    char * antigua = mat[0];
    for (i = 1; i < MATSIZE; i++)
    {
        if (comparar(mat[i], antigua) == 1)
            antigua = mat[i];
    }
    return antigua;
}

int random(int inf, int sup)
{
    return rand()%(sup-inf+1)+inf;
}
```