
 <b>Universidad Nacional de La Matanza</b> 	Tópicos de programación 03635 Recuperatorio Turno Tarde 08/07/2025
Apellido y Nombre:	
DNI:	
Calificación:	

## Recuperatorio - Tópicos de programación

Comisión: 02-2600

### Instrucciones de entrega

ATENCIÓN: para que el examen pueda ser corregido y archivado debe ser entregado obligatoriamente antes de la hora límite con el siguiente formato: DNI\_APELLIDO\_NOMBRE\_P o R.zip, ejemplo 41127133\_PEREZ\_MARIA\_PIA\_R.zip. Observe que es un .zip, no es .rar ni .7z respete la forma de entrega. La P como sufijo indica que es el PARCIAL y la R como sufijo indica que es recuperatorio. Adjunte el enunciado y **elimine las carpetas bin y obj**, está compartiendo programas y por conocidas razones los servidores eliminarán comprimidos que contengan binarios o ejecutables.

## 1. Enunciado

### 1.1. Multiplicación de matrices

Descomente la macro #EXAMEN para implementar su propia función **multiplicarMatrices**. Esta función multiplica matrices de hasta MAX\_FIL, MAX\_COL, que están seteadas ambas macros en 50 (No lo modifique). La matriz resultante es pasada como último argumento. El valor de retorno será 0 cuando no se puedan multiplicar matrices, esto ocurre cuando la cantidad de columnas de la matriz 1 es diferente a la cantidad de filas de la matriz 2. Analice el funcionamiento del ejemplo y opere de igual forma. No cambie el prototipo de la función. Si se adjunta un caso de prueba es solo eso, un simple caso de prueba, usted es responsable de probar y verificar el correcto funcionamiento de su implementación con cualquier matriz bidimensional entre 0x0 hasta un máximo de 50x50.

```
int multiplicarMatrices(int mat1[][MAX_COL],
                        int fmat1, // Cantidad de filas de la matriz 1
                        int cmat1, // Cantidad de columnas de la matriz 1
                        int mat2[][MAX_COL],
                        int fmat2, // Cantidad de filas de la matriz 2
                        int cmat2, // Cantidad de columnas de la matriz 2
                        int matR[][MAX_COL]); // Matriz resultado
```

## 1.2. strrchr recursivo

Descomente la macro `#EXAMEN` e implemente su propia versión de la función **`rstrrchr`**. Esta función debe copiar fielmente el comportamiento de `strrchr`, pero deberá ser completamente recursiva (sin bucles ni iteraciones). No utilice ninguna función de biblioteca para resolverla. Para mas información sobre esta función tiene disponible el pdf de ANSI C.

El prototipo de la función es el siguiente:

```
char *rstrrchr(const char *s, int c);
```

No lo cambie, recuerde que usted es responsable de probar y verificar el correcto funcionamiento de su implementación.

## Requerimientos funcionales

El resultado debe ser correcto. Si hubiese casos de prueba debe ser correcto para los casos entregados u otros casos posibles.

## Requerimientos no funcionales

El resultado es condición necesaria pero no suficiente. Se deben cumplir ademas los requerimientos no funcionales.

- Entregue sin errores de compilación o warnings.
- Si se entrega un proyecto y se lo indica, no modificar la firma de las funciones.
- Vectores y cadenas de texto deberán ser manipulados utilizando aritmética de punteros.
- No se permite el uso de VLA (Variable length arrays)
- Declaraciones siempre al inicio de bloque por compatibilidad ANSI C.
- El sistema debe ser eficiente:
  - En el uso de memoria
    - No declare vectores o matrices auxiliares si no es necesario.
    - No cargue en memoria mas de lo necesario.
  - En cantidad de ciclos de procesador y en el caso de matrices las soluciones deben ser óptimas.
- No almacene archivos completos en memoria excepto que la problemática planteada lo exija (Normalmente la descripción del problema define que el conjunto de datos es conocido y finito).
- Utilice nombres descriptivos, separe en funciones y cuide la prolijidad.