

Proyecto de programación en ensamblador

Estructura de Computadores

- El proyecto consiste en la programación, en ensamblador del Motorola 88110, de un conjunto de rutinas que realicen el *filtrado de una imagen* mediante un filtro programable.
- La imagen será una matriz de píxeles, cada uno de los cuales se representa mediante un *byte sin signo* que especifica su nivel de gris (0 equivale a negro y 255 a blanco).
- El filtro está basado en una operación recursiva de convolución con un núcleo representado por una matriz de 3x3 valores enteros que define una matriz 3x3 de coeficientes fraccionarios:
 - Cada elemento del filtro es en realidad el número que se obtiene al dividir el correspondiente elemento de la matriz 3x3 de coeficientes, entre la suma de los 9 coeficientes.

Enunciado feb-jul 2018

- El enunciado de este proyecto está basado en el planteado el curso pasado, por lo que aquellos alumnos que tengan que repetir o corregir el proyecto que desarrollaron entonces, podrán partir de los programas ya realizados anteriormente. Sin embargo, algunas o todas las pruebas del proyecto que se realizarán en este curso serán diferentes, por lo que, en caso necesario, *deberán adaptar la implementación de las subrutinas de modo que superen las pruebas que se establezcan, tanto para la convocatoria de febrero, como para la de julio.*
- Por otra parte debe observar con atención el apartado de este documento que describe las normas de entrega y, en particular, la especificación del contenido que debe incluir la memoria del proyecto y las fechas de entrega de los *hitos evaluables*.
- El hecho de plantear un proyecto similar al del curso pasado tiene además las siguientes implicaciones:
 - Los alumnos que ya hubieran formado parte de un grupo durante convocatorias anteriores y hubieran realizado al menos una entrega **solo podrán establecer grupo con el mismo compañero** de dicha convocatoria o, alternativamente, realizar el proyecto de **forma individual**.
 - Se realizará una revisión minuciosa de los proyectos realizados en este semestre para descartar o localizar posibles **casos de copia** que desafortunadamente se siguen produciendo (y detectando) en la mayoría de las convocatorias.

Definición del filtro

Filtro es una matriz cuadrada de orden 3 cuyos elementos son fraccionarios y están definidos por los nueve elementos de la matriz (enteros con signo)

Ejemplo:

Filtro: 1, 1, 1, 1, 0, 1, 1, 1, 1 $\Sigma = 8$

Matriz de filtro:

1/8	1/8	1/8	0,125	0,125	0,125
1/8	0/8	1/8	0,125	0,0	0,125
1/8	1/8	1/8	0,125	0,125	0,125

Cálculo del filtro de una imagen (1)

$Im_original = (m[i,j]), \quad Filtro = (f[i,j]), \quad Im_filtrada = (r[i,j])$

$$\begin{aligned} r[i,j] = & f[0,0] \cdot m[i-1,j-1] + f[0,1] \cdot m[i-1,j] + f[0,2] \cdot m[i-1,j+1] + \\ & + f[1,0] \cdot m[i,j-1] + f[1,1] \cdot m[i,j] + f[1,2] \cdot m[i,j+1] + \\ & + f[2,0] \cdot m[i+1,j-1] + f[2,1] \cdot m[i+1,j] + f[2,2] \cdot m[i+1,j+1] \end{aligned}$$

Im_original								Filtro		
1	2	3	4	5	6	7	...	0,125	0,125	0,125
5	6	7	8	9	0	1	...	0,125	0,0	0,125
9	0	1	2	3	4	5	...	0,125	0,125	0,125
3	4	5	6	7	8	9	...			
7	8	9	0	1	2	3	...			
.....										

Valor de $m[2,2]=1$ Valor de $r[2,2]=4,75 \rightarrow 4$

Cálculo del filtro de una imagen (2)

Filtro			MFiltro			
0,125	0,125	0,125	1	1	1	
0,125	0,0	0,125	1	0	1	$\Sigma = 8$
0,125	0,125	0,125	1	1	1	

Subimagen

6	7	8
0	1	2
4	5	6

- Se opera con la matriz MFiltro definida por sus nueve elementos enteros
- Se divide al final entre $\sum \mathbf{MFiltro}_{ij}$ (instrucción “divs”)

Valor de $r[2,2] = (6+7+8+0+0+2+4+5+6)/8 = 38/8$

Aplicando “divs” se obtiene como resultado: 4

Filtrado de los bordes de una imagen

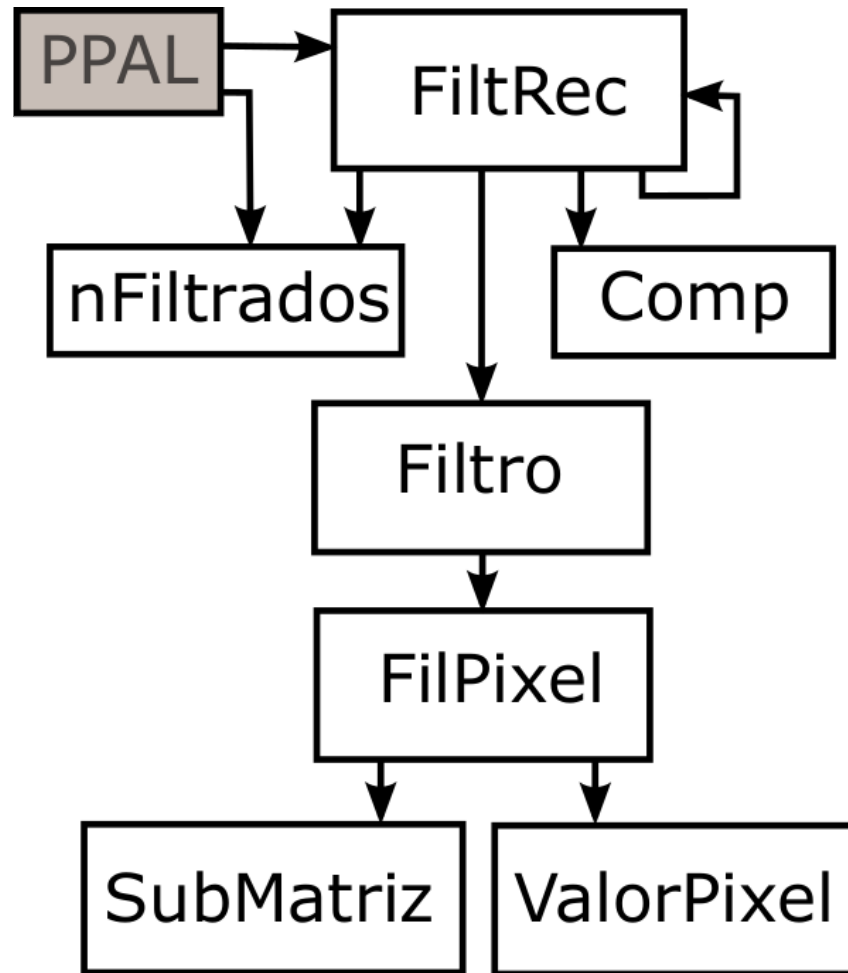
Imagen Original

1	2	3	4	5	6	7	...
5	6	7	8	9	0	1	...
9	0	1	2	3	4	5	...
3	4	5	6	7	8	9	...
7	8	9	0	1	2	3	...
...

Subimágenes de los bordes

1	1	1					
1	1	1	3	4	6	6	6
1	1	1	7	8	6	6	6
	9	0	1	2	3	4	5
	3	4	5	6	7	8	9
	7	8	9	0	1	2	3

Jerarquía de las rutinas



Número de filtrados

NFiltrados = **nFiltrados** (**oper**)

- Parámetros (en la pila):
 - oper**: De entrada. Se pasa por valor. Indica qué operación debe realizar la subrutina.
- Valor de retorno:
 - NFiltrados**: Devuelve en r29 el valor de la variable estática nF ó -1.
- Algoritmo:
 1. Si $oper = 0 \rightarrow$ Inicializa la variable estática nF a 0. $Nfiltrados = nF$.
 2. Si $oper \neq 0 \rightarrow$ Incrementa la variable estática nF. $Nfiltrados = nF$.
 3. Si $nF \geq 12$, $nF = 12$. $Nfiltrados = -1$.
 4. Retornar Nfiltrados en r29.

Compara imágenes

Diferencias = Comp (Imagen1, Imagen2)

- Parámetros (en la pila):

Imagen1: De entrada. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno).

Imagen2: De entrada. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno).

- Valor de retorno:

Diferencias: Devuelve en r29 el valor de la diferencia entre las imágenes.

- Algoritmo:

1. Inicializar a cero un acumulador de diferencias (*Dif*).
2. Recorrer los MxN píxeles de cada matriz incrementando Dif en el valor absoluto de la diferencia: $Dif = Dif + \text{abs}(Im1_{ij} - Im2_{ij})$.
3. Retornar cargando Dif en r29.

Valor del píxel filtrado (1)

VPixel = ValorPixel (SubImg, MFiltro)

- Parámetros (en la pila):
 - SubImg:** De entrada. Se pasa por dirección: 9 elementos (1 byte sin signo cada uno). Es la submatriz de 3x3 centrada en el píxel a filtrar.
 - MFiltro:** De entrada. Se pasa por dirección. Formada por:
 - Nueve valores correspondientes a los nueve elementos de la matriz 3x3 que definen el núcleo de filtrado
- Valor de retorno:
 - VPixel:** Devuelve en r29 el valor provisional del píxel filtrado (entero con signo de 32 bits) .
- Descripción:
 - Aplica el filtro Mfiltro a la submatriz SubImg.

Valor del píxel filtrado (2)

VPixel = ValorPixel (SubImg, MFiltro)

- Algoritmo
 1. Inicializa a cero el acumulador ACC.
 2. Inicializa un puntero al primer elemento de SubImg y otro al primer elemento de la matriz de filtro.
 3. Recorre, en un bucle, los 9 elementos de la matriz de filtro y los correspondientes de SubImg multiplicándolos entre sí y acumulando el resultado. En cada iteración:
 - Se pasa el valor del píxel sin signo a un registro **rxx**
 - Se multiplica **rxx** por el coeficiente del filtro y se acumula en ACC
 5. Se asigna r29 = ACC y se retorna al llamante.

Extracción de Submatriz (1)

SubMatriz (Imagen, SubImg, i, j)

- Parámetros (en la pila):
 - Imagen:** De entrada. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno). Es la imagen de la que se debe extraer una submatriz de 3x3 elementos.
 - SubImg:** De salida. Se pasa por dirección: 9 elementos (1 byte sin signo cada uno). Es la matriz de 3x3 elementos en la que se debe copiar la submatriz de *Imagen* centrada en el pixel [i, j].
 - i:** Número de fila. (0..M-1). De entrada. Se pasa por valor.
 - j:** Número de columna. (0..N-1). De entrada. Se pasa por valor.
- Descripción:
 - Obtiene la submatriz *SubImg* de 3x3 elementos centrada en el píxel [i, j] de la *Imagen*.

Extracción de Submatriz (2)

SubMatriz (Imagen, SubImg, i, j)

- Algoritmo

1. Determinar si el píxel $[i, j]$ pertenece al borde de la Imagen, es decir, si $i=0$ o $j=0$ o $i=M-1$ o $j=N-1$.
2. Si pertenece al borde: Copiar el píxel $[i, j]$ sobre los 9 píxeles de SubImg.
3. Si no pertenece al borde: Copiar la submatriz de Imagen centrada en $[i, j]$ sobre los 9 píxeles de SubImg.

Filtro de un píxel (1)

```
VPixel = FilPixel (Imagen, i, j, MFiltro)
```

- **Parámetros (en la pila):**
 - Imagen:** De entrada. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno). Es la imagen que contiene el píxel a filtrar.
 - i:** Número de fila. De entrada. Se pasa por valor.
 - j:** Número de columna. De entrada. Se pasa por valor.
 - MFiltro:** De entrada. Se pasa por dirección. Formada por:
 - Nueve valores correspondientes a los nueve elementos de la matriz 3x3 que definen el núcleo de filtrado
- **Valor de retorno:**
 - VPixel:** Devuelve en r29 el valor del píxel filtrado (entero sin signo en el rango (0..255)).
- **Descripción:**
 - Aplica la máscara de filtrado al píxel [i,j] de la imagen.

Filtro de un píxel (2)

```
VPixel = FilPixel (Imagen, i, j, MFiltro)
```

- Algoritmo
 1. Reserva espacio en el marco de pila para almacenar una submatriz de 3x3 bytes sin signo, (SubImg), (3 palabras).
 2. Llama a SubMatriz (Imagen, SubImg, i, j).
 3. Llama a ValorPixel (SubImg, Mfiltro).
 4. Calcula el peso del filtro: $\text{peso} = \sum \text{Mfiltro}_{ij}$.
 5. Si $\text{peso} \neq 0$ se divide r29 entre peso usando división entera (divs).
 6. Si $r29 < 0$ se ajusta a 0. Si $r29 > 255$ se ajusta a 255.
 7. Retorna al llamante teniendo r29 el valor del píxel [i, j] filtrado.

Filtro de una imagen (1)

Filtro (Imagen, ImFiltrada, MFiltro)

- Parámetros (en la pila):
 - Imagen:** De entrada. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno). Es la imagen que se debe filtrar.
 - ImFiltrada:** De salida. Es la imagen que resulta de aplicar el filtro a la imagen de entrada. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno).
 - MFiltro:** De entrada. Se pasa por dirección. Formada por:
 - Nueve valores correspondientes a los nueve elementos de la matriz 3x3 que definen el núcleo de filtrado
- Descripción:
 - Aplica la máscara de filtrado definida por MFiltro a la imagen de entrada. Deja el resultado en la dirección definida por ImFiltrada.

Filtro de una imagen (2)

Filtro (Imagen, ImFiltrada, MFiltro)

- Algoritmo:
 1. Copiar M y N sobre la imagen filtrada.
 2. Desde $i=0$ hasta $i=M-1$
 - Desde $j=0$ hasta $j=N-1$
 1. Preparar parámetros y llamar a FilPixel (Imagen, i, j, MFiltro).
 2. Almacenar el valor de retorno en la posición [i,j] de la imagen filtrada.
 3. Retornar al llamante.

Filtro recursivo (1)

```
Diferencia=FiltRec(ImagenIn, ImagenOut, MFiltro, NCambios)
```

- Parámetros (en la pila):

ImagenIn: De entrada. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno). Es la imagen de entrada a la que se ha de aplicar el filtro.

ImagenOut: De salida. Se pasa por dirección: M (1 palabra), N (1 palabra), MxN elementos (1 byte cada uno). Es la imagen que resulta de aplicar el filtro.

MFiltro: De entrada. Se pasa por dirección. Formada por:

- Nueve valores correspondientes a los nueve elementos de la matriz 3x3 que definen el núcleo de filtrado

NCambios: De entrada y se pasa por valor (1 palabra). Es el valor de la diferencia entre la imagen original y la filtrada que determina si se debe hacer o no una nueva llamada recursiva.

Filtro recursivo (2)

Diferencia=FiltRec(ImagenIn, ImagenOut, MFiltro, NCambios)

- Valor de retorno:

Diferencia: Devuelve en r29 el valor de la diferencia entre las imágenes usadas en la última operación de filtrado.

- Descripción:

- Realiza filtrados de una imagen de entrada ImagenIn de forma recursiva, dejando el resultado final en ImagenOut.

- Condiciones de salida de la recursividad:

1. La diferencia entre la imagen original y la filtrada es **menor** que NCambios.
2. NFiltrados < 0 (NFiltrados es el valor devuelto por la subrutina nFiltrados a la que se llama con el parámetro oper ≠ 0).

Filtro recursivo (3)

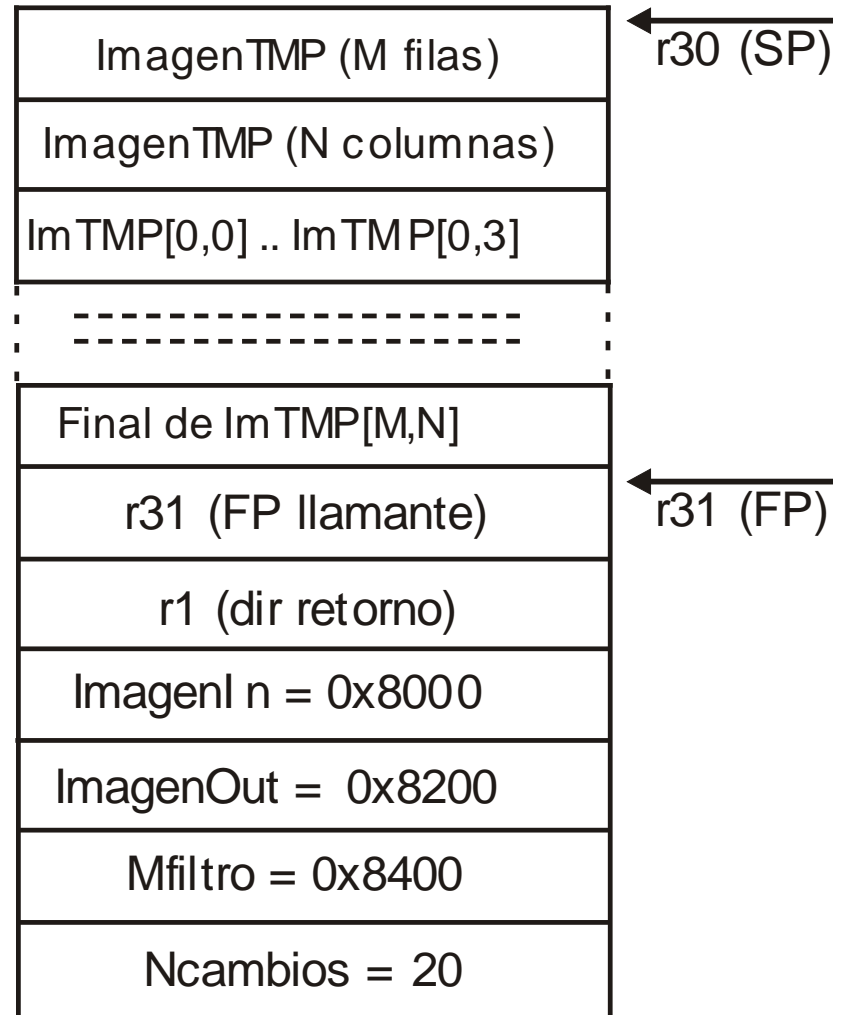
Diferencia=FiltRec(ImagenIn, ImagenOut, MFiltro, NCambios)

- Algoritmo:
 1. Llamar a Filtro (ImagenIn, ImagenOut, Mfiltro).
 2. Llamar a Comp pasando las imágenes original (ImagenIn) y filtrada (ImagenOut). Si r29<NCambios continuar en paso 6.
 3. Salvar r29 en la pila. Llamar a la subrutina nFiltrados con el parámetro oper distinto de 0. Recuperar r29 de la pila. Si Nfiltrados es < 0 → continuar en paso 6.
 4. Reservar espacio en el marco de pila para la variable temporal ImagenTmp: 4+4+M·N ajustado por exceso a múltiplo de 4. Copiar la imagen filtrada (ImagenOut) en la variable temporal ImagenTmp.
 5. Llamar a FiltRec pasando la copia de la imagen filtrada (ImagenTmp), la imagen de salida (ImagenOut), Mfiltro y NCambios. (Al retornar de esta llamada recursiva continúa en paso 6. El valor devuelto en r29 es la diferencia obtenida tras la última operación de filtrado. La imagen filtrada, ImagenOut, se habrá actualizado con el resultado de la última operación de filtrado)
 6. Retornar al llamante con r29=Diferencia.

Tratamiento de la pila

FiltRec:

```
PUSH    (r1)
PUSH    (r31)
addu    r31,r30,r0
ld      r10,r31,8
.....
```



Filtro de un píxel. Caso 11.

r30=36848 (0x8FF0)

Direcciones de memoria:

36848	00800000	02000000	02000000	30800000
-------	----------	----------	----------	----------

32768	04000000	08000000	44444444	44444444
-------	----------	----------	----------	----------

32784	44343433	44444444	44448844	44444444
-------	----------	----------	----------	----------

32800	44444444	44444444		
-------	----------	----------	--	--

32816	01000000	01000000	01000000	01000000
-------	----------	----------	----------	----------

32832	00000000	01000000	01000000	01000000
-------	----------	----------	----------	----------

32848	01000000			
-------	----------	--	--	--

Resultado:

r30=36848 (0x8FF0) r29=61 (0x3D)

Filtro de un píxel. Caso 11.

Imagen:

44	44	44	44	44	44	44	44
44	34	34	33	44	44	44	44
44	44	88	44	44	44	44	44
44	44	44	44	44	44	44	44

Mfiltro:

1	1	1
1	0	1
1	1	1

Filtro recursivo. Caso 17.

r30=36848 (0x8FF0)

Direcciones de memoria:

00000	00000000			
36848	00800000	54800000	30800000	00000000
32768	04000000	08000000	FF0000FF	FF0000FF
32784	FF0000FF	FF0000FF	FF0000FF	FF0000FF
32800	FF0000FF	FF0000FF		
32816	01000000	01000000	01000000	01000000
32832	00000000	01000000	01000000	01000000
32848	01000000			
32848		04000000	08000000	FF0000FF
32864	FF0000FF	FF0000FF	FF0000FF	FF0000FF
32880	FF0000FF	FF0000FF	FF0000FF	

Filtro recursivo. Caso 17.

Resultado:

r30=36848 (0x8FF0) r29=0 (0x00)

Direcciones de memoria

00000 0C000000

32768 04000000 08000000 FF0000FF FF0000FF

32784 FF0000FF FF0000FF FF0000FF FF0000FF

32800 FF0000FF FF0000FF

32848 04000000 08000000 FF0000FF

32864 FF0000FF FF8F7795 95778FFF FF8F7795

32880 95778FFF FF0000FF FF0000FF

Filtro recursivo. Caso 17.

Entrada:

FF	00	00	FF	FF	00	00	FF
FF	00	00	FF	FF	00	00	FF
FF	00	00	FF	FF	00	00	FF
FF	00	00	FF	FF	00	00	FF

Resultado:

FF	00	00	FF	FF	00	00	FF
FF	8F	77	95	95	77	8F	FF
FF	8F	77	95	95	77	8F	FF
FF	00	00	FF	FF	00	00	FF

r29=12 (0x0C)

Programas de prueba

LEA: MACRO...

org xxx

D1: data...

D2: data...

D3: res...

org yyy

PPAL: "inicialización de la pila"

...

"paso de parámetros (PUSH)"

...

bsr subrutina

"vaciado de la pila (POP)".

stop

Entrega (fechas)

CONVOCATORIA DE FEBRERO 2018

El plazo de entrega del proyecto estará abierto para entregar código y memoria:

Desde el viernes día 20 de octubre hasta el viernes día 17 de noviembre de 2017

Pasada esa fecha aún se podrá entregar la memoria del proyecto desde el día 24 hasta el día 30 de noviembre a las 20:00.

Cada grupo podrá disponer de las siguientes correcciones:

- Lunes 23/10/2017: primera corrección
- Viernes 27/10/2017: 1er hito evaluable (subrutinas nFiltrados y Comp, 10% de la nota)
- Lunes 6/11/2017: 2º hito evaluable (subrutinas nFiltrados, Comp, ValorPixel y SubMatriz, 15% de la nota)
- Viernes 17/11/2017: Última corrección
- Fechas a elegir para realizar un máximo de cuatro correcciones adicionales:
 - Entre 24 a 26, 30 y 31 de octubre y 2, 3, 7 a 10 y 13 a 16 de noviembre

Todas las correcciones se realizarán a partir de las 21:00. Para solicitar una corrección bastará con entregar correctamente los ficheros del proyecto antes de dicha hora límite.

El examen del proyecto está planificado para el miércoles día 22 de noviembre a las 14:00 (horario habilitado por Jefatura de Estudios para evaluación).

Entrega (contenido)

1. *Memoria*, en formato DINA4, en cuya portada deberá figurar claramente el nombre y apellidos de los autores del proyecto, identificador del grupo de alumnos (el mismo que emplean para realizar las entregas y consultas) y el nombre de la asignatura. Dicha memoria se entregará en formato electrónico *PDF*.

2. *Ficheros*:

1. **autores**: Es un fichero ASCII que deberá contener los apellidos, nombre, número de matrícula, DNI y dirección de correo electrónico de los autores del proyecto. El proyecto se realizará individualmente o en grupos de dos alumnos. Cada línea de este fichero contendrá los datos de uno de los autores de acuerdo al siguiente formato:

No Matrícula; DNI ; apellido apellido, nombre; correo electrónico

NOTA: este fichero solo se entrega en el momento de hacer el registro en el Gestor de Prácticas. Después de ese momento se accederá mediante el par usuario:clave utilizados durante el registro

2. **filtror.ens**: Contendrá las subrutinas que componen el proyecto junto con un programa principal que se haya utilizado para su depuración.
3. **memoria.pdf**: fichero conteniendo la memoria del proyecto.

Entrega (memoria)

La memoria debe contener los siguientes puntos:

- Histórico del desarrollo de las rutinas, con fechas, avances y dificultades encontradas, especificando el trabajo que realiza cada miembro del grupo o si dicho trabajo es común. Se detallará en este apartado:
 - Número total de horas invertidas en el proyecto por cada miembro del grupo.
 - Relación de visitas realizadas a los profesores del proyecto.
- Descripción *resumida* del juego de ensayo (conjunto de casos de prueba) que el grupo haya diseñado y utilizado para probar el correcto funcionamiento del proyecto.
- Observaciones finales y comentarios personales de este proyecto, entre los que se debe incluir una descripción de las principales dificultades surgidas para su realización.

Evaluación (2017/2018)

- El proyecto consta de tres partes: código y memoria (*código*), pruebas de funcionamiento (*pruebas*) y examen del proyecto (*examen*). Para superar el proyecto se deberá obtener la calificación de **apto en cada una de las tres partes**.
- Para que un grupo supere la parte de pruebas será necesario que en la última corrección (o con anterioridad), el fichero entregado por el grupo pase todas las pruebas de las subrutinas **nFiltrados**, **Comp**, **ValorPixel** y **SubMatriz**, así como que el sistema de corrección detecte un máximo de dos pruebas fallidas de **FilPixel**, cuatro de **Filtro** y seis de **FiltRec**.
- Por otra parte, el 25% de la nota de *pruebas* está condicionado al logro de los hitos evaluables siguientes en las fechas anteriormente indicadas :
 - *A: Superar todas las pruebas de las subrutinas nFiltrados y Comp (10% de la nota).*
 - *B: Superar todas las pruebas de las subrutinas nFiltrados, Comp, ValorPixel y SubMatriz (15% de la nota).*
- El 75% restante se obtendrá a partir del resultado alcanzado en la última corrección. Se obtendrá la calificación máxima si se superan todas las pruebas evaluables y además se han superado con éxito y en las fechas establecidas los dos hitos evaluables.
- La nota de la parte de pruebas representará un 70% de la calificación del proyecto en caso de que se cumplan los requisitos establecidos para superar la parte de código y memoria, mientras que el examen del proyecto (individual) representará el 30% restante.

Ejemplo con imagen real

- Procesada con una versión escrita en C que realiza las mismas operaciones que el código del proyecto.
- La implementación en C tiene la misma estructura que la descrita en este proyecto
- Los valores de la matriz de filtro utilizada son:

$$\text{MFiltro: peso}=15 \begin{bmatrix} 3 & 3 & 3 \\ 0 & 15 & 0 \\ -3 & -3 & -3 \end{bmatrix}$$

definida como:

$$\text{MFiltro: } \{3, 3, 3, 0, 15, 0, -3, -3, -3\}$$

Imagen original:



Imagen procesada (Mfiltro: $\{3,3,3, 0,15,0, -3,-3,-3\}$ 10 filtrados):

