# Trabajo Práctico

## Manipulación de imágenes sin compresión

## Modalidad de entrega

La entrega es grupal, en grupos de 2 ó 3 personas.
Los grupos de 3 personas deberán hacer algunos puntos extra, tal cual dice el enunciado.
Si algún/a integrante dejara la materia, las condiciones son las mismas para el resto de los/las integrantes del grupo: si éste tuviera 3 personas, cada integrante asumirá el compromiso de hacer el TP con los puntos extra, y en caso de que algún/a integrante dejara la materia, los/las integrantes restantes del grupo deberán hacer el compromiso asumido inicialmente.

Se deberá entregar un archivo con el siguiente formato:
TP_Topicos_2024_1c_miercoles_{NOMBRE_DE_GRUPO}.zip

Por ejemplo: Si el grupo se llamase “Invisible” y sus integrantes fueran Spinetta, Pomo y Machi, el archivo debería llamarse TP_Topicos_2024_1c_miercoles_INVISIBLE.zip
Dentro de ese archivo debe haber **solamente dos archivos** (sin ninguna carpeta que los contenga) llamados funciones_estudiante.h y funciones_estudiante.c
El archivo funciones_estudiante.c tiene en un comentario los campos a completar con los datos de sus integrantes, y una sección de comentarios para que el corrector lea. Esta última sección es de uso opcional.

Los grupos deben ser formados y notificados vía mensajería de MIEL hasta las 23:59 del miércoles 15 de mayo de 2024. El nombre de cada grupo debe ser una palabra, y no se puede repetir con otro grupo.

El trabajo práctico tiene valor de parcial y consta de una entrega (grupal) y una defensa (individual).

## Modalidad de defensa

La defensa es individual, en laboratorio. Se solicitará hacer modificaciones en el programa para extender sus funcionalidades.

## Requisitos de entrega

La resolución del TP debe ser entregada vía plataforma MIEL. La fecha de vencimiento de la entrega es el día miércoles 5 de junio a las 23:59 hs.
Las entregas que no respeten la fecha indicada no serán válidas.

El trabajo práctico entregado debe funcionar correctamente, sin warnings y cumplir con todas las funcionalidades requeridas.

Debe cumplir con las especificaciones de nombre y formato de archivo, de lo contrario la entrega será inválida y el TP desaprobado.
Los grupos deberán revisar adecuadamente el trabajo práctico previo a la entrega, dado que una vez entregado no se aceptarán reentregas.

Una vez entregado, en caso de cumplir con las condiciones planteadas de fecha, resolver adecuadamente cada uno de los puntos indicados, no tener warnings y tener el formato adecuado de archivo, el TP pasará a estar en estado “entregado”.

En caso de no cumplir con alguno de los requisitos, se podrá utilizar la fecha de recuperatorio (quitando por esto la posibilidad de recuperar el parcial) para hacer una nueva entrega, y defensa. La nueva entrega tendrá nuevos requerimientos que se informarán debidamente. Y la defensa tendrá las mismas condiciones que la defensa inicial.

# Enunciado

Se solicita realizar un programa en lenguaje C llamado *bmpmanipuleitor*, que a partir de un archivo BMP de 24 bits de profundidad genere otro archivo BMP con las modificaciones solicitadas como argumentos a main.

Por ejemplo:

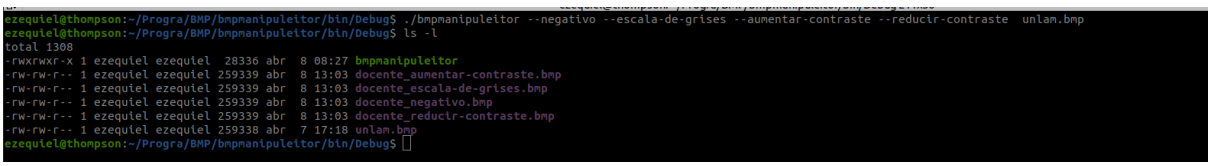
```
bmpmanipuleitor --negativo imagenOriginal.bmp
```

Debe generar un archivo llamado *estudiante_negativo.bmp* que contenga la misma imagen, pero con los colores invertidos.

Del mismo modo, deberá soportar las siguientes funcionalidades:

| | |
|---|---|
| --escala-de-grises | // deberá promediar los valores de cada color RGB y transformarlo a gris |
| --aumentar-contraste | // aumenta el contraste en un 25% |
| --reducir-contraste | // reduce el contraste en un 25% |
| --tonalidad-azul | // aumenta en un 50% la intensidad del color azul |
| --tonalidad-verde | // aumenta en un 50% la intensidad del color verde |
| --tonalidad-roja | // aumenta en un 50% la intensidad del color rojo |
| --recortar | // reduce el tamaño de la imagen al 50%, sin cambiar sus proporciones, sólamente descarta lo que exceda ese tamaño. Por ejemplo: una imagen de 1000px x 500px, deberá mantener todos los píxeles que estén entre 0 y 499 en el eje X y entre 0 y 249 el eje Y. |
| --rotar-derecha | // gira la imagen 90 grados a la derecha |
| --rotar-izquierda | // gira la imagen 90 grados a la derecha |
| --comodin | // una funcionalidad más, que el grupo decidirá cuál será y cómo implementarla. |

Nota: los argumentos pueden enviarse de a uno, o de a varios, de modo que en un solo llamado al programa, ejecute todas las modificaciones y genere todos los archivos necesarios. Ejemplo de un llamado al programa, y de los archivos generados:




Para los grupos de tres personas, se agregarán las siguientes funcionalidades (es decir, deben realizar las anteriores y éstas):

| | |
|---|---|
| --concatenar | // recibirá como argumento dos imágenes, y creará una nueva con el contenido de ambas, una arriba de la otra, en caso de tener distinto ancho, la más pequeña se rellenará con algún color a gusto (que no sea ni blanco ni negro, dejando de lado el debate de que sean o no un color) hasta completar el ancho de la otra |
| --achicar | // reduce la imagen al 50%. Por ejemplo: una imagen de 1000px x 500px, deberá reescribir todos los píxeles para que se ubiquen entre 0 y 499 en el eje X y entre 0 y 249 el eje Y, formando la misma imagen inicial (con peor calidad, claro). |
| --monocromo | // modifica la imagen para que en lugar de ser de 24 bits de profundidad, sea de 1 bit de profundidad. Este tipo de imágenes almacena en un byte 8 pixeles distintos, que pueden obviamente tomar valores 0 ó 1. El criterio para saber si se guarda un 0 ó un 1, es que el promedio de los valores RGB (ver más adelante) dé entre 0 y 127 ó entre 128 y 244. |

Estos últimos tres puntos requieren modificaciones en el encabezado del archivo BMP, cuya descripción es la siguiente:

| Bytes | Información |
|---|---|
| 0, 1 | Tipo de fichero "BM" |
| 2, 3, 4, 5 | Tamaño del archivo |
| 6, 7 | Reservado |
| 8, 9 | Reservado |
| 10, 11, 12, 13 | Inicio de los datos de la imagen |
| 14, 15, 16, 17 | Tamaño de la cabecera del bitmap |
| 18, 19, 20, 21 | Anchura (píxels) |
| 22, 23, 24, 25 | Altura (píxels) |
| 26, 27 | Número de planos |
| 28, 29 | Tamaño de cada punto |
| 30, 31, 32, 33 | Compresión (0=no comprimido) |
| 34, 35, 36, 37 | Tamaño de la imagen |
| 38, 39, 40, 41 | Resolución horizontal |
| 42, 43, 44, 45 | Resolución vertical |
| 46, 47, 48, 49 | Tamaño de la tabla de color |
| 50, 51, 52, 53 | Contador de colores importantes |

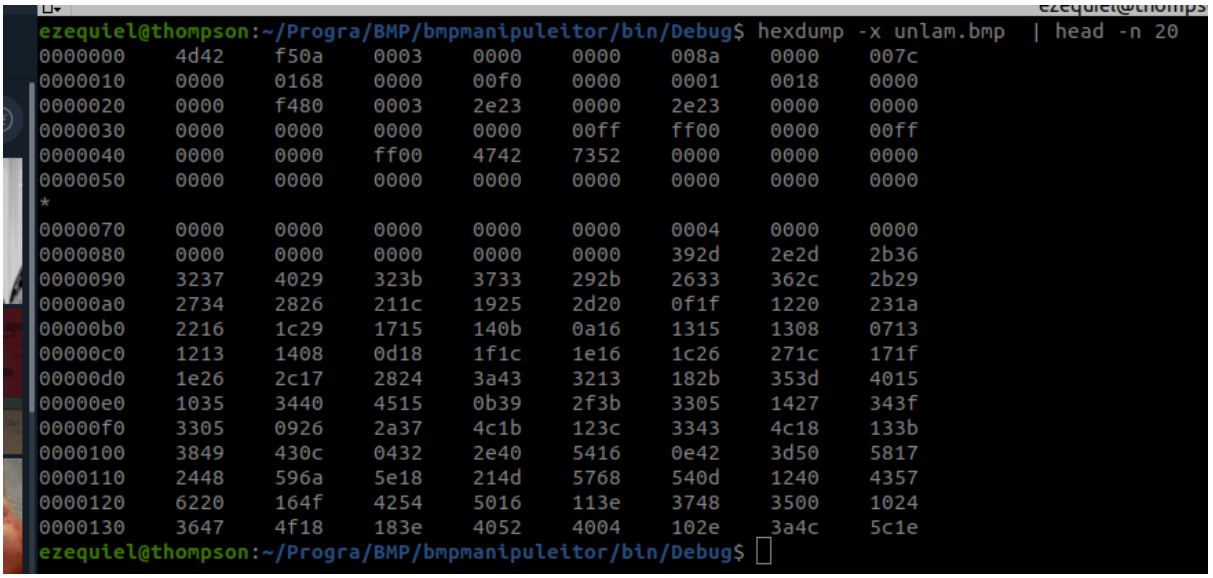
Muchos campos de este encabezado no son necesarios para la realización del TP, sí es necesario contemplar que hay campos de 16 y de 32 bits y que deben ser almacenados en el tipo de dato apropiado, para su correcta interpretación.
Alto, ancho, tamaño de imagen, tamaño de cada punto/píxel, inicio de los datos de la imagen son datos de suma importancia para la realización del TP.

Todas las modificaciones deben hacerse sobre los archivos funciones_estudiante.h y funciones_estudiante.c

## Detalles a tener en cuenta

Una imagen como las que vamos a utilizar en este TP está formada por un encabezado y un array de bits que representa la imagen. Este array de bits, al haber definido que cada pixel tiene 24 bits de profundidad, se debe interpretar como un pixel al contenido de 3 bytes consecutivos.

Podría ser de utilidad utilizar un programa que haga un dump hexadecimal, si no conocen ninguno… podrían hasta resolverlo como punto extra.




Los tres bytes consecutivos son valores que van entre 0 y 255 (unsigned char), o más claramente en hexadecimal: 0x00 hasta 0xff, que indican qué cantidad de color (rojo, verde o azul) va a tener la imagen.
Si las tres variables RGB (Red, Green, Blue; o en español Rojo, Verde y Azul) están igualadas, el color será gris, y dependiendo de su intensidad, se acercará más al blanco o al negro.
El color 0xffffff equivale al blanco, mientras que el 0x000000 equivale al negro.
Si quisiéramos escribir un color azul puro, debería ser 0x0000ff (porque tiene cero en la componente roja, cero en la componente verde y 255 en la componente azul).
De ese modo se pueden armar 2^24 colores distintos.

A continuación un ejemplo de un comando ejecutado, y los resultados (en vista previa) obtenidos:


Nota: el ejemplo fue ejecutado utilizando un sistema GNU/Linux, y como en cualquier

sistema tipo Unix, los ejecutables utilizan *./nombreDelPrograma*, en lugar del formato que utiliza Windows de ser nombreDelPrograma.exe