

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Estructuras de Datos
Proyecto #1
Ingenieros

- Ing. Luis Espino
- Ing. Jesús Guzmán
- Ing. Álvaro Hernández

Auxiliares

- Dennis Masaya
- Ricardo Cutz
- Antonio Hernández



PHOTGEN++

Objetivos

1. Familiarizarse con los lenguajes de programación C++.
2. Familiarizarse con los lenguajes HTML y SCSS.
3. Comprender y desarrollar distintas estructuras de datos lineales como lo son listas, listas enlazadas dobles, listas circulares, pilas y colas.
4. Comprender y desarrollar distintas estructuras de datos no lineales como lo son matrices dispersas y arboles binarios.
5. Definir algoritmos de búsqueda y recorrido.
6. Familiarizarse con la herramienta Graphviz para la generación de reportes de estructuras gráficos.
7. Familiarizarse con el manejo y escritura de archivos .csv, .html y .scss.
8. Poner en practica el orden lexicográfico.

Definición del problema

Se le solicita al estudiante desarrollar una aplicación de escritorio para la generación de imágenes de pixeles que puedan ser colocadas en paginas web, La necesidad surge debido al costo de almacenamiento en la nube de archivos en distintos formatos (png, jpg, etc.) por lo que se ha decidido renderizar las imágenes mediante los interpretes de HTML y CSS, ya que únicamente se esta escribiendo código esto genera un ahorro debido a que el texto ocupa mucho menos espacio

que una imagen y por lo tanto se genera un ahorro monetario en los sitios para los que se necesiten estas imágenes.

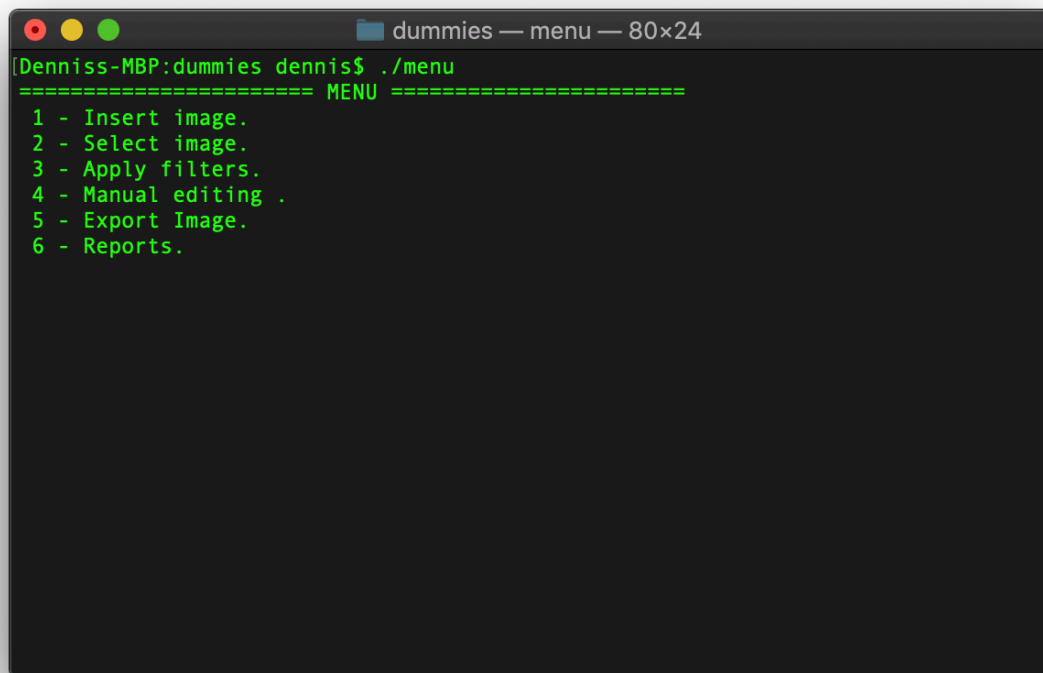
Descripción

Según los requerimientos antes mencionados se deberá de desarrollar el programa “PHOTOGEN++”, el cual será capas de una serie de archivos .csv los cuales representaran capas de una imagen de pixeles, las cuales serán colocadas en distintas estructuras desarrolladas por el estudiante, y por medio de un orden lexicográfico se generaran archivos HTML y SCSS los cuales posteriormente podrán ser insertados en la pagina deseada para poder visualizar la imagen generada por el programa.

Menú Principal

Al comenzar a correr la aplicación se debe de tener un menú principal con las siguientes opciones:

1. Insert image (Insertar imagen)
2. Select image (Seleccionar imagen)
3. Apply filrtes (Aplicar filtros)
4. Manual editing (Edicion Manual)
5. Export image (Exportar imagen)
6. Reports (Reportes)

A terminal window titled "dummies — menu — 80x24" is shown. The prompt is "[Denniss-MBP:dummies dennis\$./menu]". The output displays a menu with six options, each preceded by a number and a hyphen, and followed by a period. The menu is enclosed in a box of equals signs.

```
[Denniss-MBP:dummies dennis$ ./menu]
===== MENU =====
1 - Insert image.
2 - Select image.
3 - Apply filters.
4 - Manual editing .
5 - Export Image.
6 - Reports.
```

1. INSERT IMAGE (INSERTAR IMAGEN)

Para insertar una imagen el programa deberá de solicitar el nombre de un archivo con extensión .csv este será un archivo de “inicial” el cual contendrá un listado ordenado de otros archivos .csv los cuales representan la configuración de la imagen y las capas de la imagen que estamos importando en nuestra aplicación. Cada Imagen importada debe de ser guardada en un **cubo disperso**, el cual es una modificación de una matriz dispersa con la salvedad de que debe de contener profundidad, cada nivel de profundidad representara una capa de la imagen importada.

a. ARCHIVO INICIAL

Este archivo contendrá 2 columnas: **Layer** y **File**, las cuales representan numero de capa y archivo .csv correspondiente, se manejará el numero de capa en orden ascendente, es decir la capa 1 representa la capa mas profunda de la imagen, mientras que la capa con el numero mas alto representa la capa que se encuentra en el tope de la imagen. **(NOTA: la capa 0 esta reservada para el archivo de configuración.)**

	A	B	C
1	Layer	File	
2	0	config.csv	
3	1	body.csv	
4	2	hair.csv	
5	3	shirt.csv	
6	4	dungarees.csv	
7	5	boots.csv	
8			
9			

b. ARCHIVO DE CONFIGURACIÓN

El archivo de configuración contendrá 2 columnas: Config y Value, las cuales representan el atributo en config y el valor de dicho atributo en value, se manejan 4 atributos:

- Image width (ancho de la imagen)
- Image height (alto de la imagen)
- Pixel width (ancho de pixel)
- Pixel height (alto de pixel)

	A	B	C
1	Config	Value	
2	image_width	16	
3	image_height	16	
4	pixel_width	30	
5	pixel_height	30	
6			
7			

C. ARCHIVOS DE CAPAS

Cada archivo de capas debe de contener la información correspondiente a cada capa, en cada celda se contendrá la información del color correspondiente a dicho pixel en formato rgb de la siguiente manera: **R-G-B**, ejemplo: **157-118-230**, en caso la celda se encuentre una x significa que en dicho lugar no se encuentra ningún color o simplemente es **“transparente”**.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
2	x	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
3	x	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
4	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
5	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
6	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
7	x	x	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
8	x	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
9	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
10	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
11	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
12	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
13	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204
14	x	x	x	255-229-204	255-229-204	255-229-204	x	x	255-229-204	255-229-204	255-229-204	x	x	x	x	x
15	x	x	255-229-204	255-229-204	255-229-204	x	x	x	255-229-204	255-229-204	255-229-204	255-229-204	x	x	x	x
16	x	255-229-204	255-229-204	255-229-204	255-229-204	x	x	x	255-229-204	255-229-204	255-229-204	255-229-204	255-229-204	x	x	x
17																

2. SELECT IMAGE (SELECCIONAR IMAGEN)

Cada imagen que sea cargada al sistema debe de ser guardada en una estructura de tipo: **árbol binario (cada nodo del árbol almacena un cubo disperso que representa la imagen)**, esto facilitara el acceso a cada imagen, las imágenes se deben de guardar en orden alfabético. Al momento de seleccionar la opción 2 (Select Image), debe de mostrarse la lista de las imágenes que han sido cargadas al sistema, estas deben de aparecer en orden alfabético, por lo tanto se debe de realizar un recorrido **inorden** al árbol binario y el sistema debe de permitir que se seleccione una, la imagen seleccionada será la imagen para la cual apliquen las funciones del menú 3,4 y 5 respectivamente.

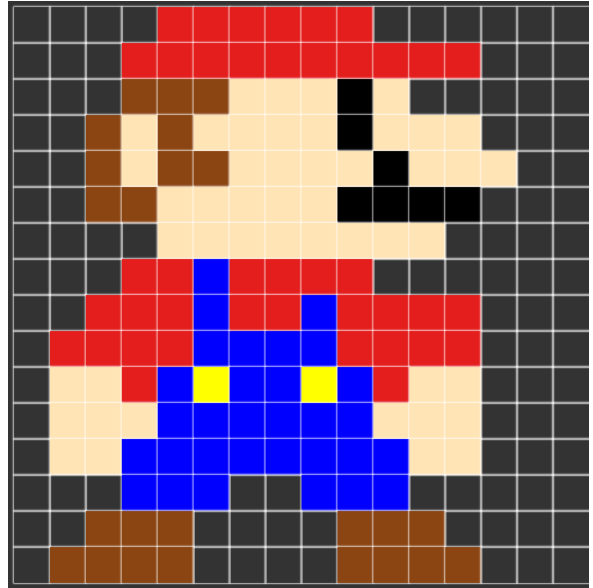
```

dummies — images — 80x24
[Denniss-MBP:dummies dennis$ ./images
===== IMAGES =====
1 - boo
2 - Geoff
3 - Mario1
4 - Mario2
5 - mushroom
6 - Pikachu

```

3. APPLY FILTERS (APLICAR FILTROS)

La aplicación permitirá la aplicación de filtros sobre la imagen seleccionada, dichos filtros únicamente deben de ser aplicados mientras se trabaje en la imagen actual, una vez se cambie de imagen se debe de borrar cualquier filtro aplicado y la imagen debe de ser guardada de la forma original, se contara con una lista de filtros, cada vez que se genere un filtro, el equivalente a este filtro se debe de guardar en una **lista enlazada doble circular (es decir por cada filtro aplicado se deberá de guardar una copia del cubo con el filtro aplicado en cada nodo de la lista)**, cada vez que se cambia de imagen esta lista enlazada doble circular desaparece, sin embargo mientras se este trabajando sobre la misma imagen, se podrá exportar tanto la imagen original, como una de las imágenes que equivalen a los filtros aplicados.

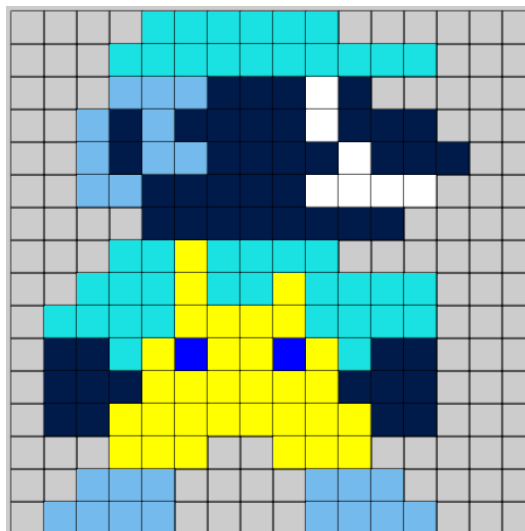


Se debe de tener un menú que incluya todos los filtros a aplicar y cuando se tenga seleccionado uno en específico la aplicación preguntara las siguientes opciones:

- Filtro a imagen completa: Se debera de generar la imagen aplicando el filtro a todas las capas de la misma.
- Filtro a una capa en específico: Se debera de solicitar el nombre de la capa a aplicar el filtro y solo a esta se le debera de aplicar el mismo, todas las demas conservaran el color con el que fueron creadas en su capa.

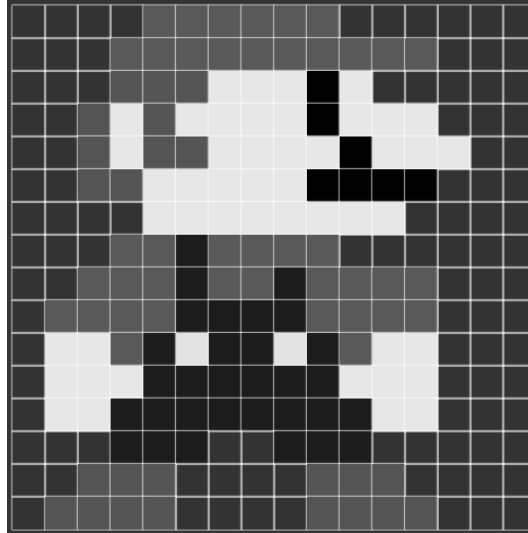
a. NEGATIVE (NEGATIVO)

Este filtro altera la imagen únicamente en sus colores, se debe de calcular el negativo específico de cada píxel y posteriormente pintar dichos pixeles con su negativo equivalente.



b. GRAYSCALE (ESCALA DE GRISES)

Este filtro de igual manera únicamente afectara los colores de la imagen, aplicando un filtro de escala de grises a cada pixel de la imagen, de igual manera posteriormente se debe de pintar cada pixel con su gris equivalente.

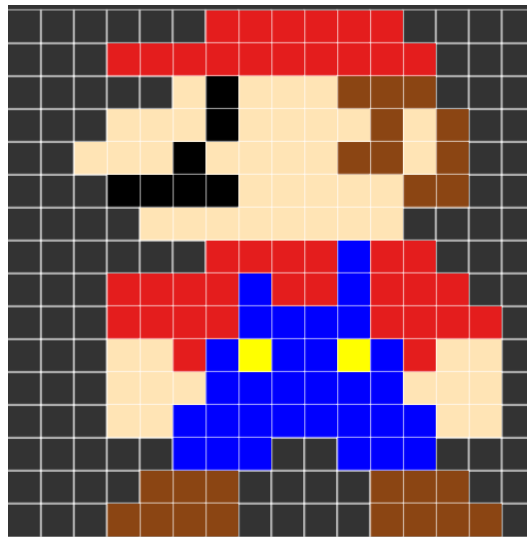


c. MIRROR (ESPEJO)

El filtro de Espejo no modificara los colores de la imagen, sin embargo si la posición de los pixeles de la imagen, por lo tanto se tendrán subsecciones de este filtro, el cual modifica la imagen en distintos ejes.

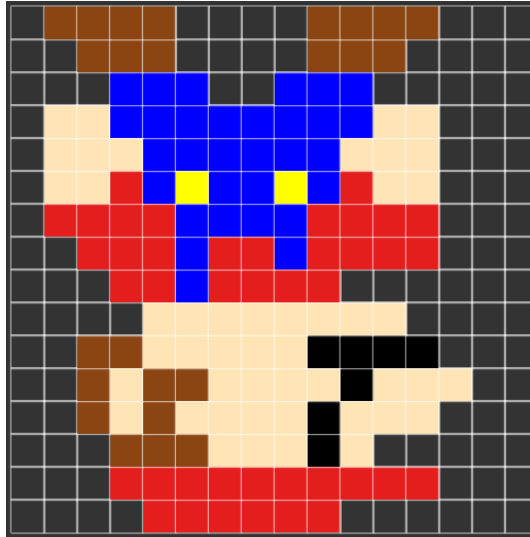
i. X-MIRROR (ESPEJO EN EJE X)

Este filtro modifica la orientación sobre el eje x de la imagen, por lo tanto se generara un espejo sobre el eje x. Este filtro no genera modificación en el color de los pixeles de la imagen.



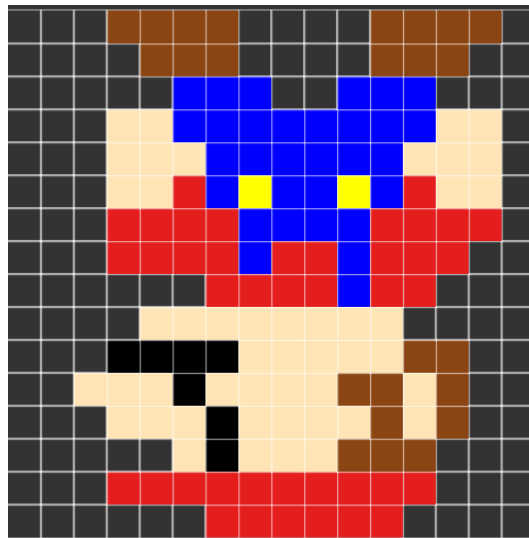
ii. Y-MIRROR (ESPEJO EN EJE Y)

Este filtro modifica la orientación sobre el eje y de la imagen, por lo tanto se genera un espejo sobre el eje y. Este filtro no genera modificación en el color de los pixeles de la imagen.



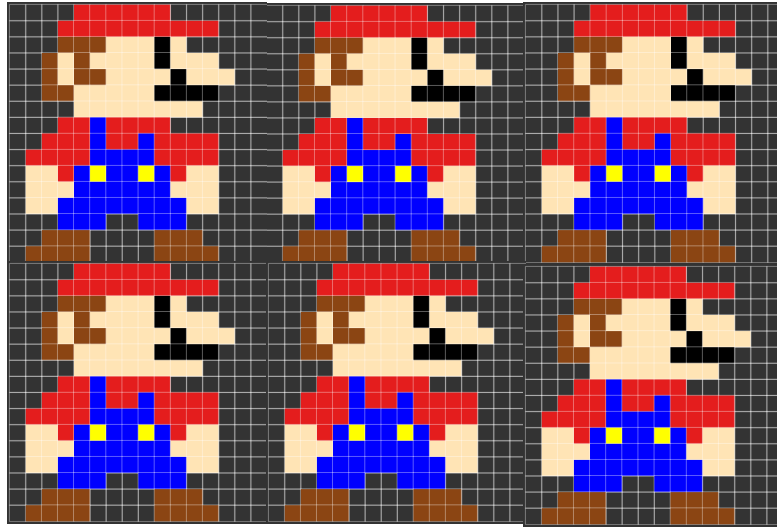
iii. DOUBLE MIRROR (ESPEJO EN AMBOS EJES)

Este filtro modifica la orientación sobre ambos ejes de la imagen, por lo tanto se genera un doble espejo. Este filtro es equivalente a aplicar El filtro espejo de eje x seguido del filtro de espejo de eje y. Este filtro no genera modificación en el color de los pixeles de la imagen.



d. COLLAGE (COLLAGE)

Este filtro genera un collage compuesto por la misma imagen una x cantidad de veces, para poder aplicar este filtro se debe de pedir la cantidad de repeticiones que se desean sobre el eje x y la cantidad de repeticiones que se desean sobre el eje y, y a partir de esta información se debe de generar la imagen. Como ejemplo se puede tomar (x:3,y:2).



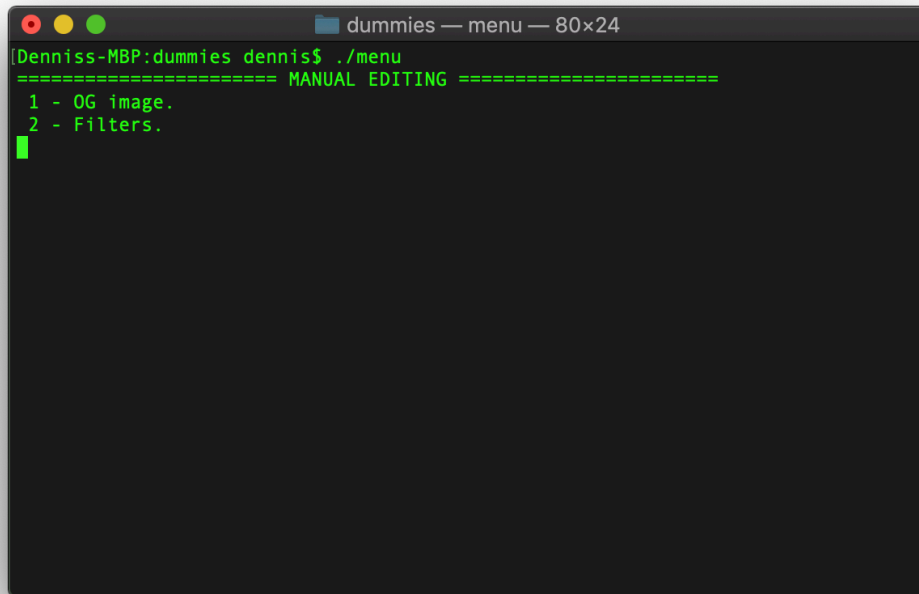
e. MOSAIC (MOSAICO)

Este filtro debe de generar una imagen reemplazando cada pixel de la imagen, por la misma imagen, aplicando un filtro de color de manera que el pixel se vea del color original de dicho pixel, sin embargo sea visible la imagen por la cual dicho pixel esta formado.



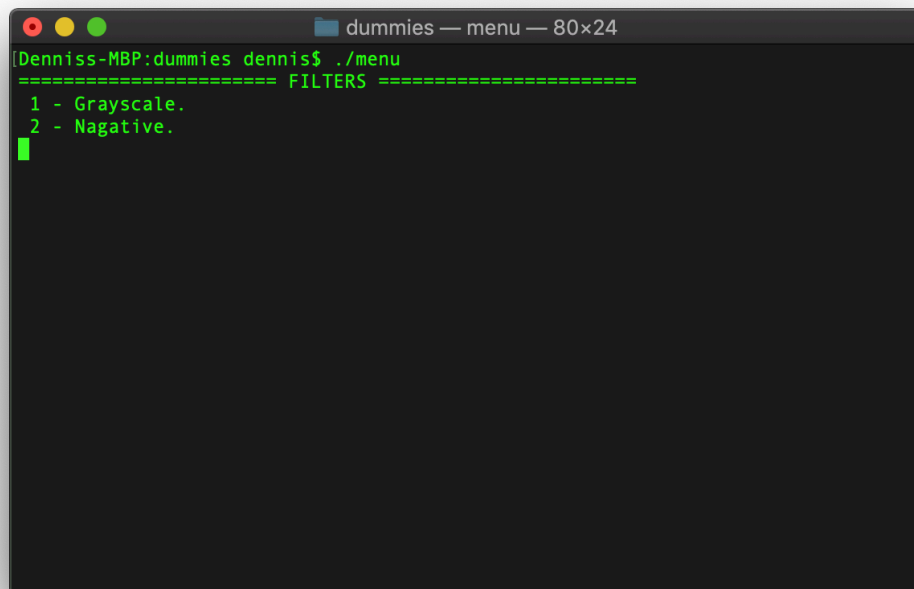
4. MANUAL EDITING (EDICIÓN MANUAL)

Se debe de tener una opción en el menú la cual permita la edición manual de cualquier celda del cubo (original o de uno de los filtros aplicados), esta opción permitirá ingresar coordenadas **x, y**, **nombre de capa** y un **valor de color en formato (RGB)**, el cual permitirá editar individualmente la imagen que se generara posteriormente en la opción 5 (Export image).



```
dummies — menu — 80x24
[Denniss-MBP:dummies dennis$ ./menu
===== MANUAL EDITING =====
1 - OG image.
2 - Filters.
█
```

En caso de seleccionar la opción 2 (filtros), se debe de mostrar la lista de filtros que se le han aplicado a la imagen actual.



```
dummies — menu — 80x24
[Denniss-MBP:dummies dennis$ ./menu
===== FILTERS =====
1 - Grayscale.
2 - Negative.
█
```

Si existieran mas filtros conformarían el resto de opciones (3,4,5, etc.)

5. EXPORT IMAGE (EXPORTAR IMAGEN)

Una vez se hayan o no aplicado los filtros deseados sobre la imagen, se debe de ser capas de exportar la misma para poder ser visualizada mediante su navegador favorito, y debido a que estas imágenes serán utilizadas en distintas paginas web, debe de generarse un archivo .html y un archivo .scss. El proyecto debe de tener una carpeta llamada "Exports", en la cual se exportaran todas las imágenes, cada vez que se exporte una imagen, se deberá de crear un folder con el nombre de la imagen, y dentro de este folder deberán de ir los archivos .html y .scss.

Plantillas HTML y SCSS: <https://github.com/Dennis201503413/PixelArt/tree/master/Template>

a. ARCHIVO HTML

Se proveerá al estudiante una plantilla de tipo HTML, la cual deberá de modificar de acuerdo con la imagen que generara, en términos simples el archivo HTML será un listado de elementos "div", que representaran el arreglo generado mediante el programa c++, Adicionalmente a esto se deberá de generar una stylesheet (hoja de estilos), la cual se encuentre en la misma ruta del archivo HTML generado.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <link rel="stylesheet" href="mario.scss">
5 </head>
6 <body>
7
8 <div class="canvas">
9   <div class="pixel"></div>
10  <div class="pixel"></div>
11  <div class="pixel"></div>
12  <div class="pixel"></div>
13  <div class="pixel"></div>
14  <div class="pixel"></div>
15  <div class="pixel"></div>
16  <div class="pixel"></div>
17  <div class="pixel"></div>
```

b. ARCHIVO SCSS

Se proveerá al estudiante una plantilla de tipo SCSS que deberá de generar, en dicha plantilla deberá de definir el tamaño del lienzo, el tamaño de cada píxel, y deberá de pintar los pixeles uno a uno definidos por los elementos “div” definidos anteriormente en el archivo HTML.

```
1  body {
2    background: #333;
3    height: 100vh;
4    display: flex;
5    justify-content: center;
6    align-items: center;
7  }
8
9  .canvas {
10   width: 480px;
11   height: 480px;
12 }
13
14 .pixel {
15   width: 30px;
16   height: 30px;
17   float: left;
18   box-shadow: 0px 0px 1px #fff;
19 }
20
21 /* Hat */
22 .pixel:nth-child(5),
23 .pixel:nth-child(6),
24 .pixel:nth-child(7),
25 .pixel:nth-child(8),
```

6. REPORTS (REPORTES)

Se debe de implementar una sección de reportes, esta sección contendrá un submenú el cual permitirá generar reportes de las estructuras utilizadas en el proyecto, **TODOS** los reportes deberán de ser implementados por medio de la herramienta **GRAPHVIZ**. Para que cada una de las estructuras tenga validez es necesario tener el reporte correspondiente.

a. IMPORTED IMAGES REPORT

El reporte de imágenes importadas (Imported Image Report) debe de mostrar la estructura del árbol binario en donde se encuentran almacenadas las imágenes que la aplicación esta manejando en el momento. Cada uno de los nodos del árbol debe de mostrar el nombre de la imagen que esta almacenada junto con las dimensiones de esta, y el tamaño de pixel que se maneja.

b. IMAGE LAYER REPORT

Cada una de las imágenes que maneja la aplicación esta compuesta por capas, así que al seleccionar este reporte se debe de desplegar un menú con el nombre de cada una de las imágenes que están cargadas en la aplicación (ver menú Select Image el menú principal) y debe de seleccionar la imagen que se desea. Luego la aplicación debe de pedir el número de capa que se desea graficar, si se selecciona un número de capa inválido debe de volver a pedirla otra vez. Al seleccionar el número de capa, la aplicación debe desplegar la gráfica de la matriz que corresponde a dicha capa, es necesario que la gráfica tenga la forma correspondiente de la matriz para que se tome como válida. También se debe de tener la opción de graficar la imagen completa, esto graficara todas las capas que conforman la imagen (1 capa por cada archivo png o graphviz).

c. LINEAR MATRIX REPORT

Al igual que el reporte anterior (b), se debe de implementar un menú que despliegue el nombre de las imágenes cargadas al sistema y se debe seleccionar la imagen que se desea. Luego de seleccionar la imagen, se debe de pedir el número de capa a linealizar (se deben de tomar las mismas validaciones que el reporte Image Layer) luego de seleccionar la capa se debe de pedir la forma de linealización, por columnas o por filas. El reporte debe de desplegar la representación lineal de la capa seleccionada. Cada nodo de la representación lineal debe de mostrar sus coornadas (x,y) junto con el color que se encuentra en esa coordenada.

d. TRAVERSAL REPORT

Este reporte debe de realizarse sobre el árbol binario que contiene almacenadas las imágenes. Al seleccionar este reporte se debe de desplegar un menú con las siguientes: **Inorder** Traversal, **Postorder** Traversal y **Preorder** Traversal. Cada una de estas opciones representa un recorrido sobre el árbol binario, al selecciona una de esas opciones inmediatamente se debe de desplegar la imagen con el recorrido seleccionado, cada uno de los nodos debe de mostrar el nombre de la imagen.

e. FILTERS REPORT

Para este reporte deben de existir dos opciones: **All filters report** e **Individual Filter Report**.

- i. **All filters report (Reporte de todos los filtros):** Al momento de trabajar con filtros sobre una imagen, se genera la una lista enlazada doble circular, al seleccionar este reporte se debe de desplegar la imagen que represente la lista en ese momento, es decir que si no se han aplicado filtros a la imagen, no debe de desplegar nada y mostrar

que no se han aplicado filtros (un mensaje por consola). Cada nodo debe de mostrar el nombre del filtro aplicado.

- ii. **Individual filter report (Reporte de filtro individual):** Este reporte debe de tener un menú que debe desplegar el nombre de los filtros que existe en la lista doblemente enlazada circular, el usuario debe seleccionar uno de ellos para realizar el reporte sobre el filtro seleccionado. Este reporte es el mismo que el Image Layer Report (b) pero sobre el cubo que representa el filtro seleccionado.

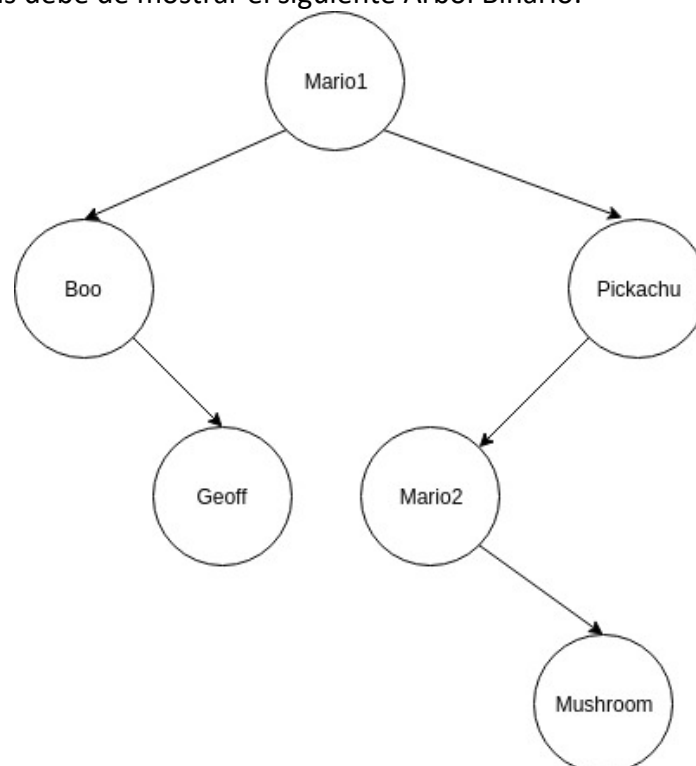
Ejemplo de Reportes:

Para este ejemplo se considera que se han cargado las siguientes imágenes al sistema (Se han cargado al sistema en ese orden específico):

1. Mario1
2. Pickachu
3. Boo
4. Geoff
5. Mario2
6. Mushroom

Imported Images Report (Ejemplo)

Ya que se han ingresado las imágenes anteriormente mencionadas, la salida del reporte de imágenes importadas debe de mostrar el siguiente Árbol Binario:



La imagen es ilustrativa para el ejemplo, el reporte de salida debe de mostrar el nombre de la imagen junto a sus dimensiones y tamaño de cada pixel.

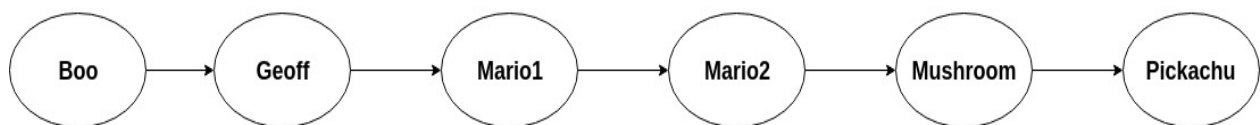
Traversal Report

Considerando que el sistema tiene el Árbol Binario descrito en el ejemplo anterior, se desea los reportes de cada uno de los recorridos del árbol.

Inorder Traversal

Debemos de recordar que el recorrido inorden visita: subárbol izquierdo, raíz, subárbol derecho

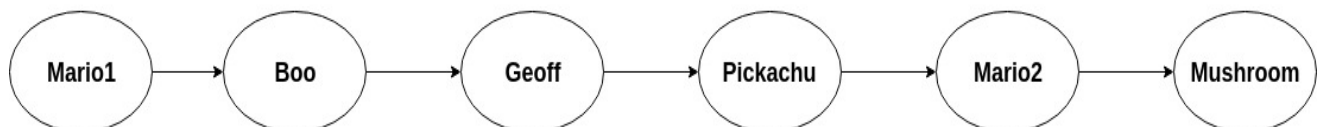
Inorder Traversal



Preorder Traversal

Debemos de recordar el recorrido preorden visita: raíz, subárbol izquierdo, subárbol derecho

Preorder Traversal



Posorder Traversal

Debemos de recordar que el recorrido posorden visita: subárbol izquierdo, subárbol derecho, raíz

Posorder Traversal

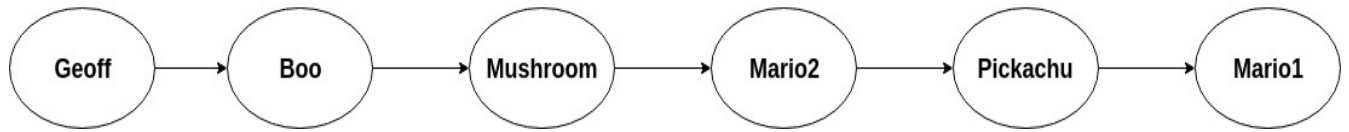


Image Layer Report

Consideremos que se agrega otra imagen al sistema, esta imagen representa un letra “R” dibujada en 2 capas (Este reporte es sobre el cubo original de la imagen).

• CAPA 1:

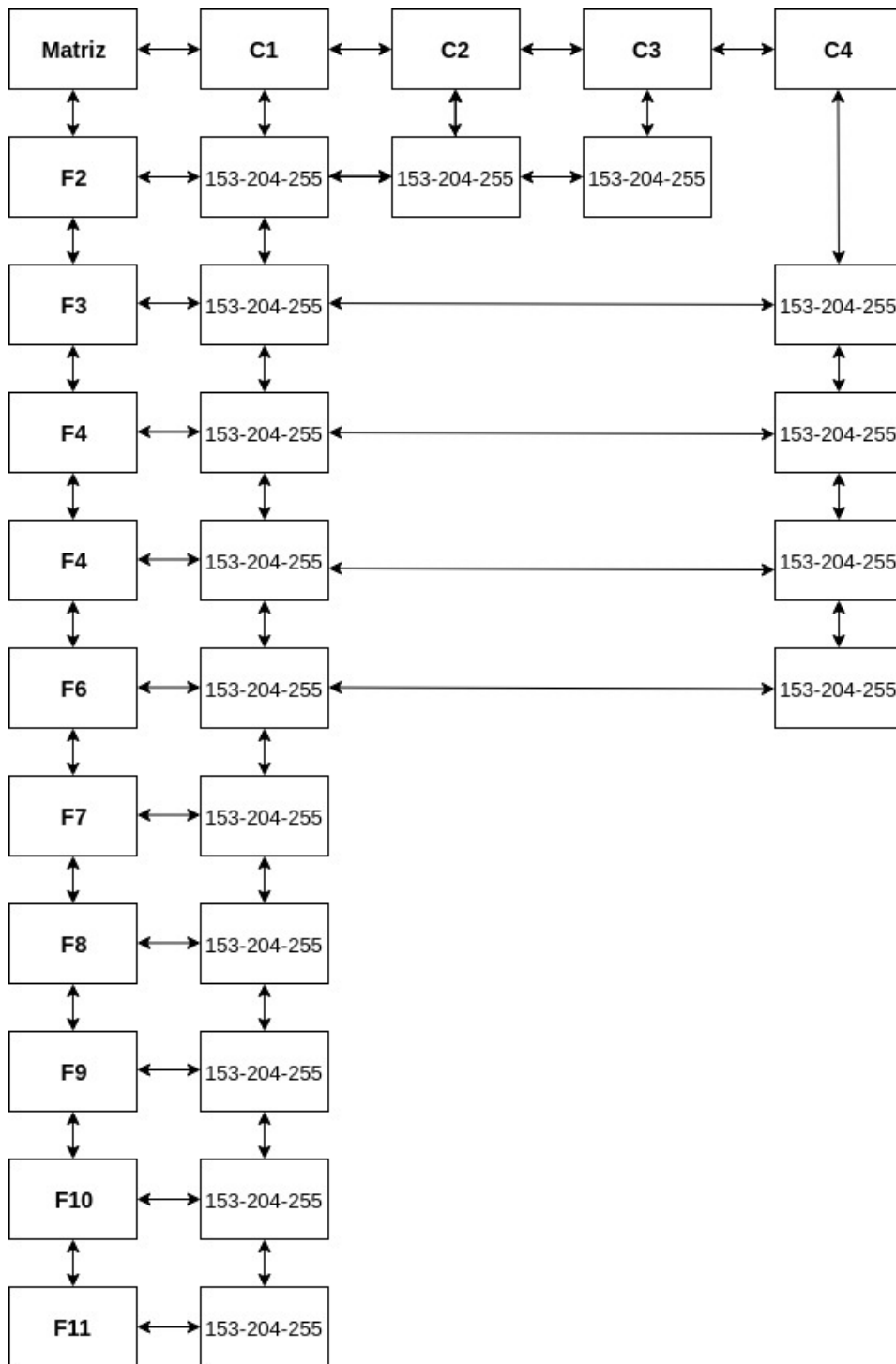
	A	B	C	D	E	F	
1	x	x	x	x	x	x	
2	x	x	x	x	x	x	
3	x	153-204-255	153-204-255	153-204-255	x	x	
4	x	153-204-255	x	x	153-204-255	x	
5	x	153-204-255	x	x	153-204-255	x	
6	x	153-204-255	x	x	153-204-255	x	
7	x	153-204-255	x	x	153-204-255	x	
8	x	153-204-255	x	x	x	x	
9	x	153-204-255	x	x	x	x	
10	x	153-204-255	x	x	x	x	
11	x	153-204-255	x	x	x	x	
12	x	153-204-255	x	x	x	x	
13	x	x	x	x	x	x	
14	x	x	x	x	x	x	
15	x	x	x	x	x	x	
16							

• CAPA 2:

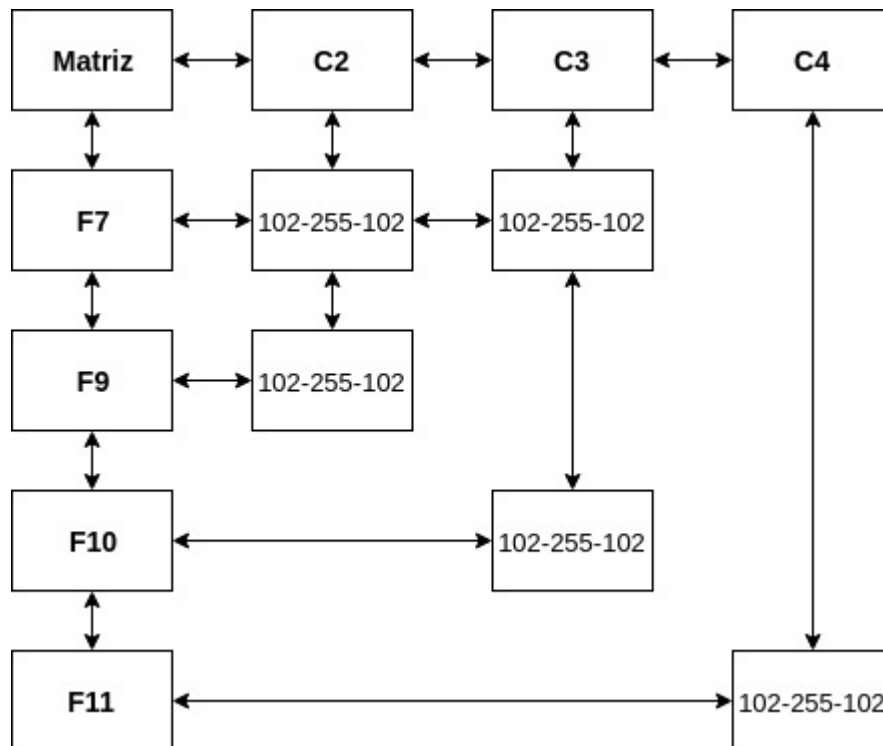
	A	B	C	D	E	F	
1	x	x	x	x	x	x	
2	x	x	x	x	x	x	
3	x	x	x	x	x	x	
4	x	x	x	x	x	x	
5	x	x	x	x	x	x	
6	x	x	x	x	x	x	
7	x	x	x	x	x	x	
8	x	x	102-255-102	102-255-102	x	x	
9	x	x	x	x	x	x	
10	x	x	102-255-102	x	x	x	
11	x	x	x	102-255-102	x	x	
12	x	x	x	x	102-255-102	x	
13	x	x	x	x	x	x	
14	x	x	x	x	x	x	
15	x	x	x	x	x	x	
16							

Con esta imagen definida en 2 capas se desea obtener los reportes de cada una de las capas:

- Gráfica Capa 1



- Gráfica Capa 2



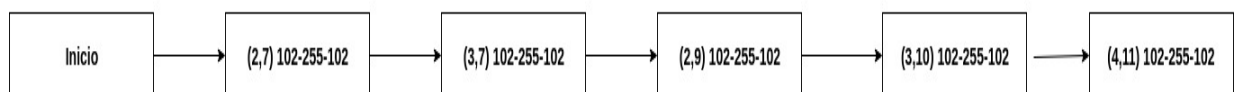
Cabe resaltar que la matriz debe de poder visualizarse correctamente, de lo contrario no tendrá validez el reporte.

Linear Matrix Report

Tomando el ejemplo del reporte Image Layer, si deseamos visualizar la linealización de la capa 2:

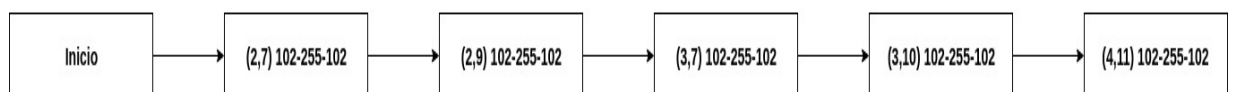
- Por Filas:

Linealización Por Filas:Capa 2



- Por Columnas:

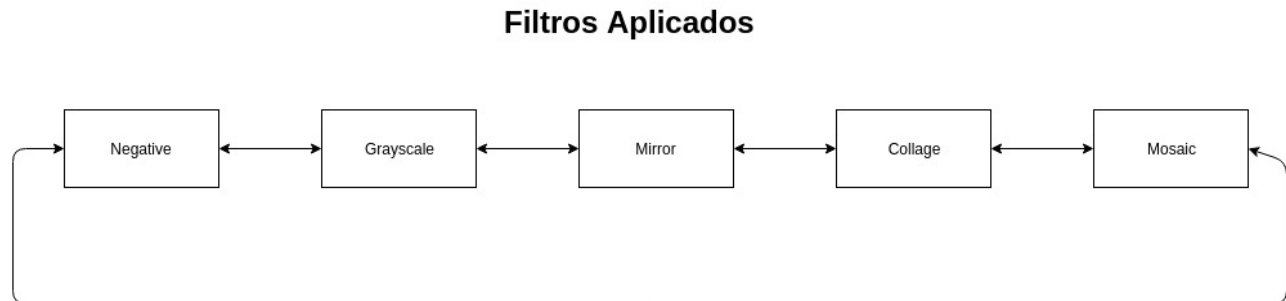
Linealización Por Columnas:Capa 2



Filters Report

All Filters Report

El reporte debe de mostrar los filtros que se han aplicado a la imagen con la que se esta trabajando actualmente:



Individual Filter Report

Antes de generar este reporte, la aplicación ya debió de mostrar los filtros existentes sobre la imagen como un menú para que el usuario seleccione el filtro que desea. Luego de seleccionar el filtro (cubo que representa el filtro) se debe de poder seleccionar la capa que se desea reportar y luego generar la imagen del filtro. Este reporte es generar el **Image Layer Report** pero sobre el cubo del filtro.

Es necesario resaltar que para optar a la calificación de cada una de las estructuras el reporte es necesario.

Restricciones

Las estructuras deben de ser desarrolladas por los estudiantes sin el uso de ninguna librería o estructura predefinida en el lenguaje a utilizar. Los reportes son esenciales para verificar si se trabajaron correctamente las estructuras solicitadas. Si no se tiene el reporte de alguna estructura se anularan los puntos que tengan relación tanto al reporte como a la estructura en cuestión.

Penalizaciones

1. Falta de seguimiento de desarrollo continuo por medio Github tendrá una penalización del 10% (mínimo 2 commits por semana).
2. Falta de seguimiento de instrucciones conforme al método de entrega (nombre del repositorio) tendrá una penalización del 5%.

3. Falta de puntualidad conforme a la entrega tendrá una penalización de la siguiente manera:
 - a. 0-6 horas – 20%.
 - b. 6-12 horas – 40%
 - c. 12-18 horas 60%
 - d. 18-24 horas – 80%.
 - e. Pasados 24 horas tendrá una nota de 0 y no se calificara.

Observaciones

1. Lenguaje a utilizar: **C++**
2. Herramienta para desarrollo de reportes gráficos: **Graphviz**.
3. Forma de despliegue de las imágenes generadas: **HTML y SCSS**
4. La aplicación debe de ser capaz de generar y abrir con un visor de imágenes predeterminado las imágenes generadas con Graphviz.
5. La aplicación debe de ser capaz de generar y abrir los HTML generados con el navegador predeterminado del sistema.
6. La entrega se realizará por medio de: **Github**, cada estudiante deberá crear un repositorio con el nombre: **EDD_2S2019_PY1_#carnet**, y agregar a su auxiliar correspondiente como colaborador del mismo, para poder analizar su progreso y finalmente a partir del mismo repositorio realizar la calificación correspondiente.
Dennis Masaya: <https://github.com/Dennis201503413>
Ricardo Cutz: <https://github.com/ricardcutzh>
Antonio Hernandez: <https://github.com/fernando29hernandez>
7. Además de tener a su auxiliar como colaborador del repositorio para tener un control y orden de las personas que entreguen deberán de colocar el Link de su repositorio en la Tarea que cada auxiliar asignara en su classroom correspondiente.
8. Fecha y hora de entrega: **Sábado 14 de Septiembre, a las 23:59 horas**.
9. **Copias serán penalizadas con una nota de 0 y castigadas según lo indique el reglamento.**