Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Monterrey

Propuesta Compilador
"QUETZAL"

Diseño de compiladores (Ago 19 Gpo 2)

Ing. Héctor Gibrán Ceballo Cancino
Ing. Elda Guadalupe Quiroga González

Lizzie Marielle Guajardo Mozo
A00818258

Alejandro González Valles
A00818616

Monterrey, N.L. México 30 de septiembre del 2019

Visión del proyecto

Nuestro proyecto tiene como visión utilizar todo nuestro conocimiento obtenido a través de nuestra carrera profesional para desarrollar una herramienta de programación que sea capaz de realizar las tareas básicas de un lenguaje de programación vinculado a un paquete de utilidades para el manejo y manipulación de colores en archivos de imagen.

Objetivo del lenguaje y área de aplicación

El objetivo del lenguaje Quetzal es proporcionar al usuario una forma sencilla y legible de programar imperativamente. Quetzal permite el manejo de tipos de datos básicos como enteros, números flotantes y strings, también de un tipo de dato especial para color en formato hexadecimal que, en conjunto con funciones especiales, permitirá manipular los pixeles de un archivo de imagen para producir versiones alternas según filtros de color, escala de grises o modificación directa de la matriz con el conjunto de valores hexadecimales.

La herramienta busca ser de utilidad para estudiantes principiantes de programación, en particular educación media y media superior, que se estén introduciendo a la rama de gráficas computacionales y que el manejo de la herramienta no se vea comprometido aun si no cuentan con conocimiento previo sobre teoría de color.

Requerimientos del proyecto

1. Elementos básicos

func	string	pixels	if
bool	char	colorFilter	return
float	color	bool	null
int	grayscale	while	else
print	true	false	\n
read	openImg	saveImg	define

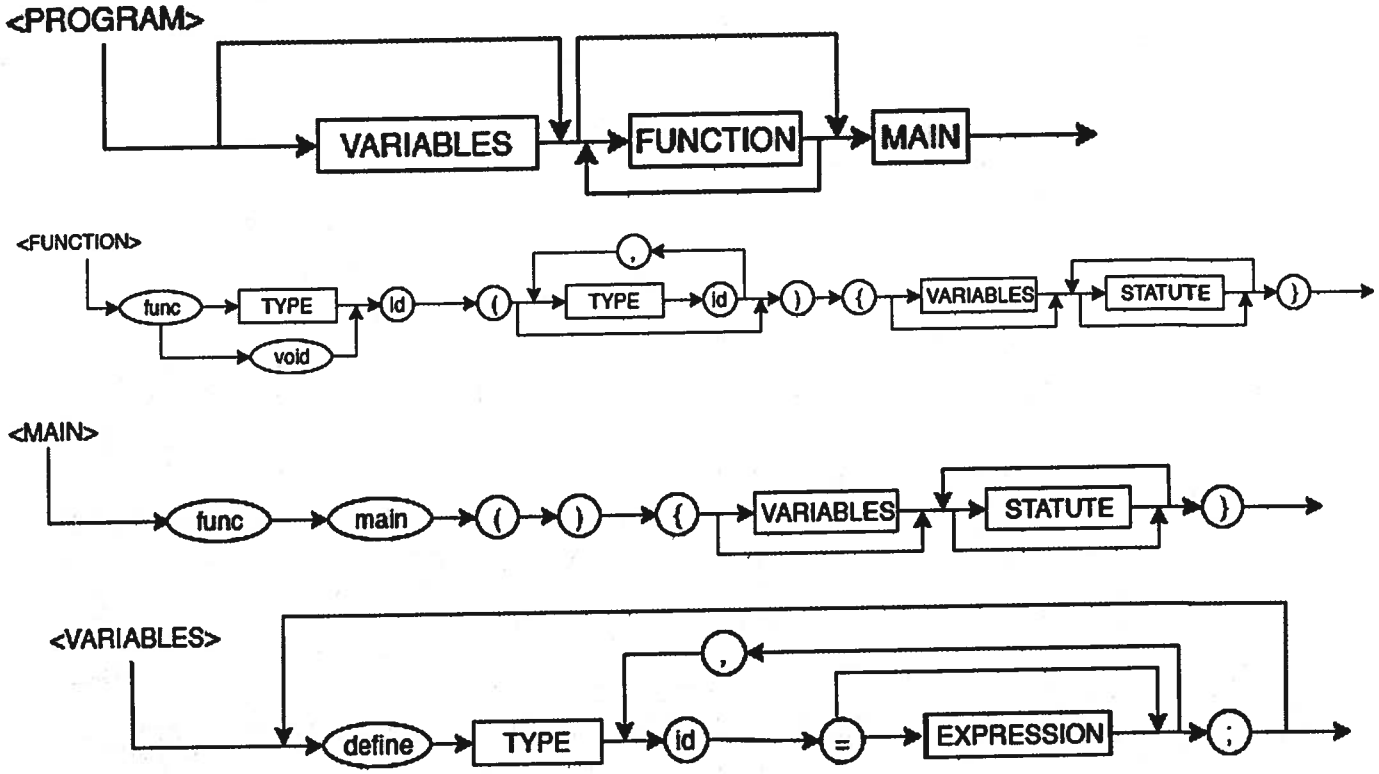
Reserved color names:

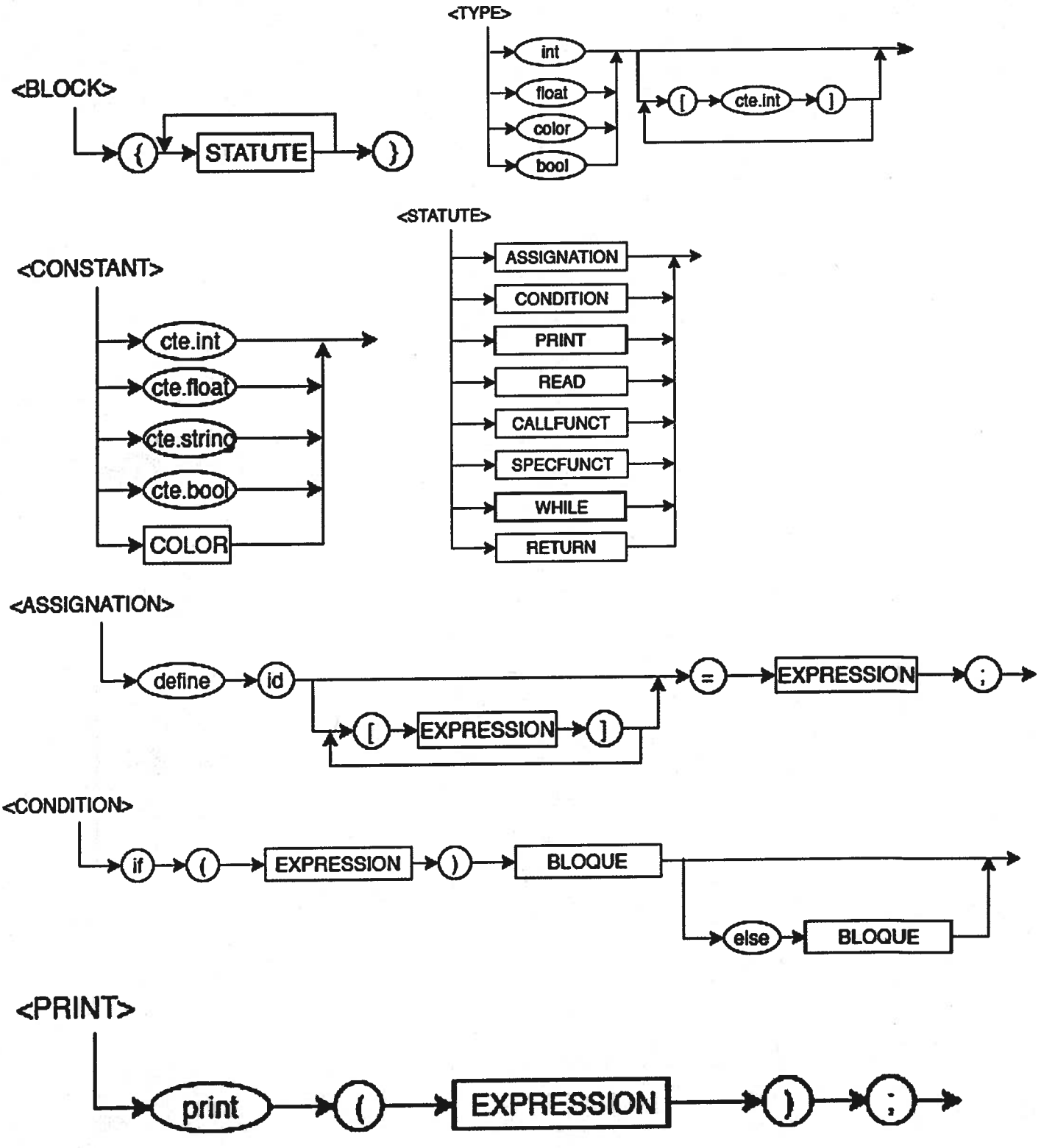
White	Red	Lime	Blue
Silver	Maroon	Green	Navy
Gray	Yellow	Aqua	Fuchsia
Black	Olive	Teal	Purple

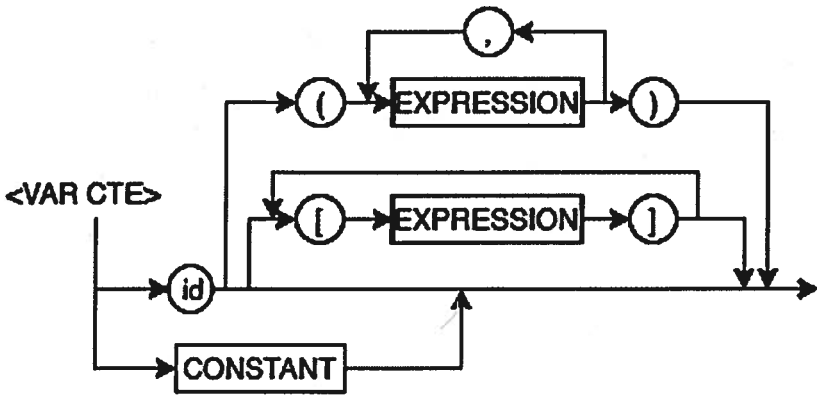
Símbolos

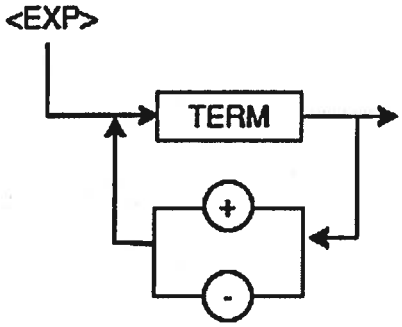
- ||,&&
- +,-,*,/
- <,>,=,!
- . , : ; " '
- {,},(,),[,]

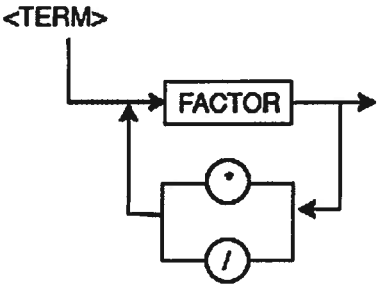
2. Diagramas de Sintaxis

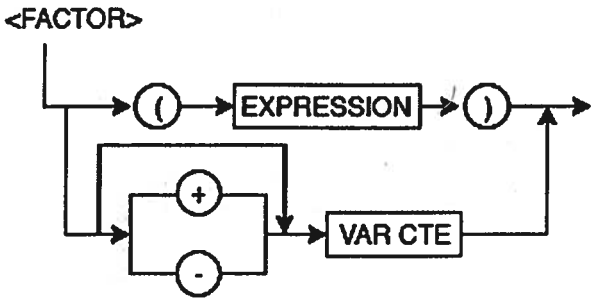


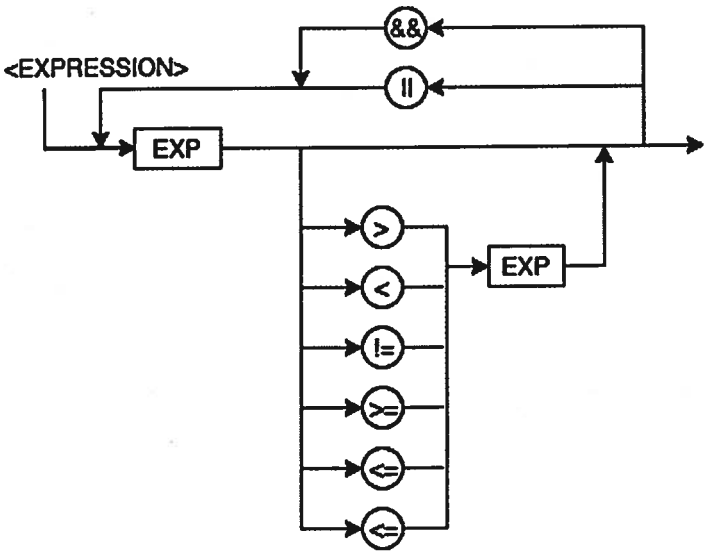

<BLOCK>
{
STATUTE
}
<TYPE>
int
float
color
bool
[
cte.int
]
<CONSTANT>
cte.int
cte.float
cte.string
cte.bool
COLOR
<STATUTE>
ASSIGNATION
CONDITION
PRINT
READ
CALLFUNCT
SPECFUNCT
WHILE
RETURN
<ASSIGNATION>
define
id
=
EXPRESSION
;
[
EXPRESSION
]
<CONDITION>
if
(
EXPRESSION
)
BLOQUE
else
BLOQUE
<PRINT>
print
(
EXPRESSION
)
;

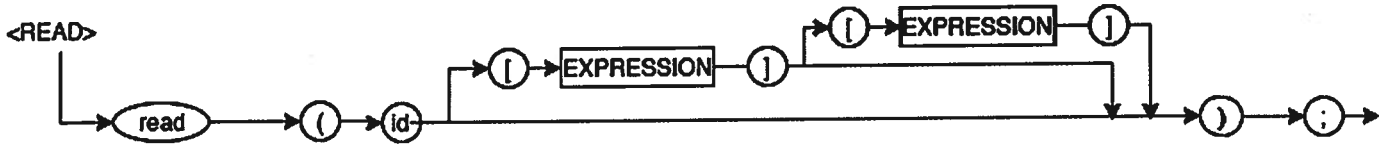

<VAR CTE>
id
CONSTANT
(
EXPRESSION
)
,
[
EXPRESSION
]

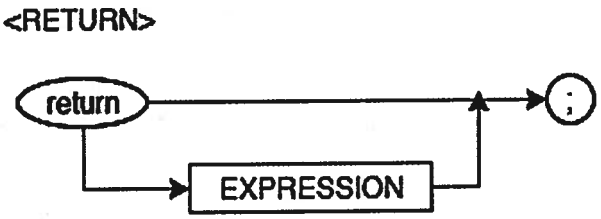

<EXP>
TERM
+
-

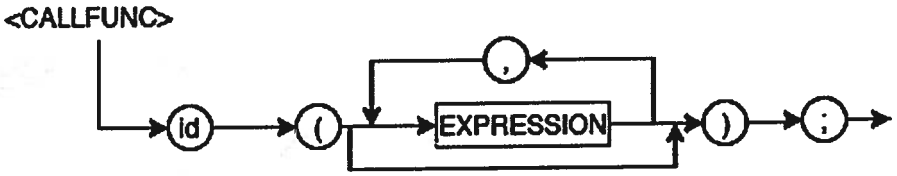

<TERM>
FACTOR
*
/


<FACTOR>
(
EXPRESSION
)
+
-
VAR CTE


<EXPRESSION>
&&
||
EXP
>
<
!=
>=
<=
<=
EXP

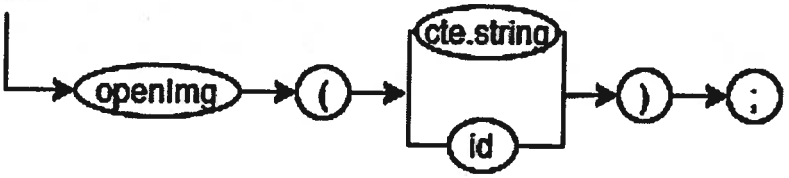

<READ>
read
(
id
[
EXPRESSION
]
[
EXPRESSION
]
)
;


<RETURN>
return
;
EXPRESSION

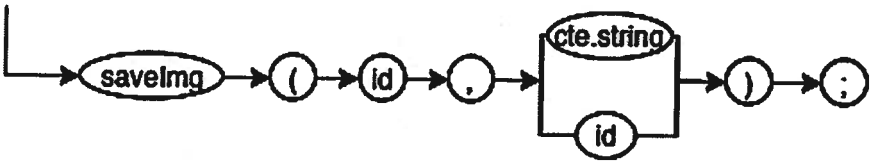

<CALLFUNC>
,
id
(
EXPRESSION
)
;


<WHILE>
while
(
EXPRESSION
)
BLOCK

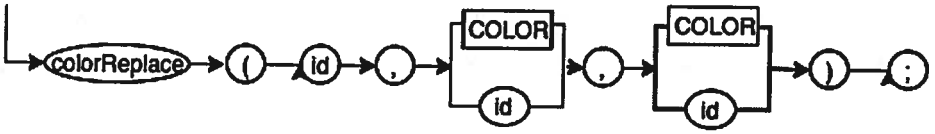
<OPENIMG>



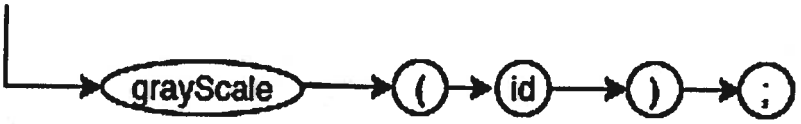
<SAVEIMG>

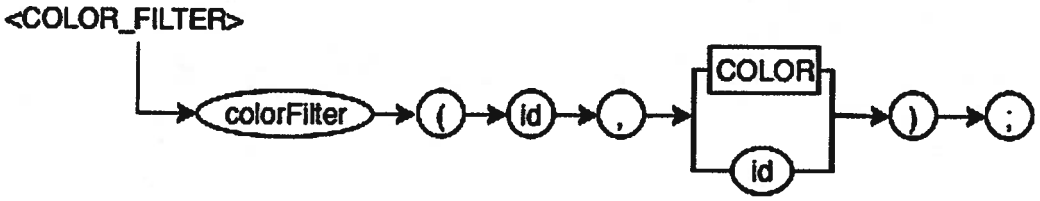


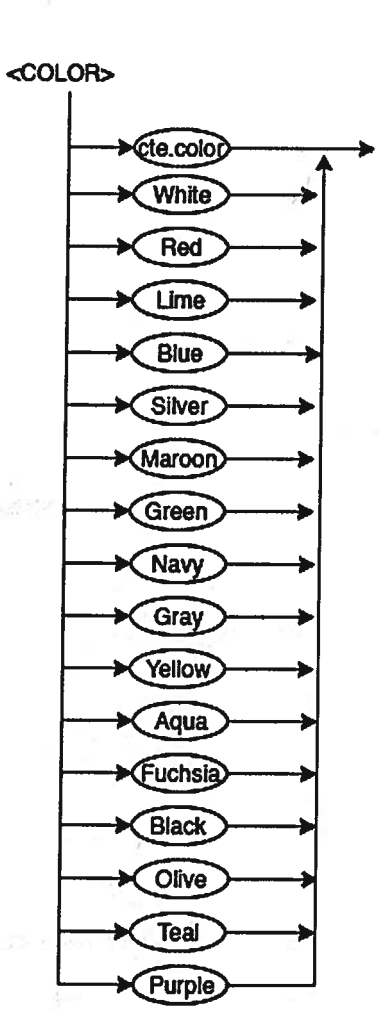
<COLOR_REPLACE>

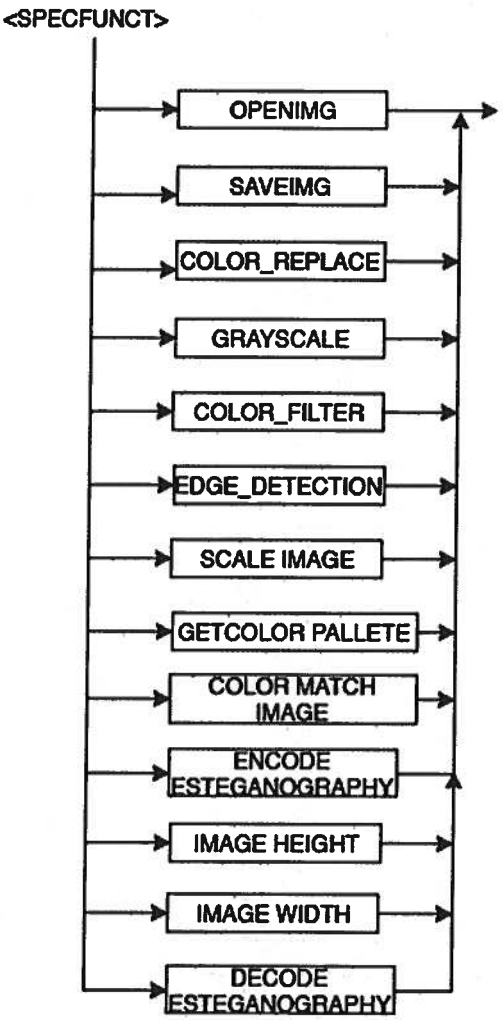


<GRAYSCALE>




<COLOR_FILTER>
colorFilter
(
id
,
COLOR
id
)
;


<COLOR>
cte.color
White
Red
Lime
Blue
Silver
Maroon
Green
Navy
Gray
Yellow
Aqua
Fuchsia
Black
Olive
Teal
Purple


<SPECFUNCT>
OPENIMG
SAVEIMG
COLOR_REPLACE
GRAYSCALE
COLOR_FILTER
EDGE_DETECTION
SCALE IMAGE
GETCOLOR PALLETE
COLOR MATCH IMAGE
ENCODE ESTEGANOGRAPHY
IMAGE HEIGHT
IMAGE WIDTH
DECODE ESTEGANOGRAPHY

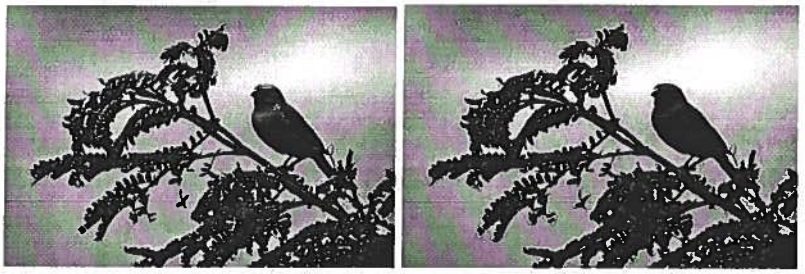
3. Principales características semánticas
    - No se pueden utilizar variables o funciones si no han sido declaradas previamente
    - Los tipos de datos deben de coincidir al momento de: asignarle valor a una variable, pasar parámetros a una función, realizar operaciones entre tipos de datos.
    - En matrices las todas las filas deben compartir el mismo número de columnas y viceversa.
    - En arreglos y matrices solamente se permite un solo tipo de dato dentro de la estructura.
    - La función Main es obligatoria y es la primera función en ejecutarse.
    - Las funciones saveImg, openImg, getImgWidth y getImgHeight deben de recibir una variable string y esta debe a hacer referencia a un directorio de archivo.
    - La función openImg retorna una matriz de colores.
4. Descripción de las funciones especiales
    1. **print:** Imprime en la consola los valores especificados
    2. **read:** Lee un valor desde la consola y se puede guardar en una variable
    3. **getImageHeight:** Recibe como parámetro de la ubicación de un archivo de una imagen y regresa un valor entero que representa la altura de la imagen(cantidad de pixeles en y)
        a. define iHeight;
        b. iHeight = getImageHeight("C:\Users\lizzi\Desktop\Bird.JPG");
        c. *//iHeight es igual a 258*
        d. 
    4. **getImageWidth:** Recibe como parámetro de la ubicación de un archivo de una imagen y regresa un valor entero que representa la anchura de la imagen(cantidad de pixeles en x)
        a. define iWidth;
        b. iWidth = getImageHeight("C:\Users\lizzi\Desktop\Bird.JPG");
        c. *//iWidth es igual a 388*

d. 

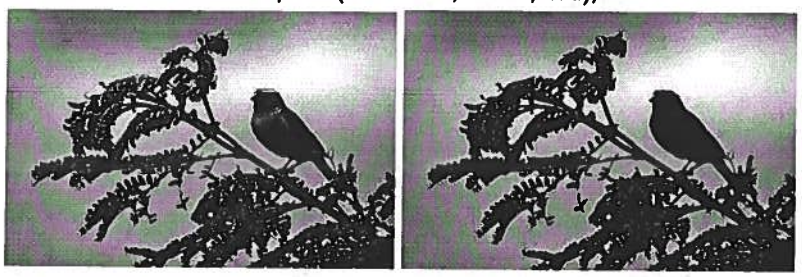
e.

5. **openImg:** Recibe como parámetro la ubicación de un archivo de una imagen y lo guarda en una variable de matriz de colores.
   a. define color birdPhoto[iWidth][iHeight];
   b. birdPhoto = openImg("C:\Users\lizzi\Desktop\Bird.JPG");
   c. *//Si ya conozco las dimensiones de la imagen puede inicializar el arreglo de colores con el tamaño indicado*
   d. define color colorMatrix[10][13];
   e. colorMatrix = openImg("C:\Users\lizzi\Desktop\Example.JPG");
   f. 
   g. *//colorMatrix=*

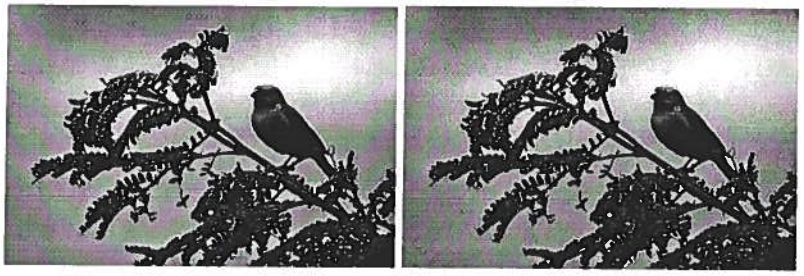
{{#FFFFFF,#FFFFFF,#000000,#000000,#000000,#FFFFFF,#FFFFFF,#FFFFFF,#000000,#000000,#000000,#FFFFFF,#FFFFFF},
{#FFFFFF,#000000,#BF1E2E,#BF1E2E,#BF1E2E,#000000,#FFFFFF,#000000,#BF1E2E,#BF1E2E,#BF1E2E,#000000,#FFFFFF},
{#000000,#BF1E2E,#FFFFFF,#FFFFFF,#BF1E2E,#BF1E2E,#000000,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#000000},
{#000000,#BF1E2E,#FFFFFF,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#000000},
{#000000,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#000000},
{#FFFFFF,#000000,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#000000,#FFFFFF},
{#FFFFFF,#FFFFFF,#000000,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#000000,#FFFFFF,#FFFFFF},
{#FFFFFF,#FFFFFF,#FFFFFF,#000000,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#BF1E2E,#000000,#FFFFFF,#FFFFFF,#FFFFFF},
{#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#000000,#BF1E2E,#BF1E2E,#BF1E2E,#000000,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF},
{#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#000000,#BF1E2E,#000000,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF},
{#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#000000,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF,#FFFFFF}}

6. **saveImg:** Recibe como parámetro una matriz de colores y lo guarda como imagen según la ruta de archivo especificada.
   a. *Ver ejemplos de saveImg debajo.
7. **grayscale:** Recibe una matriz de colores y la convierte la imagen a escala de grises.
   a. newBirdPhoto = grayScale(birdPhoto);
   b. 
   c. saveImg(newBirdPhoto,"C:\Users\lizzi\Desktop\GrayBird.JPG")
8. **color_replace:** Toma una matriz de colores y recibe dos colores como parámetros, el primero será recoloreado por el segundo.

a. newBirdPhoto = colorReplace(birdPhoto,Yellow,Red);

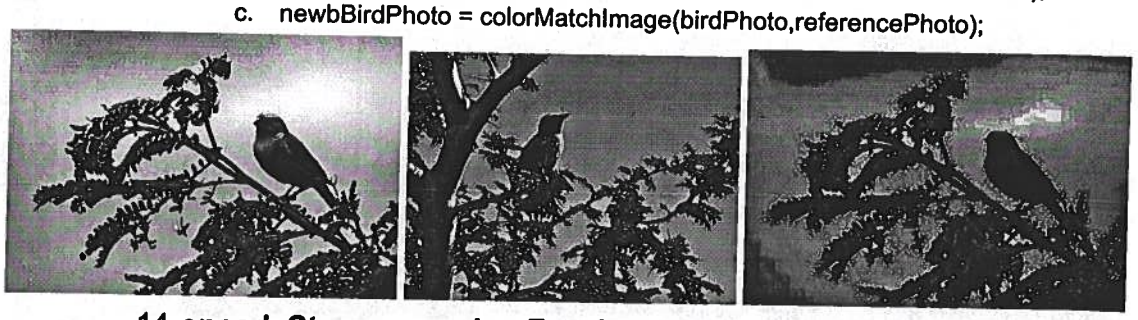


b. saveImg(newBirdPhoto,"C:\Users\lizzi\Desktop\RedBird.JPG")

9. **color_filter:** Recibe una matriz de colores y un color. Esta función convierte en escala de gris todos los colores de la matriz que no sean iguales al segundo parámetro.
   a. newbBirdPhoto = colorFilter(birdPhoto,Blue);
   b. 
   c. saveImg(newBirdPhoto,"C:\Users\lizzi\Desktop\BlueSkyBird.JPG")
10. **edgeDetection:** Toma una matriz de colores. Esta función detecta automáticamente los bordes de los principales elementos de una imagen y los delinea.
   a. newbBirdPhoto = edgeDetection(birdPhoto);
   b. 
   c. saveImg(newBirdPhoto,"C:\Users\lizzi\Desktop\BlueSkyBird.JPG")
11. **scaleImage:** Toma como parámetro una matriz de colores y dos números flotantes. La imagen será escalada en x según el valor del segundo parámetro (Ej. 2 = 200%) y en y según el valor del tercer parámetro. Un número negativo escalará la imagen según la magnitud y la invertirá de forma horizontal o vertical según sea el caso.
   a. newbBirdPhoto = scaleImage(birdPhoto,-2,1);

b. saveImg(newBirdPhoto,"C:\Users\lizzi\Desktop\BlueSkyBird.JPG")

12. **getColorPalette:** Función que recibe como parámetro una matriz de colores y un número entero *n*. Se regresa un arreglo con *n* colores representando los *n* colores predominantes en la imagen
    a. birdPhotoPalette = getColorPalette(birdPhoto,4);
    b. // birdPhotoPalette = [9BD5F8,D9ECFB,5E671A,9C9B55]
    c. 
13. **colorMatchImage:** Función que recibe como parámetro dos matrices de colores, siendo la primer matriz la imagen fuente y la segunda la imagen de referencia. Esta función manipula los píxeles de la imagen fuente de modo que el histograma de pixeles concuerde con el histograma de la imagen de referencia.
    a. define color referencePhoto[960][720];
    b. referencePhoto = openImg("C:\Users\lizzi\Desktop\Reference.JPG");
    c. newbBirdPhoto = colorMatchImage(birdPhoto,referencePhoto);



14. **encodeSteganography:** Esta función se basa en la técnica de Esteganografía, la cual oculta un mensaje dentro de una imagen. La función recibe una imagen y un mensaje de texto y retorna la imagen con el código guardado. A simple vista la imagen no parece ser diferente a la original.
15. **decodeSteganography:** Esta función recibe una imagen modificada con esteganografía para obtener el mensaje encriptado que contenga. Recibe una matriz de colores y imprime el mensaje oculto

5. Tipos de Datos en el lenguaje
    - Entero
    - Flotante
    - String

    - Bool
    - Color: Valor hexadecimal de 0 a FFFFFF

Lenguaje y SO de Desarrollo

- Lenguaje: Python
- SO de Desarrollo: Windows 10
- Herramienta: ANTLR

Bibliografía

- **https://scikit-image.org/**
- **https://github.com/antlr/antlr4/blob/master/doc/index.md**
- **https://numpy.org/**