

ALGORITMOS DE COMPUTACION GRAFICA

Clase 02 Primitivas Bidimensionales

Objetivo:

Primitivas bidimensionales.

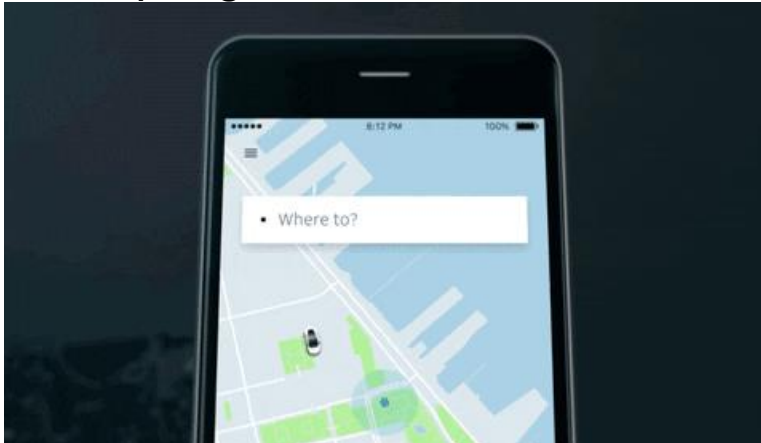
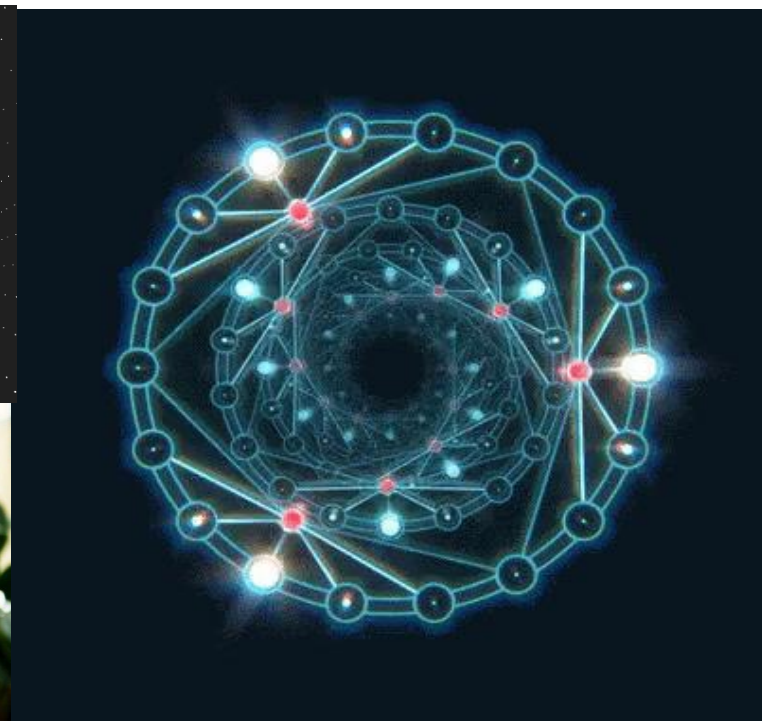
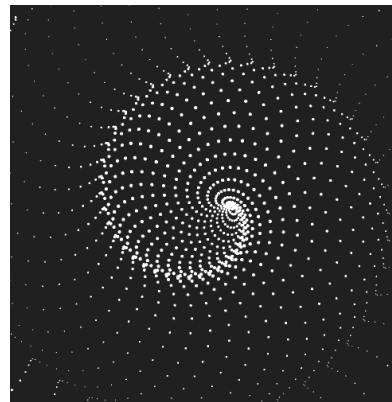
Algoritmo de línea bresenham y DDA.

Implementación de algoritmos de línea.

Guía de laboratorio:

Implementación usando Python con opengl.

Lunes 08:00 - 11:20



jreategui@untels.edu.pe

MBA-ISO 27001-ISO 9001-ISO 22301

«La simplicidad es la máxima sofisticación.».

Leonardo Da Vinci

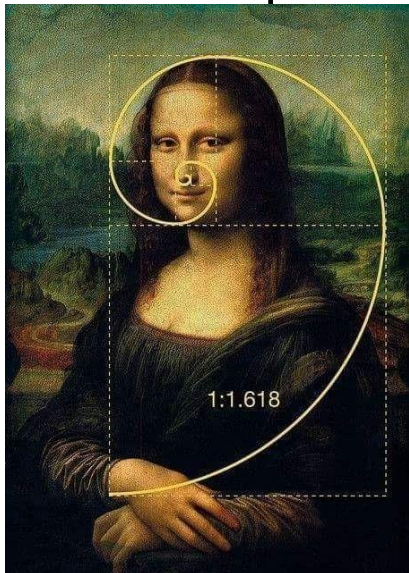
Primitivas Bidimensionales

Un conjunto común de primitivas bidimensionales incluye **líneas**, **puntos** y **polígonos**, aunque algunas personas prefieren considerar primitivas de triángulos, **porque cada polígono se puede construir a partir de triángulos**.

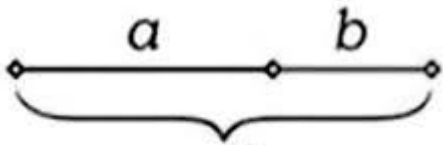
Todos los demás elementos gráficos se construyen a partir de estas primitivas.

En tres dimensiones, los triángulos o polígonos colocados en un espacio tridimensional se pueden utilizar como primitivas para modelar formas 3D más complejas.

En algunos casos, las curvas (como curvas de Bézier, círculos , etc.) pueden considerarse primitivas; en otros casos, las curvas son formas complejas creadas a partir de muchas formas primitivas y rectas.



$$\varphi = \frac{1 + \sqrt{5}}{2} \approx 1.6180339887$$


$$\frac{a+b}{a} = \frac{a}{b} = \varphi$$

Primitivas Bidimensionales

Primitivas comunes:

El conjunto de primitivas geométricas se basa en la dimensión de la forma que se representa:

- Punto (dimensión 0), una única ubicación sin altura, ancho ni profundidad.
- Línea o curva (unidimensional), que tiene longitud pero no ancho, aunque una característica lineal puede curvarse a través de un espacio de mayor dimensión.
- Región plana (bidimensional), que tiene longitud y ancho.
- Región volumétrica (tridimensional), que tiene longitud, ancho y profundidad.
- En SIG, la superficie del terreno a menudo se denomina coloquialmente como "2 1/2 dimensiones", porque solo es necesario representar la superficie superior.
- Por lo tanto, la elevación se puede conceptualizar como una propiedad de campo escalar o una **función del espacio bidimensional**, lo que le otorga una serie de eficiencias de modelado de datos sobre objetos tridimensionales reales.

Primitivas Bidimensionales

Primitivas Gráficas:

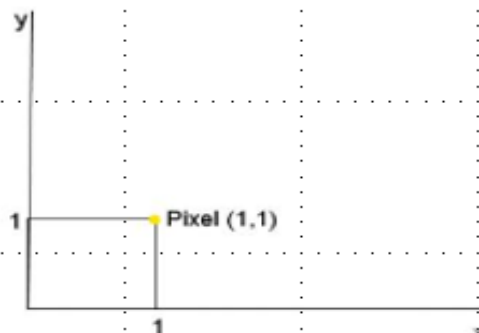
El término primitivas gráficas se refiere a las partes básicas para la construcción de imágenes, dentro de ellas se encuentran incluidas las cadenas de caracteres y las identidades geométricas, tales como los **puntos**, **las líneas rectas**, **las líneas curvas**, **los rellenos con color** (habitualmente polígonos), y las formas que se definen mediante matrices de puntos de color.

Plano cartesiano

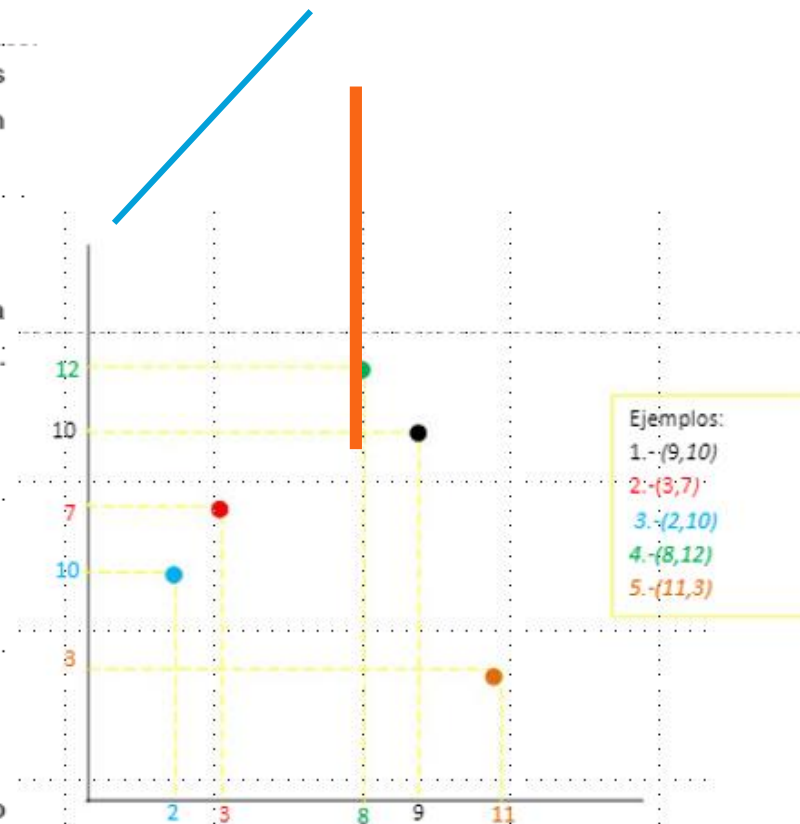
Son un sistema de referencia respecto de un eje (recta), dos ejes (plano), o tres ejes (en el espacio) perpendiculares entre sí, cortadas en un punto llamado origen de coordenadas.

Pixel

Punto de pantalla que se puede iluminar por el haz de electrones (forma abreviada de **picture element**; **elemento de imagen**).



Representación en el plano cartesiano



Primitivas Bidimensionales

Línea

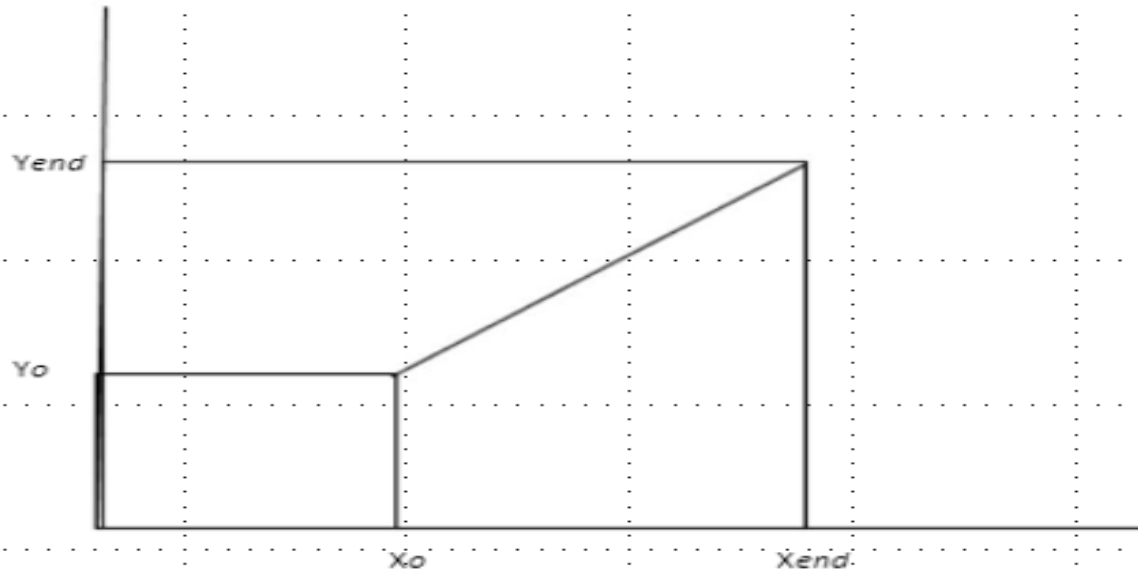
Segmento de haz de electrones que se iluminan a lo largo de una escena definida por coordenadas de los dos extremos del segmento. Para ser plasmada la línea en monitor digital, el sistemas gráfico debe primero proyectar las coordenadas de pantalla de valor entero y determinar las posiciones de pixel más próximas a lo largo de la línea que conecta los dos extremos.

Para determinar las posiciones de los pixeles a lo largo de un trayecto de una línea recta se utilizan las propiedades geométricas de la línea.

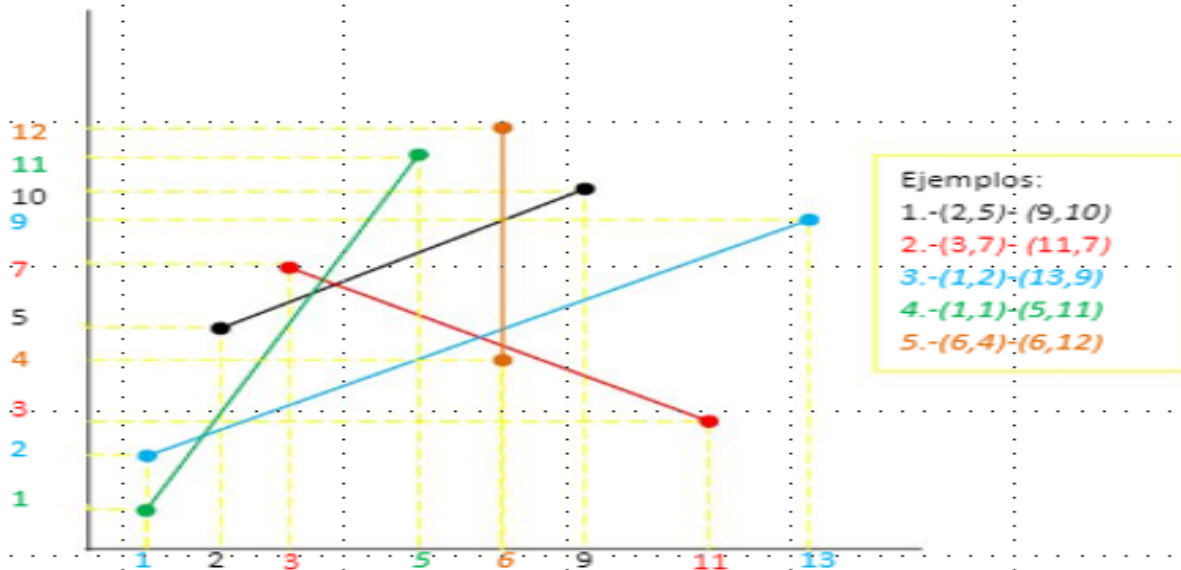
Ecuación punto - pendiente
$$Y = m \cdot x + b$$



Primitivas Bidimensionales



Representación grafica de la línea.



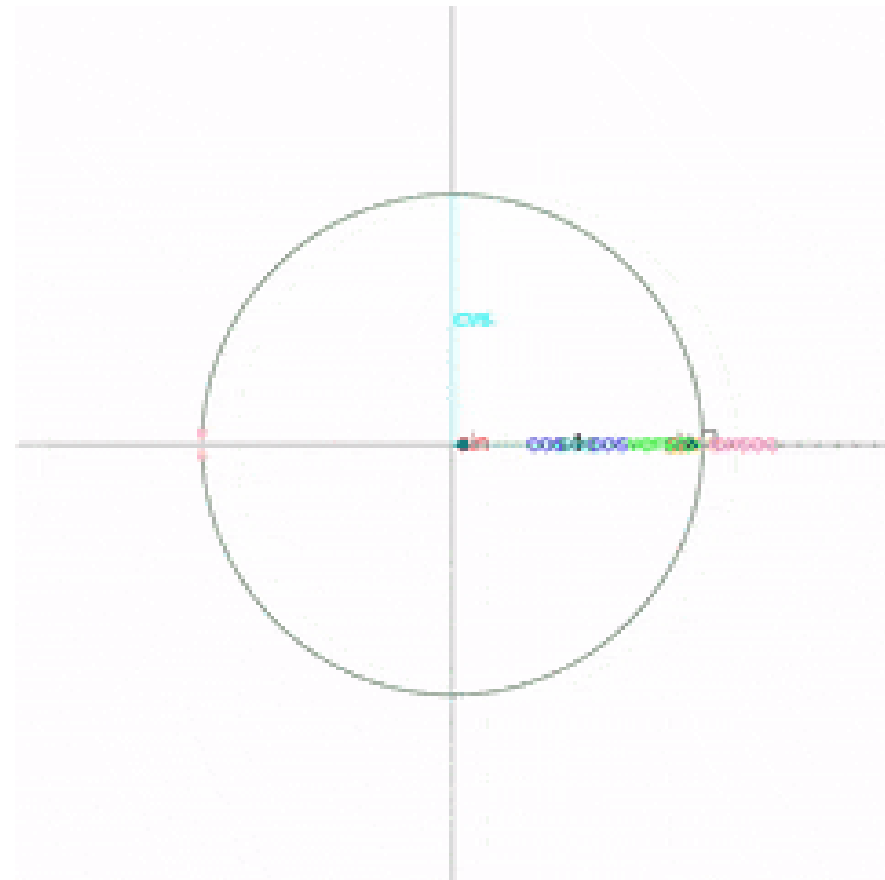
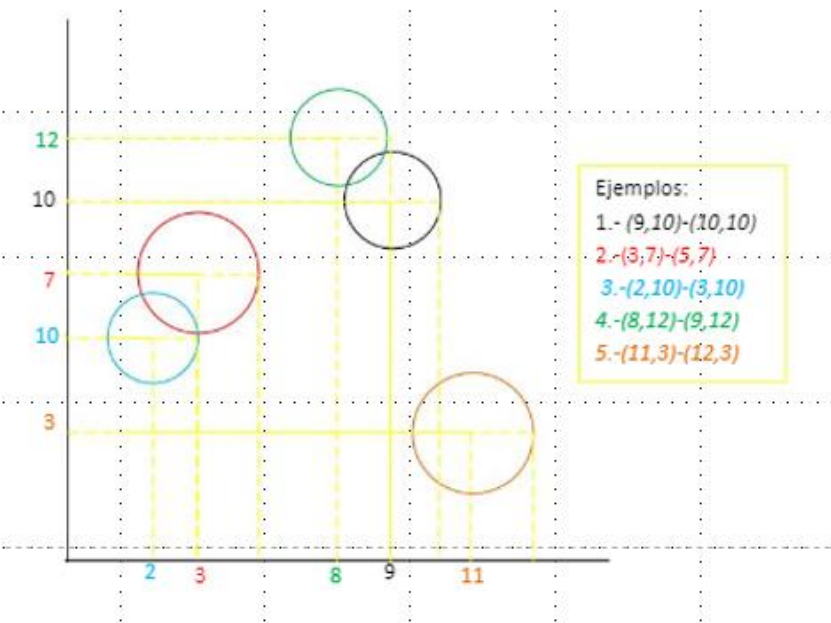
Primitivas Bidimensionales

Círculo

Se define como el conjunto de puntos que se encuentran a una distancia determinada r con respecto a una posición central (x, y) .

Para cualquier punto (x,y) del círculo, relación de distancia se expresa mediante el teorema de Pitágoras en coordenadas cartesianas.

$$(x - x_c)^2 + (y - y_c)^2 = r^2$$



Primitivas Bidimensionales

Polígono

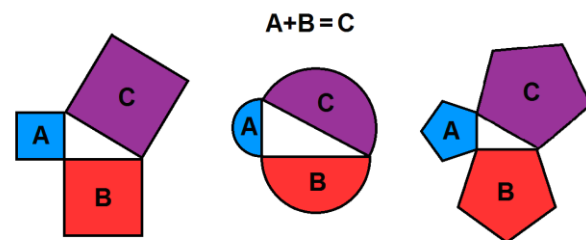
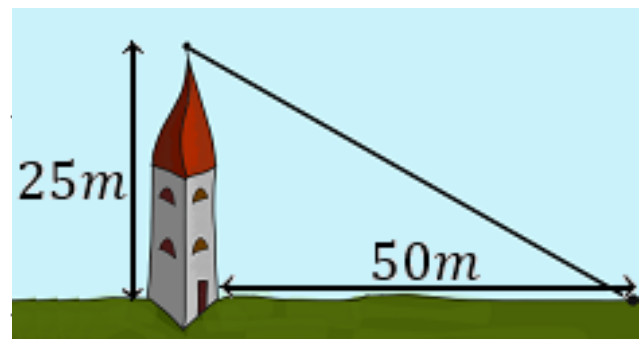
El polígono se define como una figura plana especificada mediante un conjunto de tres o más puntos denominados *vértices*, que se conectan subsecuentemente mediante segmentos lineales, denominados *bordes* o *aristas* de polígono. Algo importante a considerar es que en la geografía, es necesario que las aristas del polígono no tengan ningún punto en común aparte de los extremos.

Por definición, un polígono debe tener todos sus vértices en un mismo plano y las aristas no pueden cruzarse. Como ejemplos de polígonos se puede citar a triángulos, rectángulos, octógonos y decágonos.

Para una aplicación infografica, en ocasiones un conjunto designado de vértices no se plasman en el mismo plano, esto se debe a errores de redondeo en el cálculo de los valores numéricos. Cada polígono en una escena está contenido dentro de un plano de extensión infinita, la ecuación general es:

$$Ax + By + Cz + D = 0$$

Donde (x, y, z) es cualquier punto del plano y los coeficientes A, B, C, D son constantes que describen las propiedades especiales del plano.

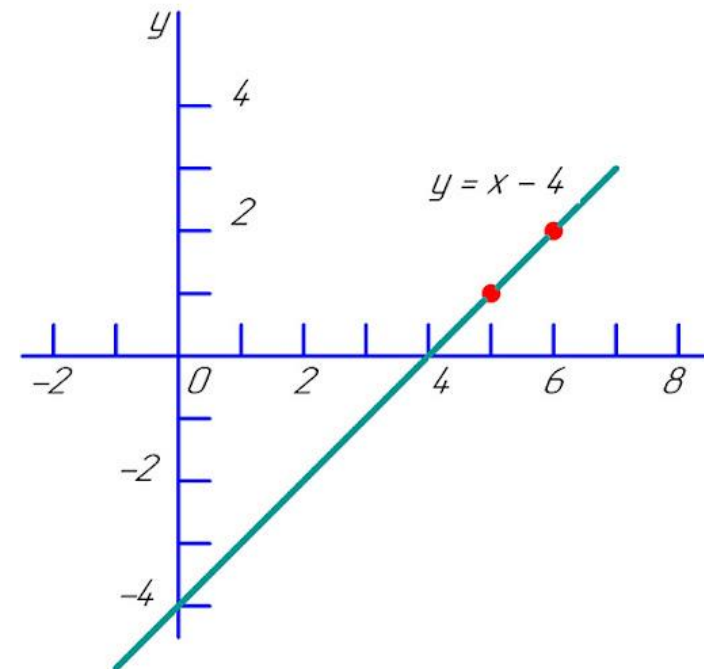
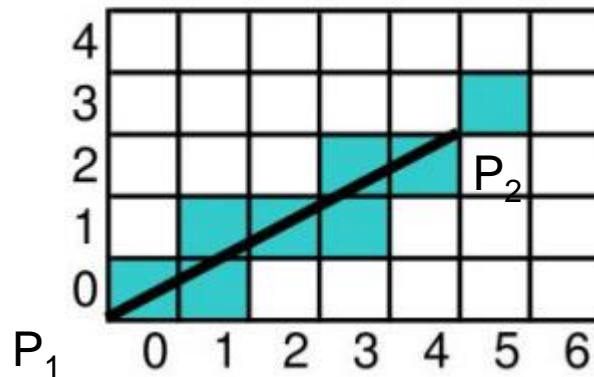


Primitivas Bidimensionales

Representación de líneas:

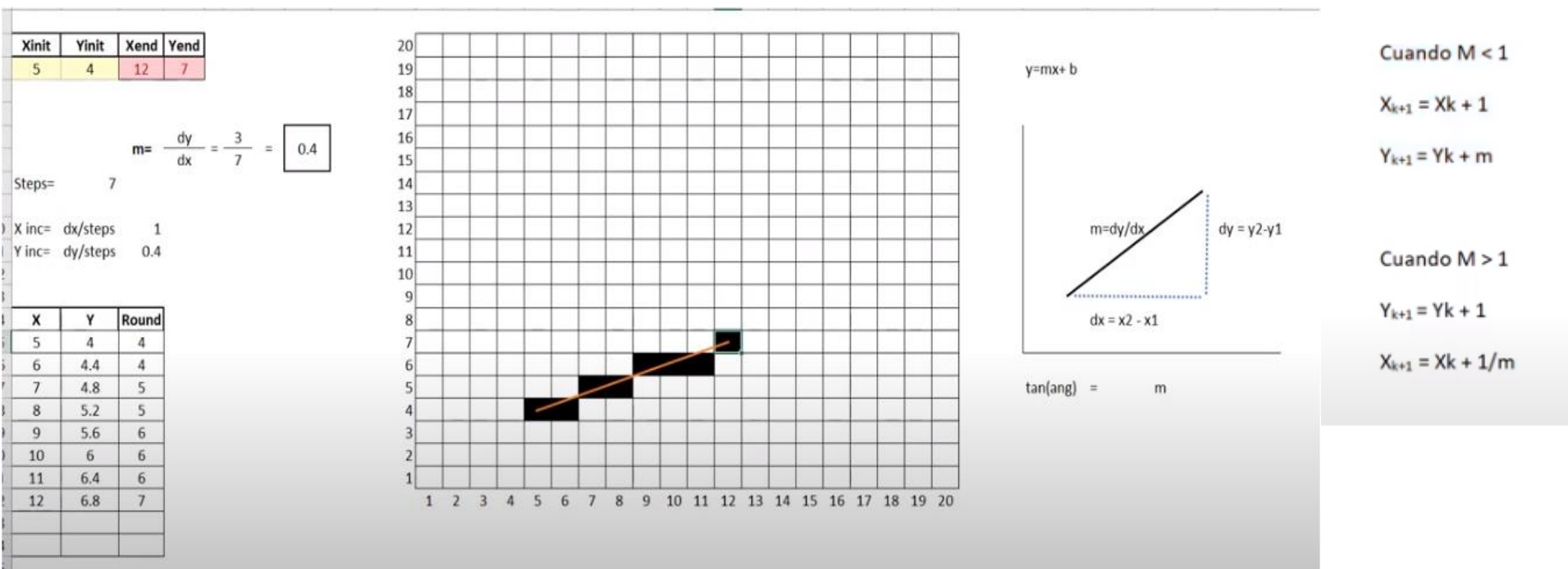
Para la representación de líneas debe considerarse 2 puntos de control $P_1(x_1, y_1)$ y $P_2(x_2, y_2)$.

Determinar los pixeles que debe de marcar para genera una línea.



Algoritmo de Línea DDA

<https://youtu.be/5EAoxZGI5Y8>

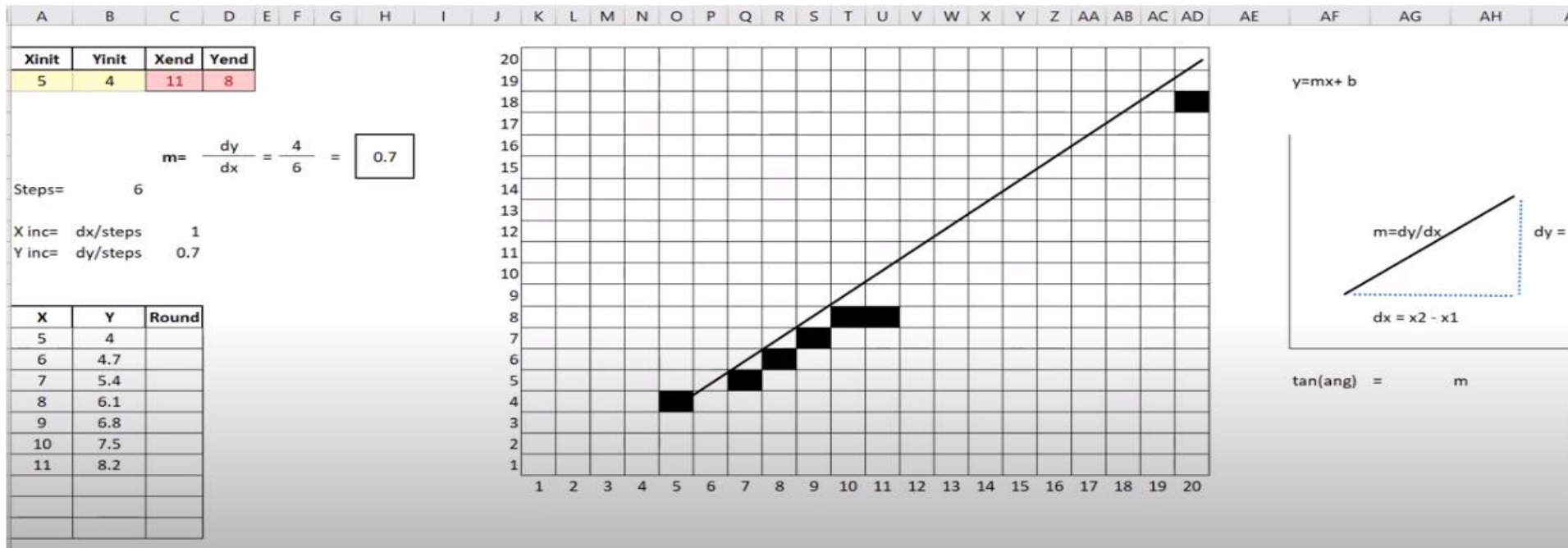


En gráficos por ordenador, una implementación de hardware o software de un **analizador diferencial digital (DDA)** se usa para la interpolación lineal de variables sobre un intervalo entre un punto de comienzo y un punto de fin. Los DDAs se usan para rastreo de líneas, triángulos y polígonos. En la implementación más simple del algoritmo DDA interpola valores en intervalo $[(x_{\text{inicio}}, y_{\text{inicio}}), (x_{\text{fin}}, y_{\text{fin}})]$ por cálculo para cada x_i las ecuaciones:

$$x_i = x_{i-1} + 1, y_i = y_{i-1} + \Delta y / \Delta x, \text{ donde } \Delta x = x_{\text{fin}} - x_{\text{inicio}} \text{ y } \Delta y = y_{\text{fin}} - y_{\text{inicio}}.$$

Algoritmo de Línea Bresenham

https://youtu.be/2_BCYD_FwII



Un algoritmo preciso y efectivo para la generación de líneas de rastreo, desarrollado por **Bresenham**, convierte mediante rastreo las líneas al utilizar sólo cálculos incrementales con enteros que se pueden adoptar para desplegar circunferencias y otras curvas.

El algoritmo de línea de Bresenham se basa en probar el signo de un parámetro entero, cuyo valor es proporcional a la diferencia entre las separaciones de las dos posiciones de pixel de la trayectoria real de la línea.

Es el campo de la informática visual, donde se utilizan computadoras para generar imágenes visuales y espaciales del mundo real.

También podemos definirlo como el **arte de transmitir información usando imágenes que son generadas mediante la computación.**

Tiene por objetivo proporcionar un contexto dentro del cual se desarrolla la actividad del Cómputo Gráfico, abarcando aspectos históricos y tecnológicos, para así comprender la importancia de ésta área de desarrollo.

En la actualidad, **los gráficos por computador se emplean en una gran variedad de aplicaciones, como en interfaces gráficas de usuario, tipografía digital, paseos arquitectónicos virtuales, aplicaciones médicas y juegos de vídeo,** entre otras.

La computación gráfica comprende una gran variedad de técnicas que pueden ser agrupadas de acuerdo al **número de dimensiones** que se empleen en la representación del modelo geométrico a visualizar, en **2D** y **3D**.

Aplicación de la Computación Gráfica

Actualmente existen muchas aplicaciones, en diversos campos de la ingeniería e investigación científica, que demandan una gran cantidad de recursos computacionales.

La Computación Gráfica cubre áreas muy diversas, que abarcan desde la visualización científica o ingenieril hasta el arte y el tratamiento fotográfico.

- Interfaces Gráficas de Usuario (GUI: Graphical User Interface)
- Gráficos estadísticos
- Cartografía
- Medicina
- Diseño Asistido por Computadora (CAD: Computer-Aided Design)
- Multimedia (educativos)
- Entretenimiento (juegos).
- Gestión de Riesgos (Inundaciones, terremotos, tsunamis, volcanes, contaminación, etc).

Ejercicios



colab python



<https://colab.research.google.com/notebooks/intro.ipynb#recent=true>

Te damos la bienvenida a Colaboratory

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Índice

Primeros pasos

Ciencia de datos

Aprendizaje automático

Más recursos

Ejemplos de aprendizaje automático

Sección

¿Qué es Colaboratory

Colaboratory, también llamado Colab, es un entorno de ejecución de código en la nube que permite ejecutar código Python y Jupyter Notebook directamente en el navegador. No requiere configuración y da acceso gratuito a los recursos de Google Cloud. Permite compartir tu trabajo y colaborar con otros.

Colab puede facilitar tu aprendizaje de Python y Jupyter Notebook. Obtener más información.

Primeros pasos

El documento que estás viendo es un cuaderno de Jupyter. Puedes escribir y ejecutar código Python directamente en el navegador. Por ejemplo, a continuación se muestra cómo crear una variable y almacenar en ella un valor.

```
[ ] seconds_in_a_day = 86400
```

Si quieres ejecutar el código, usa la combinación de teclas **Shift + Enter**.

Las variables que defines en una celda se pueden usar después en otras celdas:

Ejemplos

Recientes

Google Drive

GitHub

Subir

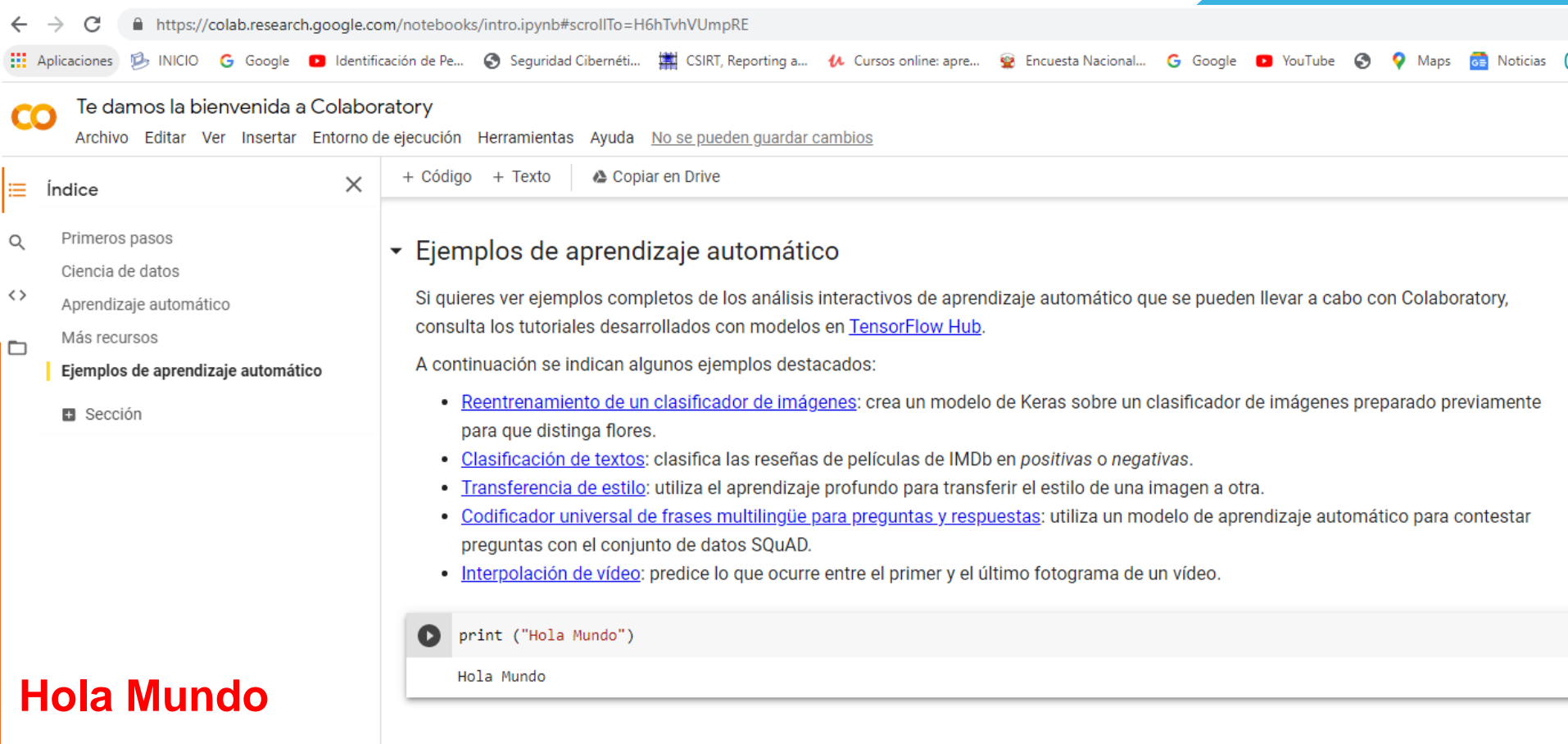
Filtrar cuadernos

Título	Abierto por primera vez	Abierto por última vez	
Te damos la bienvenida a Colaboratory	3 abr 2021	hace 0 minutos	
Ejercicios.ipynb	hace 6 días	hace 6 días	
Copia de Te damos la bienvenida a Colaboratory	hace 9 días	hace 9 días	
Practica01.ipynb	hace 13 días	hace 9 días	
1_Primeros_pasos_en_python_.ipynb	hace 11 días	hace 11 días	

NUEVO CUADERNO

CANCELAR

Ejercicios



The screenshot shows the Google Colaboratory web interface. The browser address bar displays the URL: `https://colab.research.google.com/notebooks/intro.ipynb#scrollTo=H6hTvhVUmpRE`. The interface includes a top navigation bar with tabs for 'Aplicaciones', 'INICIO', 'Google', 'Identificación de Pe...', 'Seguridad Cibernética...', 'CSIRT, Reporting a...', 'Cursos online: apre...', 'Encuesta Nacional...', 'Google', 'YouTube', 'Maps', and 'Noticias'. Below this, a header area says 'Te damos la bienvenida a Colaboratory' and lists menu items: 'Archivo', 'Editar', 'Ver', 'Insertar', 'Entorno de ejecución', 'Herramientas', 'Ayuda', and a warning 'No se pueden guardar cambios'. The left sidebar contains an 'Índice' (Index) with a search bar and a list of items: 'Primeros pasos', 'Ciencia de datos', 'Aprendizaje automático', 'Más recursos', and 'Ejemplos de aprendizaje automático' (which is selected and highlighted with a yellow bar). Under 'Ejemplos de aprendizaje automático', there is a sub-item 'Sección'. The main content area is titled 'Ejemplos de aprendizaje automático' and contains the following text: 'Si quieres ver ejemplos completos de los análisis interactivos de aprendizaje automático que se pueden llevar a cabo con Colaboratory, consulta los tutoriales desarrollados con modelos en [TensorFlow Hub](#). A continuación se indican algunos ejemplos destacados:'. This is followed by a bulleted list of five examples: 1. 'Reentrenamiento de un clasificador de imágenes': crea un modelo de Keras sobre un clasificador de imágenes preparado previamente para que distinga flores. 2. 'Clasificación de textos': clasifica las reseñas de películas de IMDb en *positivas* o *negativas*. 3. 'Transferencia de estilo': utiliza el aprendizaje profundo para transferir el estilo de una imagen a otra. 4. 'Codificador universal de frases multilingüe para preguntas y respuestas': utiliza un modelo de aprendizaje automático para contestar preguntas con el conjunto de datos SQuAD. 5. 'Interpolación de vídeo': predice lo que ocurre entre el primer y el último fotograma de un vídeo. Below the list, there is a code cell with a play button icon, containing the code `print ("Hola Mundo")`. The output of the code cell is 'Hola Mundo'.

Hola Mundo

```
print ("Hola Mundo")
nums = [1,2, 3, 4]
for n in nums:
    print(n)
```

Revisión de los Avances del Trabajo Final

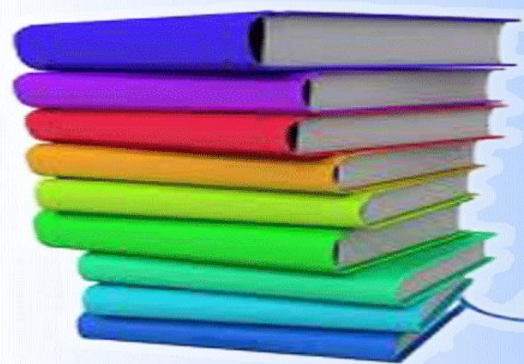
¿Qué bibliografía encontraron sobre su tema de investigación?

Escritos científicos del 2017 a la fecha

Tesis de grados del 2017 a la fecha

Otros

Preguntas pertinentes...



Referencias
Bibliográficas





**Un descanso de 15'
Minutos
(Break Gráfico Digital)**



MI DESAYUNO DE HOY

DESARROLLO DE APLICACIONES ALGORITMOS DE COMPUTACION GRAFICA



PRACTICA DE ALGORITMOS DE COMPUTACION GRAFICA



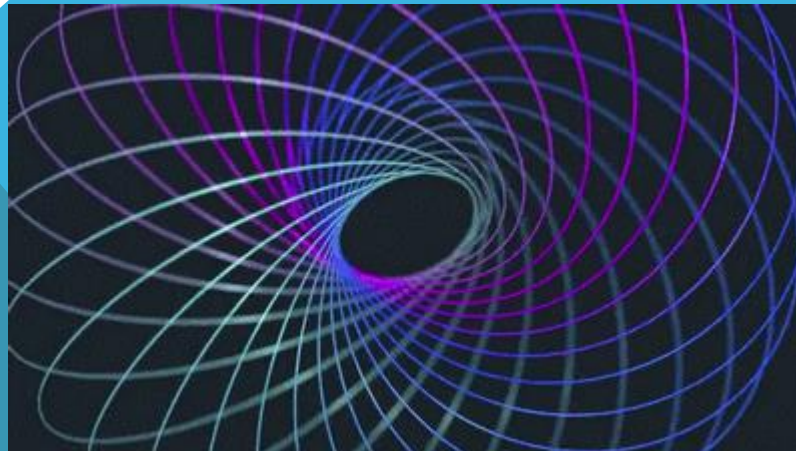
MA. Juan Carlos Reátegui Morales

jreategui@untels.edu.pe

MBA-ISO 27001-ISO 9001-ISO 22301

Creando el Entorno de Python

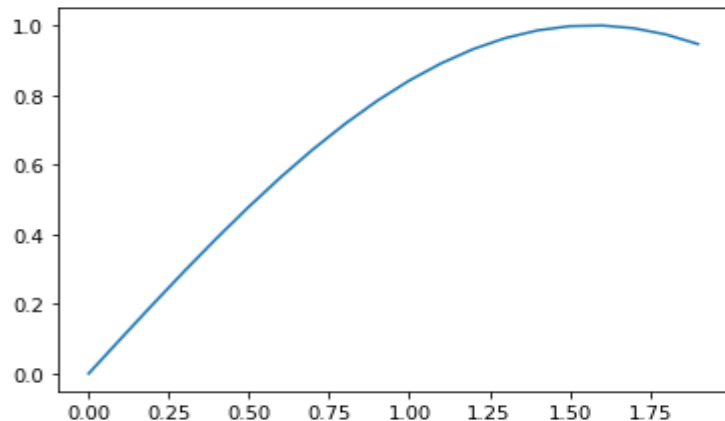
<https://www.youtube.com/watch?v=TMdyG6DaJkI>



Ejercicios Asíncronos

```
#-----  
#Genera la función Seno  
#Autor:  
#Fecha: 25-04-2022  
#-----  
import math  
import numpy as np  
from matplotlib import pyplot as plt  
x= np.array(range(20))*0.1  
print ("X=",x)  
y= np.zeros(len(x))  
print ("Y=",y)  
print ("Función Seno")  
for i in range(len(x)):  
    y[i]=math.sin(x[i])  
# Creamos el gráfico  
plt.plot(x,y)  
plt.show()
```

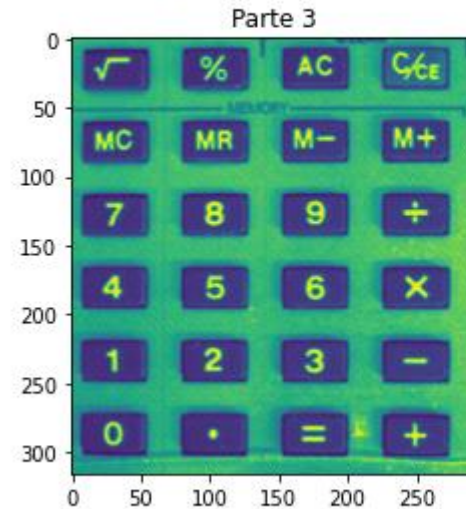
```
X= [0.  0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.  1.1 1.2 1.3 1.4 1.5 1.6 1.7  
    1.8 1.9]  
Y= [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]  
Función Seno
```



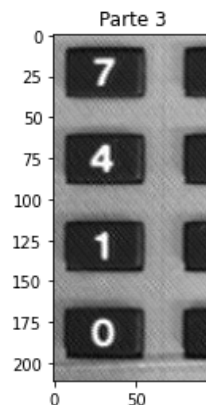
Ejercicios Asíncronos

```
#-----  
#Recorta una imagen  
#Autor:  
#Fecha: 25-04-2022  
#-----  
import numpy as np  
import matplotlib.image as pinta  
import matplotlib.pyplot as plt  
from google.colab import files  
#Lee la imagen  
upload=files.upload()  
a=pinta.imread('keyb.tif')  
#Recorta la tercera imagen  
Dim=np.shape(a)  
a_recorte_3=a[int(Dim[0]/3):int(Dim[0]),0:int(Dim[1]/3)]  
#Imprime Imagen  
plt.imshow(a_recorte_3,cmap='gray')  
plt.imshow(a)  
plt.title('Parte 3')
```

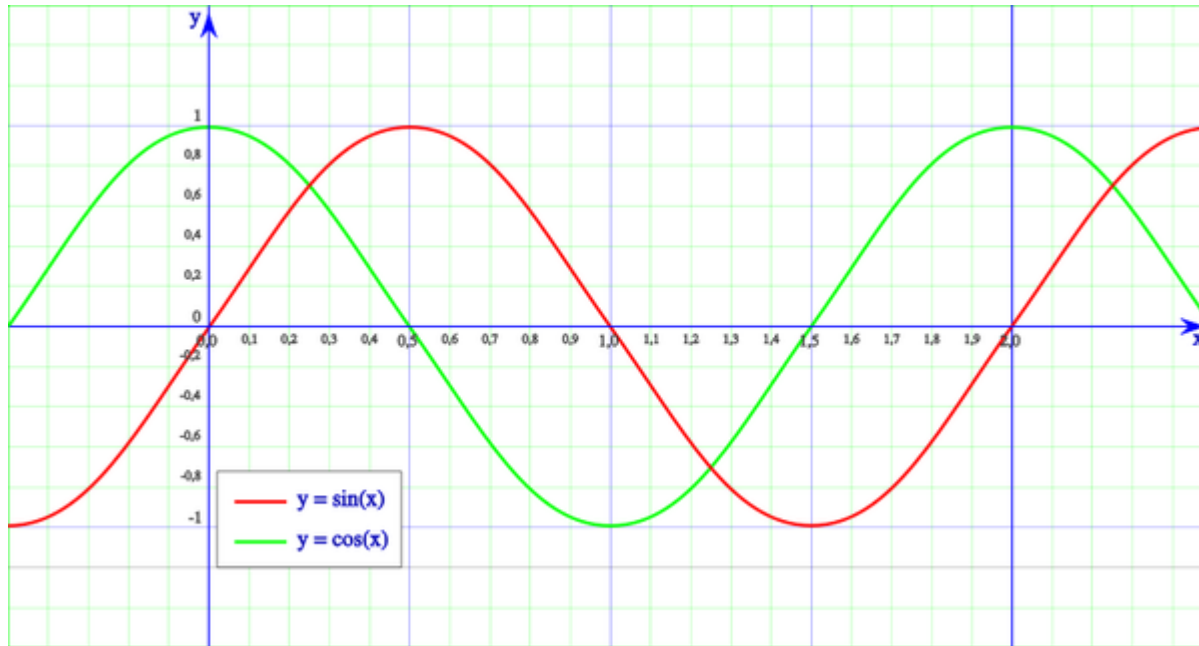
• **keyb.tif**(image/tiff) - 94802 bytes, last modified: 4/4/2021 - 100% done
Saving keyb.tif to keyb (5).tif
Text(0.5, 1.0, 'Parte 3')



• **keyb.tif**(image/tiff) - 94802 bytes, last modified: 4/4/2021 - 100% done
Saving keyb.tif to keyb (2).tif
Text(0.5, 1.0, 'Parte 3')



Ejercicios Asíncronos



Graficar la función seno

Graficar la función Coseno

Graficar la función Tangente

Ejercicios Asíncronos (Reto)

Cortar e imprimir la calculadora cortada en 4 partes en Python



Control de Aprendizaje

Preguntas de Control:

- ¿Qué es una primitiva bidimensional?. Para que sirven.
- ¿En que consiste el algoritmo DDA?
- ¿ En que consiste el algoritmo Bresenham?
- ¿Qué opina de los gráficos con Phyton?. ¿Cuál puede ser su utilidad practica?
- ¿Para que sirven las gráficas de línea en computación gráfica?
- ¿Cómo podemos generar valor con aplicaciones de Computación Gráfica?

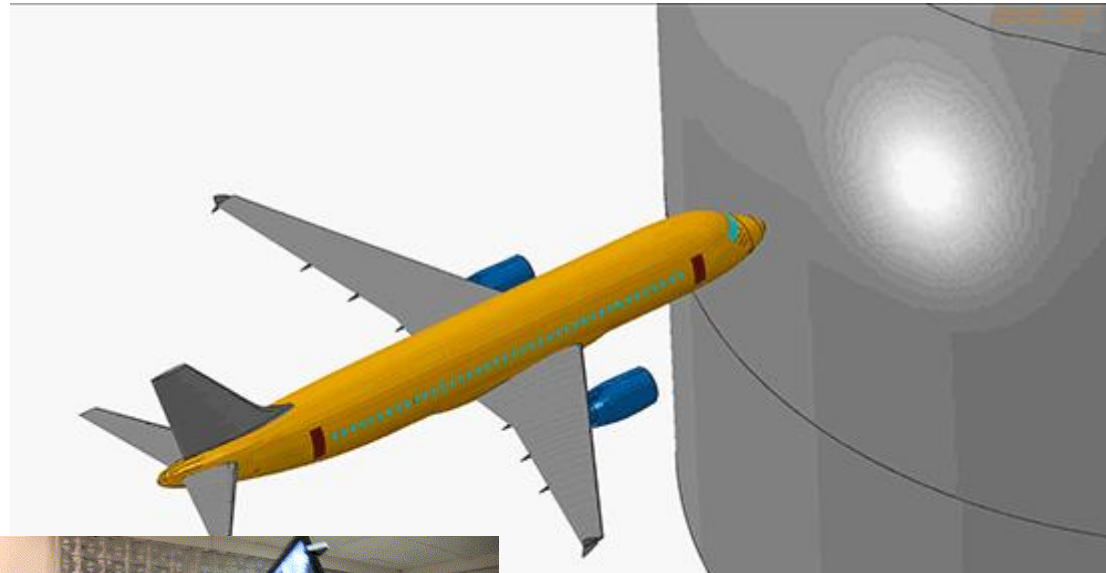
ALGORITMOS DE COMPUTACION GRAFICA

Lunes 08:00 - 11:20 Clase 03 Transformaciones Geométricas Bi-Direccionales

Objetivo: Generar transformaciones geométricas bidimensionales.

Implementación de algoritmos de transformación.

Implementación de las coordenadas homogéneas.



MA. Juan Carlos Reátegui Morales
jreategui@untels.edu.pe

MBA-ISO 27001-ISO 9001-ISO 22301

Todo lo que una persona puede imaginar, otras podrán hacerlo realidad.

Julio Verne

Muchas gracias...

Juan Carlos Reátegui
Morales

980929404
jreategui@untels.edu.pe