## Requerimientos

Windows, Mac OSX o cualquier otra plataforma con Python 3.5.1 instalado.

<https://www.python.org/downloads/>

## Objetivo

Desarrollar un programa que implemente lo siguiente:

1. Mostrar la forma en que se realiza la elección y el encadenamiento de constructores.
2. El uso de los diferentes tipos de vista que provee.
3. La forma en la que se realiza la búsqueda dinámica de métodos.

## Pseudocódigos

**invierteArreglo(arreglo[ ], tam)**

desde i 🡨 1 hasta tam/2

temporal 🡨 arreglo[i]

arreglo[i] 🡨 arreglo [ tam –i ]

arreglo[ tam – i] 🡨 temporal

i 🡨 i+1

**distanciaAngulo(arreglo[ ][ ],tam)**

resultados[ ][ ]

desde i 🡨 1 hasta tam

resultados[ i ][ 1 ] 🡨 raíz(( arreglo[ i +1][ 1 ] – arreglo[ i ][ 1 ])2  + (arreglo[ i +1][ 2 ] – arreglo[ i ][ 2 ])2 )

resultados[ i ][ 2 ] 🡨 angtan()

i 🡨 i+1

**Regresa resultados[][]**

## Modo de uso.

1. Abrir el programa e ingresar en primera instancia los valores del arreglo
2. Posteriormente se deben ingresar los puntos en el plano cartesiano que serán utilizados para calcular las distancia a la que se encuentra uno respecto de otro y el ángulo que forman entre ellos, para ello es necesario ingresar cada punto separando con una coma cada componente

* Ejemplo:

Si se desea ingresar el punto (4,9) sera suficiente con ingresar al programa 4,9

1. Una vez hecho lo anterior, el programa realiza las operaciones pertinentes y despliega en pantalla dos resultados

* El primero de ellos corresponde con la lista invertida
* El segundo corresponde a pares ordenados, cuyo primero elemento es la distancia entre los puntos ingresados, y el segundo el ángulo que forman entre ellos

Cabe aclarar que se pueden meter tantos puntos como se haya seleccionado el tamaño del arreglo, la distancia y el ángulo se irán calculando tomando en cuenta pares de puntos

* Ejemplo

Si se ingresan los puntos {(0,0),(1,1),(2,1)}

La salida del programa será: {(1.4142,45),(1,0)}

Donde se puede observar que primero se tomaron los puntos (0,0) y (1,1) para el cálculo y después los puntos (1,1) y (2,1) para el siguiente

De manera general a n puntos en el plano de entrada, se obtendrán n-1 vectores resultado

## Pruebas

