ESCUELA POLITÉCNICA NACIONAL

MÉTODOS NUMÉRICOS

JIMÉNEZ JARAMILLO YASID GABRIEL

[Actividad extracurricular 09] Función atan2

La función `atan2` es una extensión de la función arcotangente (`atan`) que calcula el ángulo en radianes correspondiente a un punto en el plano cartesiano (x, y). A diferencia de `atan(y / x)`, `atan2` considera el signo de ambos argumentos (x y y) para determinar el cuadrante correcto del ángulo, proporcionando un resultado en el rango $[-\pi, \pi)$.

Para su uso en python es necesario importar la libreria math y la siguiente sintaxis:

```
import math

math.atan2(y, x)
```

```
-2.5535900500422257
```

¿Por qué se recomienda usar la función atan2?

- **Manejo de todos los cuadrantes**: `atan2` calcula el ángulo correctamente para cualquier combinación de signos de `x` y `y`, ubicando el ángulo en el cuadrante correcto.
- **Evita divisiones por cero**: No requiere dividir `y` entre `x` explícitamente, evitando errores o excepciones cuando `x = 0`.
- **Mayor precisión**: Es más precisa que realizar una división manual seguida de `atan`, ya que aprovecha optimizaciones internas.
- **Resultados en un rango completo**: Retorna ángulos en el rango $[-\pi, \pi)$, a diferencia de `atan`, que devuelve valores en el rango $[-\pi/2, \pi/2)$.

Diferencias entre función atan y atan2.

Característica	atan(y/x)	atan2(y, x)
Parámetros	Una relación (y/x)	Dos valores ((y), (x))
Rango de salida	([-π/2, π/2])	([-π, π])
Cuadrantes soportados	Solo considera el ángulo en el 1er y 4to cuadrante	Identifica los 4 cuadrantes
Manejo de división por cero	Puede lanzar un error si (x = 0)	No hay errores, maneja (x = 0)
Usabilidad	Menos común en problemas geométricos	Amplia utilidad en geometría

EJEMPLOS

```
# Ejemplo 1: Diferencia entre atan y atan2 en el primer cuadrante
y, x = 2, 3
angle_atan = math.atan(y / x)  # Usamos atan con la relación y/x
```

```
angle_atan2 = math.atan2(y, x)  # Usamos atan2 directamente con y y x

print(f"Ejemplo 1: Primer cuadrante")
print(f"atan(y/x): {angle_atan} radianes")
print(f"atan2(y, x): {angle_atan2} radianes\n")
```

```
Ejemplo 1: Primer cuadrante
atan(y/x): 0.5880026035475675 radianes
atan2(y, x): 0.5880026035475675 radianes
```

Análisis: En este caso, tanto `atan` como `atan2` proporcionan el mismo resultado, ya que estamos en el primer cuadrante, donde tanto x como y son positivos. Ambas funciones manejan correctamente esta situación.

```
# Ejemplo 2: Diferencia entre atan y atan2 en el segundo cuadrante
y, x = 2, -3
angle_atan = math.atan(y / x)  # Usamos atan con la relación y/x
angle_atan2 = math.atan2(y, x)  # Usamos atan2 directamente con y y x

print(f"Ejemplo 2: Segundo cuadrante")
print(f"atan(y/x): {angle_atan} radianes")
print(f"atan2(y, x): {angle_atan2} radianes\n")
```

```
Ejemplo 2: Segundo cuadrante
atan(y/x): -0.5880026035475675 radianes
atan2(y, x): 2.5535900500422257 radianes
```

Análisis: Aquí, `atan` no puede distinguir que estamos en el segundo cuadrante, ya que solo toma en cuenta la relación y/x y devuelve un valor en el primer o cuarto cuadrante (negativo en este caso). Sin embargo, `atan2` maneja correctamente la señal de x y y, proporcionando el ángulo en el segundo cuadrante (aproximadamente 146.3°).

```
# Ejemplo 3: Diferencia entre atan y atan2 en el tercer cuadrante
y, x = -2, -3
angle_atan = math.atan(y / x)  # Usamos atan con la relación y/x
angle_atan2 = math.atan2(y, x)  # Usamos atan2 directamente con y y x

print(f"Ejemplo 3: Tercer cuadrante")
print(f"atan(y/x): {angle_atan} radianes")
print(f"atan2(y, x): {angle_atan2} radianes\n")
```

```
Ejemplo 3: Tercer cuadrante
atan(y/x): 0.5880026035475675 radianes
atan2(y, x): -2.5535900500422257 radianes
```

Análisis: En el tercer cuadrante, `atan` nuevamente no puede diferenciar entre x y y siendo negativos. Esto resulta en un valor positivo de 0.588 radianes, que no es correcto para este

cuadrante. Por otro lado, `atan2` devuelve el ángulo correcto para el tercer cuadrante, con un valor negativo de -2.554 radianes (aproximadamente -146.3°).

```
# Ejemplo 4: Diferencia entre atan y atan2 en el cuarto cuadrante
y, x = -2, 3
angle_atan = math.atan(y / x)  # Usamos atan con la relación y/x
angle_atan2 = math.atan2(y, x)  # Usamos atan2 directamente con y y x

print(f"Ejemplo 4: Cuarto cuadrante")
print(f"atan(y/x): {angle_atan} radianes")
print(f"atan2(y, x): {angle_atan2} radianes\n")
```

```
Ejemplo 4: Cuarto cuadrante
atan(y/x): -0.5880026035475675 radianes
atan2(y, x): -0.5880026035475675 radianes
```

Análisis: Aquí, tanto `atan` como `atan2` devuelven el mismo resultado porque estamos en el cuarto cuadrante, donde $x>0$ y $y<0$, y ambos métodos correctamente determinan un ángulo negativo (aproximadamente -33.7°).

```
# Ejemplo 5: Caso especial con x = 0
y, x = 5, 0
# En este caso, no intentamos dividir por 0, ya que atan no lo permite
# Se maneja la condición para evitar el error de división por cero en
atan
try:
    angle_atan = math.atan(y / x)  # Esto fallará si x es 0
except ZeroDivisionError:
    angle_atan = "Error - División por cero"

angle_atan2 = math.atan2(y, x)  # atan2 maneja este caso correctamente

print(f"Ejemplo 5: Caso con x = 0")
print(f"atan(y/x): {angle_atan}")
print(f"atan2(y, x): {angle_atan2} radianes\n")
```

```
Ejemplo 5: Caso con x = 0
atan(y/x): Error - División por cero
atan2(y, x): 1.5707963267948966 radianes
```

Análisis: Como se esperaba, cuando $x=0$, `atan` no puede calcular el ángulo debido a la división por cero. En cambio, `atan2` maneja correctamente esta situación y devuelve $\pi/2$ radianes (90°) cuando $y>0$. Esto muestra la ventaja de `atan2`, que está diseñada para manejar casos donde $x=0$.

REPOSITORIO:

https://github.com/ImYasid/METODOS_NUMERICOS.git

REFERENCIAS BIBLIOGRÁFICAS:

[1] Richard L. Burden, 2017. Análisis Numérico. Lugar de publicación: 10ma edición. Editorial Cengage Learning.

DECLARACIÓN DEL USO DE INTELENGIA ARTIFICIAL

Se utilizo IA para la optimización de código adicional al mejoramiento de la gramática del texto para un mejor entendimiento.