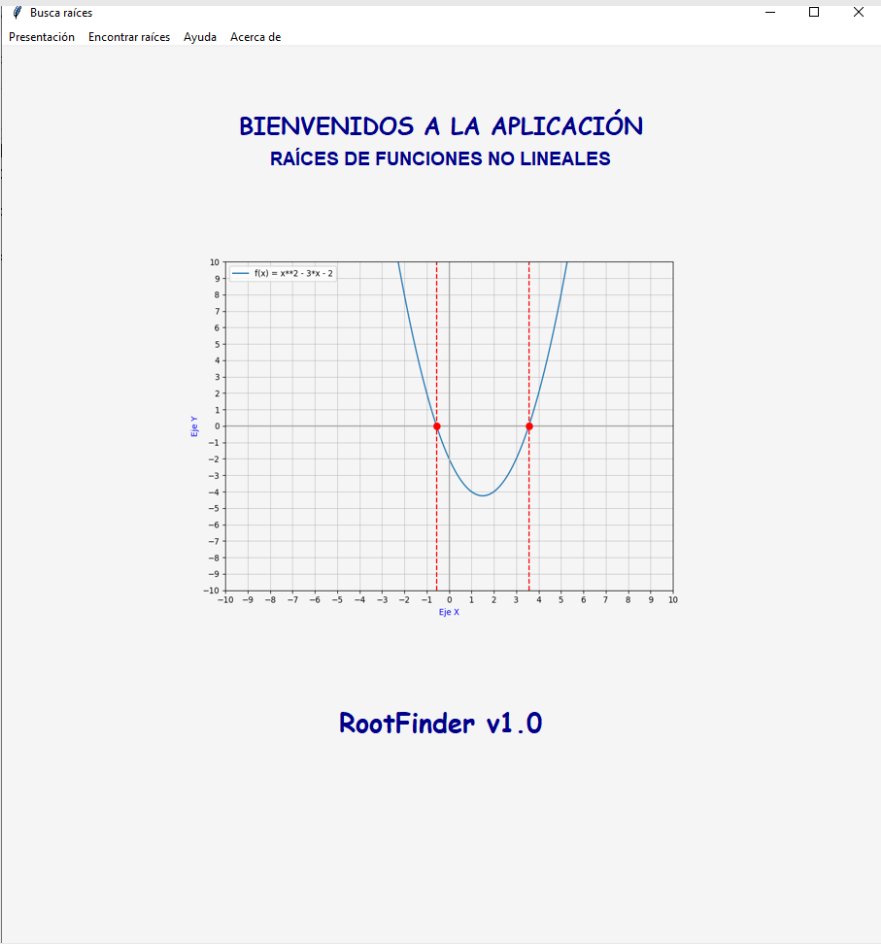


# SOFTWARE DE APLICACIÓN

## RAÍCES DE FUNCIONES NO LINEALES



# MANUAL DE USUARIO

## **SOFTWARE DE APLICACIÓN**

**RootFinder v1.0:** Determina los valores de  $x$  que satisfacen  $f(x)=0$ , donde  $f$  representa una función no lineal.

### **AUTORES:**

- LEONEL COYLA IDME
- ALFREDO MAMANI CANQUI
- ELQUI YEYE PARI CONDORI
- JUAN REYNALDO PAREDES QUISPE
- JOSÉ PÁNFILO TITO LIPA

## ÍNDICE

1.	INTRODUCCIÓN .....	4
2.	REQUISITOS DEL SISTEMA .....	5
2.1.	SISTEMA OPERATIVO.....	5
2.2.	HARDWARE .....	5
2.3.	SOFTWARE .....	5
3.	INSTALACIÓN Y CONFIGURACIÓN .....	5
3.1.	INSTALAR PYTHON .....	5
3.2.	EJECUTAR LA APLICACIÓN .....	5
3.3.	CONFIGURACIÓN OPCIONAL .....	6
3.4.	SOLUCIÓN DE PROBLEMA.....	6
3.5.	ACTUALIZACIÓN .....	6
4.	DESCRIPCIÓN DE LA APLICACIÓN RootFINDER v1.0 .....	8
4.1.	MENÚ PRESENTACIÓN .....	8
4.3.	MENÚ AYUDA .....	12
4.4.	MENÚ ACERCA DE .....	12
5.	PREGUNTAS FRECUENTES (FAQS) SOBRE EL SOFTWARE .....	14
5.1.	¿QUÉ ES ROOTFINDER Y CUÁL ES SU PROPÓSITO? .....	14
5.2.	¿CUÁLES SON LOS REQUISITOS MÍNIMOS DEL SISTEMA PARA EJECUTAR? .....	14
5.3.	¿CÓMO INSTALO O CONFIGURO ROOTFINDER? .....	14
5.4.	¿QUÉ SE DEBE REALIZAR SI EL SOFTWARE NO FUNCIONA CORRECTAMENTE?.....	15
6.	ADVERTENCIAS Y SUGERENCIAS DEL SOFTWARE DE APLICACIÓN.....	15
6.1.	ADVERTENCIAS.....	15
6.2.	SUGERENCIAS.....	15
7.	CÓDIGO .....	16

## 1. INTRODUCCIÓN

RootFinder v1.0 es un software educativo diseñado para facilitar el aprendizaje de los estudiantes que cursan la asignatura de Métodos Numéricos. Su principal función es graficar ecuaciones no lineales y calcular de manera aproximada la raíz de dichas ecuaciones. Estos procedimientos son fundamentales en el análisis numérico, ya que permiten encontrar soluciones a problemas matemáticos que no se resuelven fácilmente de forma algebraica.

La aplicación ofrece una interfaz gráfica amigable y didáctica, donde el usuario puede ingresar una función matemática y observar su comportamiento en un sistema de coordenadas. El gráfico generado permite identificar visualmente el punto donde la curva corta al eje  $x$ , es decir, la raíz de la ecuación. Además, el software realiza el cálculo de la raíz, lo que ayuda al estudiante a comprobar sus resultados y mejorar su comprensión de los métodos numéricos.

Gracias a su diseño práctico y funcional, RootFinder v1.0 se convierte en una herramienta de apoyo ideal para el proceso de enseñanza-aprendizaje. Su uso permite reforzar conceptos teóricos mediante la experimentación visual e interactiva, contribuyendo al desarrollo de habilidades analíticas y al entendimiento de la importancia de los métodos numéricos en la resolución de problemas matemáticos reales.

Adicionalmente, RootFinder v1.0 está pensado para ser utilizado tanto en entornos académicos como de autoaprendizaje. Su facilidad de instalación, sus herramientas intuitivas y sus resultados inmediatos permiten que cualquier estudiante o docente pueda utilizar el software sin complicaciones, promoviendo un aprendizaje autónomo y dinámico dentro y fuera del aula.

PROYECTO REGISTRADO CON URL:

<https://github.com/leonelcoyla/RootFinder>

## 2. REQUISITOS DEL SISTEMA

Para garantizar un rendimiento óptimo de RootFinder v1.0, el software diseñado para construir gráficos de ecuaciones no lineales y calcular la raíz de dicha ecuación, es fundamental que tu sistema cumpla con los requisitos mínimos y recomendados. Asegúrate de que tanto el hardware como el software de tu equipo estén correctamente configurados antes de instalar y utilizar el programa.

### 2.1. Sistema Operativo

Al aplicativo RootFinder v1.0 es compatible con los siguientes sistemas operativos:

- Windows 7, 8, 10 o superior
- Linux (Ubuntu 18.04 o superior)
- macOS (versión 10.12 o superior)

### 2.2. Hardware

- Procesador: Intel Core i3 o superior
- Memoria RAM: 2 GB mínimo (recomendado 4 GB)
- Espacio en Disco: 100 MB libres
- Resolución de pantalla mínima: 1024x768

### 2.3. Software

- Python 3.10 o superior
- Librerías de Python necesarias:
  - matplotlib
  - tkinter
  - pandas

## 3. INSTALACIÓN Y CONFIGURACIÓN

Para poner en funcionamiento RootFinder v1.0, debes asegurarte de que los siguientes componentes estén correctamente instalados en tu equipo:

### 3.1. Instalar Python

- Descargar Python desde la página oficial:  
<https://www.python.org/downloads/>
- Durante la instalación, marcar la opción "Add Python to PATH"
- Finalizar el proceso de instalación.
- 

### 3.2. Ejecutar la Aplicación

- Guarde el código dado en un archivo con el nombre RootFinder.py.
- Abra la terminal, diríjase a la carpeta donde almacenó el archivo y ejecútelo.:

python RootFinder.py

- También es posible ejecutar el archivo RootFinder.exe directamente desde su ubicación.
- Otra opción es instalar el programa ejecutando mysetup.exe desde la carpeta donde se encuentra.

Archivo principal

RootFinder.py #código fuente

RootFinder.exe archivo ejecutable

mysetup.exe archivo instalador de la aplicación

CONTACTO DEL DESARROLLADOR: +51951679658

### 3.3. Configuración opcional

**RootFinder v1.0** está configurado para funcionar con una resolución de pantalla mínima de **1024x768 px**. Si deseas usar una resolución mayor, puedes modificar el tamaño de la ventana en el código.

- Abre el archivo **RootFinder.py** en un editor de texto.
- Busca la línea donde se establece el tamaño de la ventana (en el código proporcionado, es `root.geometry("900x900")`)
- Cambia las dimensiones a tu preferencia. Por ejemplo:

```
root.geometry("900x900") # Ajusta el tamaño de la ventana
```

### 3.4. Solución de problema

Problema	Solución
Python no se reconoce como comando	Asegurarse de haber marcado <i>Add Python to PATH</i> en la instalación.
Error al instalar librerías	Ejecutar <code>pip install --upgrade pip</code> antes de instalar.
La aplicación no abre	Verificar que el archivo esté completo y en la ubicación correcta.

### 3.5. Actualización

- Para actualizar RootFinder v.1.0:
- Verificar si existe una nueva versión disponible en el sitio oficial o repositorio.

- Descargar la nueva versión.
- Reemplazar los archivos antiguos.
- Ejecutar nuevamente la aplicación.

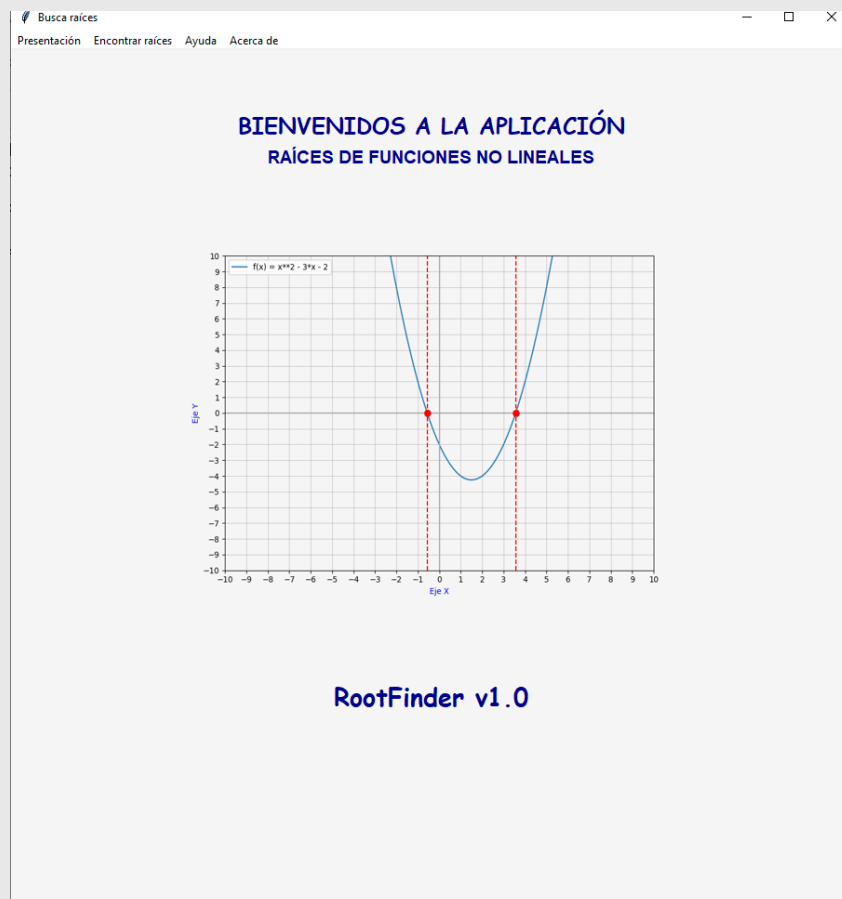
## 4. DESCRIPCIÓN DE LA APLICACIÓN RootFinder v1.0

RootFinder v1.0 es una aplicación educativa desarrollada con el objetivo de facilitar el aprendizaje de conceptos matemáticos fundamentales, como la representación gráfica de funciones no lineales y el cálculo de raíces de ecuaciones. Esta herramienta está orientada principalmente a estudiantes que desean comprender de manera visual y práctica cómo se comportan las funciones en el plano cartesiano.

El software permite al usuario ingresar una ecuación no lineal, visualizar su gráfica de forma precisa y determinar de manera aproximada la raíz o solución de dicha ecuación. RootFinder v1.0 ofrece una interfaz intuitiva, amigable y con colores llamativos, lo que hace que la experiencia de aprendizaje sea más dinámica y accesible.

### 4.1. Menú Presentación

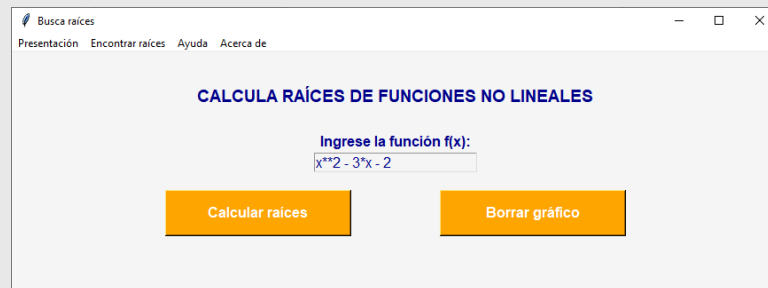
Al iniciar RootFinder v1.0, los usuarios son recibidos en el menú de presentación, donde se muestra un mensaje de bienvenida, acompañado del logotipo representativo de la aplicación y su nombre resaltado. Este espacio está diseñado para introducir al usuario a la interfaz y al propósito principal del software, permitiéndole conocer sus funciones antes de proceder a graficar ecuaciones no lineales y calcular sus respectivas raíces.



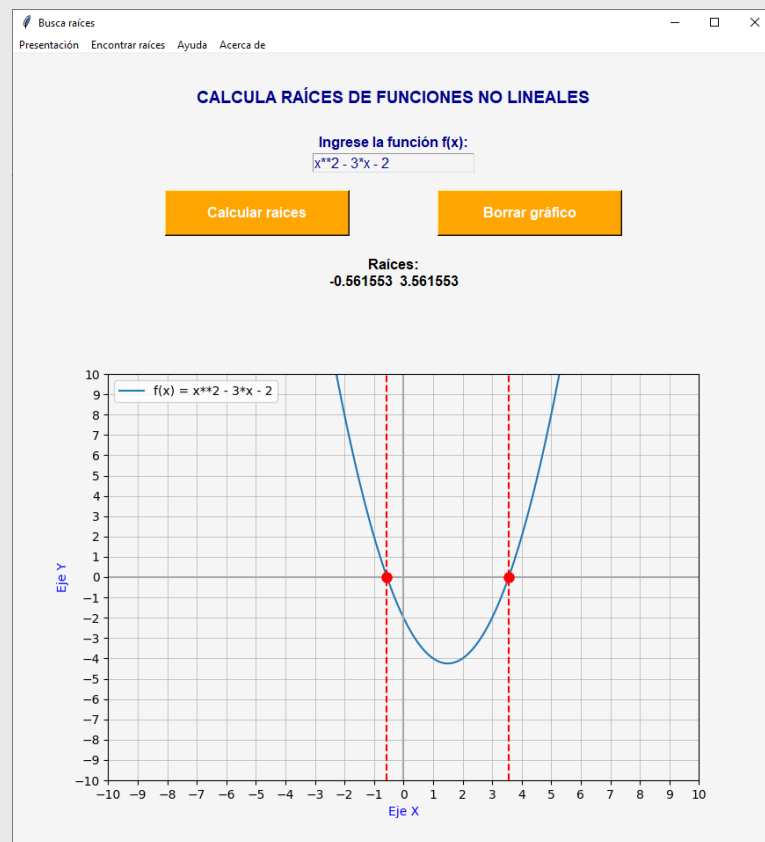


## 4.2. RootFinder v1.0

La pestaña "Gráfico" de *RootFinder v1.0* es la sección central de la aplicación, diseñada para interactuar con ecuaciones no lineales. En primer lugar, el usuario debe ingresar la ecuación matemática que desea graficar en un campo de texto específico.

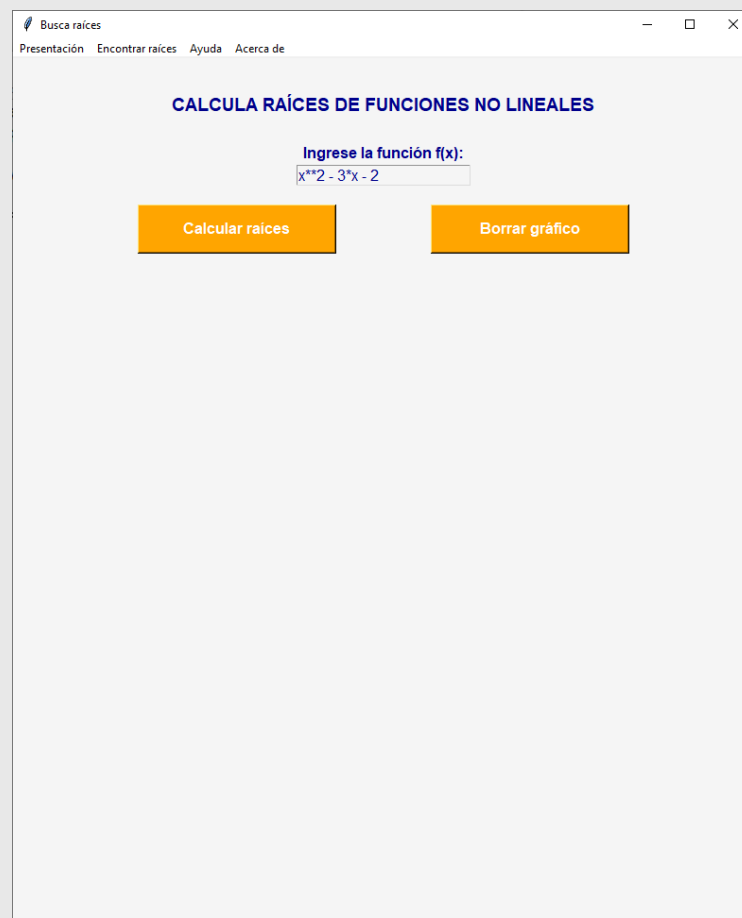


Una vez ingresada la ecuación, haga clic en el botón 'Encontrar Raíces'. Esto generará el gráfico de la ecuación y mostrará la(s) raíz(es) en el plano cartesiano, permitiendo al usuario visualizar el comportamiento de la función de manera clara y precisa.



Al generar el gráfico y mostrar las raíces de la ecuación en el intervalo de interés, el software indica los puntos donde la curva intersecta el eje x, lo que representa visualmente las soluciones de la ecuación. Además de crear el gráfico, RootFinder v1.0 resalta las raíces con marcas visuales, ayudando a los estudiantes a comprender cómo se encuentran las soluciones de las ecuaciones no lineales en el contexto de los métodos numéricos

Al finalizar el análisis, el usuario tiene la opción de borrar el gráfico generado, lo que le permite ingresar una nueva ecuación y realizar un nuevo cálculo. Este ciclo interactivo facilita el aprendizaje, permitiendo que los estudiantes experimenten con diferentes funciones y valores, y mejoren su comprensión de los conceptos teóricos mediante la visualización dinámica y el cálculo de raíces.



El código de RootFinder v1.0 está diseñado para trabajar con una amplia variedad de **funciones matemáticas**, siempre que sean compatibles con la sintaxis de **SymPy** y **NumPy**. A continuación, te menciono los principales tipos de funciones que puedes ingresar:

#### **Tipos de funciones compatibles:**

##### **1. Funciones polinómicas:**

Ejemplo:  $x^2 - 4x + 3$ ,  $x^5 + 2x - 7$

2. **Funciones racionales:**

Ejemplo:  $(x^2 - 1)/(x + 1)$

3. **Funciones trigonométricas:**

Ejemplo:  $\sin(x)$ ,  $\cos(x)$ ,  $\tan(x)$ ,  $\sin(x) - x/2$

4. **Funciones exponenciales y logarítmicas:**

Ejemplo:  $\exp(x) - 2$ ,  $\log(x + 1)$

5. **Funciones combinadas:**

Ejemplo:  $\sin(x) + x^2 - \log(x + 2)$

6. **Funciones raíz o radicales:**

Ejemplo:  $\sqrt{x + 4}$ ,  $x^{1/3}$

**Importante:**

- La variable debe ser escrita como x.
- Usa \*\* para potencias, no ^ (por ejemplo, escribe  $x^2$  y no  $x^2$ ).
- No ingreses funciones fuera del dominio válido (por ejemplo,  $\log(x)$  con  $x \leq 0$ ) ya que puede generar errores.

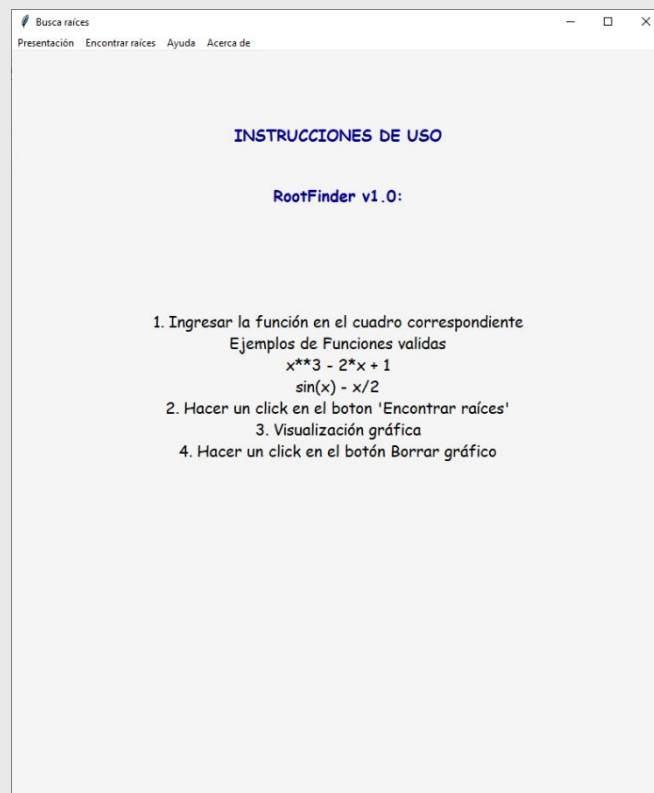
**Método de cálculo de raíces**

Para hallar las raíces utilizamos el método de Newton, mediante la función `opt.newton` pertenece al módulo `scipy.optimize` de Python y se utiliza para **encontrar raíces de funciones no lineales** mediante el **método de Newton-Raphson** o una variante de él. Aplicando la librería `from scipy import optimize as opt`.

### 4.3. Menú Ayuda

El menú "Ayuda" ofrece una guía práctica y accesible que facilita a los estudiantes la comprensión y el uso adecuado del software RootFinder v1.0. Este recurso está diseñado para proporcionar información clara y directa a los usuarios.

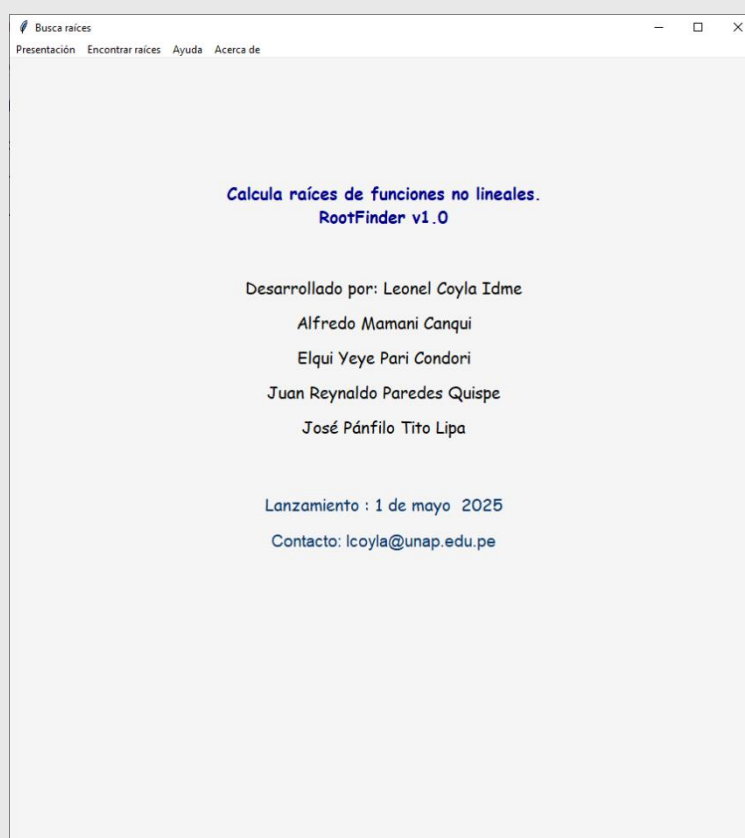
Su objetivo principal es brindar orientación de manera sencilla, permitiendo a los estudiantes familiarizarse con las principales funciones de la aplicación y así aprovechar al máximo las herramientas que ofrece el programa.



### 4.4. Menú Acerca de

En la sección "Acerca de", los usuarios podrán encontrar información importante sobre RootFinder v1.0, una aplicación educativa desarrollada para facilitar la elaboración de gráficos de ecuaciones no lineales y determinar sus respectivas raíces. Este apartado proporciona detalles como el nombre completo del software, la versión actual, el nombre del desarrollador principal, Leonel Coyla Idme, la fecha de lanzamiento, así como los datos de contacto disponibles para consultas, sugerencias o soporte técnico.

Este apartado tiene como finalidad brindar detalles sobre la autoría y procedencia de la herramienta, promoviendo la confianza de los usuarios al utilizarla. Además, facilita un canal de comunicación directo con el desarrollador para resolver dudas, realizar consultas o compartir sugerencias. Gracias a esta sección, los estudiantes, docentes y padres pueden conocer de manera clara la procedencia del software y confirmar que están utilizando una herramienta confiable y diseñada con fines educativos específicos.



## **5. PREGUNTAS FRECUENTES (FAQS) SOBRE EL SOFTWARE**

### **5.1. ¿Qué es RootFinder y cuál es su propósito?**

RootFinder v1.0 es un software educativo diseñado para facilitar el aprendizaje de los estudiantes que cursan la asignatura de Métodos Numéricos. Su propósito principal es graficar ecuaciones no lineales y calcular de manera aproximada la raíz de dichas ecuaciones. Estos procedimientos son fundamentales en el análisis numérico, ya que permiten encontrar soluciones a problemas matemáticos que no se resuelven fácilmente de forma algebraica.

### **5.2. ¿Cuáles son los requisitos mínimos del sistema para ejecutar?**

Para ejecutar correctamente la aplicación RootFinder v1.0, se requiere un sistema operativo Windows 7 o superior, aunque también es compatible con macOS y distribuciones de Linux. Es necesario contar con al menos 2 GB de memoria RAM y un procesador de doble núcleo a 1.5 GHz o superior. Se recomienda una resolución de pantalla mínima de 1024x768 píxeles para una visualización adecuada de la interfaz. Además, debe tener instalado Python 3.8 o superior, junto con las bibliotecas Tkinter, SymPy, NumPy, Matplotlib y Pillow. Es fundamental tener acceso a un entorno gráfico que permita ejecutar interfaces basadas en Tkinter. El espacio en disco requerido es mínimo, aproximadamente 100 MB. Se sugiere mantener el sistema actualizado para garantizar compatibilidad y rendimiento óptimo.

### **5.3. ¿Cómo instalo o configuro RootFinder?**

Para instalar y configurar RootFinder v1.0, primero debes tener Python 3.8 o una versión superior instalado en tu equipo; puedes descargarlo desde la página oficial de Python y asegurarte de marcar la opción "Add Python to PATH" durante la instalación. Luego, crea una carpeta exclusiva donde guardarás el archivo principal RootFinder.py junto con la imagen Graph.png. Abre una terminal y ejecuta el comando `pip install sympy numpy matplotlib pillow` para instalar las bibliotecas necesarias. Una vez instaladas, navega en la terminal hasta la carpeta del proyecto y ejecuta el comando `python RootFinder.py` para iniciar la aplicación. RootFinder funcionará correctamente en sistemas Windows, macOS o Linux que cuenten con al menos 2 GB de RAM y un procesador de doble núcleo. Se recomienda una resolución mínima de pantalla de 1024x768 píxeles para una visualización óptima. Es importante contar con un entorno gráfico disponible, especialmente en sistemas Linux. También puedes crear un acceso directo o archivo .bat para facilitar su ejecución.

#### **5.4. ¿Qué se debe realizar si el software no funciona correctamente?**

Si el software RootFinder v1.0 no funciona correctamente, lo primero que se debe hacer es verificar que todas las bibliotecas necesarias estén correctamente instaladas, incluyendo sympy, numpy, matplotlib, pillow y que se esté utilizando Python 3.8 o una versión superior. También es importante comprobar que no existan errores de escritura en la función ingresada y que la imagen Graph.png esté ubicada en la misma carpeta del programa. Si el problema persiste, se recomienda reiniciar la aplicación y, de ser necesario, el sistema operativo. En caso de que los errores continúen, se debe contactar al desarrollador principal a través del correo electrónico proporcionado en la sección "Acerca de" del software (lcoyla@unap.edu.pe), describiendo detalladamente el inconveniente, incluyendo capturas de pantalla o mensajes de error si es posible.

### **6. ADVERTENCIAS Y SUGERENCIAS DEL SOFTWARE DE APLICACIÓN.**

#### **6.1. Advertencias**

Es importante tener en cuenta que RootFinder v1.0 está diseñado exclusivamente con fines educativos, por lo que no debe utilizarse para aplicaciones técnicas o profesionales que requieran alta precisión en cálculos numéricos. El uso incorrecto del software, como el ingreso de funciones mal escritas o fuera del dominio permitido, puede generar errores o resultados inesperados. Asimismo, modificar el código fuente sin conocimientos adecuados puede afectar su funcionamiento. Se recomienda no eliminar ni cambiar el nombre del archivo de imagen Graph.png, ya que esto podría impedir la correcta visualización en la sección de presentación. Por último, se sugiere utilizar la aplicación en sistemas compatibles y mantener las bibliotecas actualizadas para evitar problemas de ejecución.

#### **6.2. Sugerencias**

Se sugiere a los usuarios explorar y practicar con diversas funciones matemáticas dentro de la aplicación RootFinder v1.0 para familiarizarse con el comportamiento gráfico de las ecuaciones no lineales y el proceso de búsqueda de raíces. Esta interacción no solo refuerza el aprendizaje teórico, sino que también permite desarrollar habilidades analíticas al interpretar los resultados visuales obtenidos. Además, se recomienda utilizar funciones reales aplicadas a contextos cotidianos o problemas universitarios, lo que facilitará una comprensión más significativa y motivadora del tema.

## 7. Código

```
1 import tkinter as tk
2 import sympy as sp
3 import scipy.optimize as opt
4 import numpy as np
5 import matplotlib.pyplot as plt
6 from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
7 from tkinter import PhotoImage
8 from PIL import Image, ImageTk
9
10 ejemploEcuacion = "x**2 - 3*x - 2"
11 entry_function = None
12
13 def on_entry_focus_in(event):
14     if entry_function.get() == ejemploEcuacion:
15         entry_function.delete(0, tk.END)
16         entry_function.config(fg="black")
17
18 def on_entry_focus_out(event):
19     if entry_function.get().strip() == "":
20         entry_function.insert(0, ejemploEcuacion)
21         entry_function.config(fg="gray")
22
23 def encontrarRaices():
24     global entry_function
25     limpiarFrame()
26     for widget in canvasFrame.winfo_children():
27         widget.destroy()
28
29     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
30     tk.Label(frameContenido, text="CALCULA RAÍCES DE FUNCIONES NO LINEALES",
31             font=("Arial", 14, "bold"), fg="#00008B", bg="whitesmoke").pack()
32     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
33     tk.Label(frameContenido, text="Ingresa la función f(x):", font=("Arial", 12, "bold"),
34             fg="#00008B", bg="whitesmoke").pack()
35
36     entry_function = tk.Entry(frameContenido, font=("Arial", 12), fg="#00008B", bg="whitesmoke")
37     entry_function.insert(0, ejemploEcuacion) # Muestra una ecuación como ejemplo de ingreso
38     entry_function.bind("<FocusIn>", on_entry_focus_in)
39     entry_function.bind("<FocusOut>", on_entry_focus_out)
40     entry_function.pack()
41
42 def newtonRaices():
43     try:
44
45         exprCadena = entry_function.get()
46         x = sp.symbols('x')
47         expr = sp.sympify(exprCadena)
48
49         funcionlamdified = sp.lambdify(x, expr, 'numpy')
50         primerad = sp.lambdify(x, sp.diff(expr, x), 'numpy') # primera derivada
51
52         # Para detectar logaritmos
53         if 'log' in exprCadena or 'ln' in exprCadena:
54             valoresx = np.linspace(0.01, 2, 10000)
55         else:
56             valoresx = np.linspace(-10, 10, 4000)
57
58         # Evaluación f(x)
59         valoresy = funcionlamdified(valoresx)
60
61         # Filtrado de valores válidos
62         mascara_valida = np.isfinite(valoresy)
63         valoresx = valoresx[mascara_valida]
64         valoresy = valoresy[mascara_valida]
65
66         ventanas = []
67         #*****
68         for x1, x2, y1, y2 in zip(valoresx[:-1], valoresx[1:], valoresy[:-1], valoresy[1:]):
69             if y1 == 0:
70                 ventanas.append(x1)
```



```

71         elif y1 * y2 < 0:
72             x0 = (x1 + x2) / 2
73             try:
74                 raiz = opt.newton(funcionlamdified, x0, fprime=primerad, tol=1e-10, maxiter=50)
75             except Exception:
76                 raiz = x0 # En caso de error, usar punto medio
77             ventanas.append(raiz)
78
79         # Revisamos el último valor
80         if valores[-1] == 0:
81             ventanas.append(valores[-1])
82
83         # Eliminar raíces muy cercanas
84         listav = []
85         tol = 1e-5
86
87         listav += [r for r in ventanas if not any(abs(r - existente) < tol for existente in listav)]
88
89         if listav:
90             resultados = "Raíces:\n" + " ".join(f"{r:.6f}" for r in listav)
91             ventanaLabel.config(text=resultados.strip(), font=("Arial", 12, "bold"))
92         else:
93             ventanaLabel.config(text="No tiene raíces")
94
95         muestraPlot(expr, listav)
96
97     except Exception as e:
98         ventanaLabel.config(text=f"Error: Ingrese la ecuación correctamente", font=("Arial"), fg="red")
99
100 def borrarGrafico():
101     entry_function.delete(0, tk.END)
102     ventanaLabel.config(text="")
103     infoLabel.config(text="")
104     for widget in canvasFrame.winfo_children():
105         widget.destroy()
106
107     # Botón Encontrar raíces
108     tk.Button(
109         frameContenido, text="Calcular raíces", command=newtonRaices,
110         fg="white", bg="orange", font=("Arial", 12, "bold"),
111         height=2, width=20
112     ).pack(side="left", padx=50, pady = 20)
113
114     # Botón Borrar grafico
115     tk.Button(
116         frameContenido, text="Borrar gráfico", command=borrarGrafico,
117         fg="white", bg="orange", font=("Arial", 12, "bold"),
118         height=2, width=20
119     ).pack(side="left", padx=50, pady = 20)
120
121 def muestraPlot(exprFunc, ventanas):
122     x = sp.symbols('x')
123     funcionlamdified = sp.lambdify(x, exprFunc, 'numpy')
124
125     valoresx = np.linspace(-10, 10, 400)
126     valoresx_validos = valoresx[valoresx > 0] # positivo
127     valoresy = funcionlamdified(valoresx)
128
129     fig, eje = plt.subplots(figsize=(10, 8))
130     eje.plot(valoresx, valoresy, label=f"f(x) = {exprFunc}")
131
132     eje.set_xticks(np.arange(-10, 11, 1))
133     eje.set_yticks(np.arange(-10, 11, 1))
134
135     # Color de ejes
136     eje.axhline(0, color='darkgray', linewidth=1.5)
137     eje.axvline(0, color='darkgray', linewidth=1.5)
138     #Color de etiqueta en el eje
139     eje.set_xlabel("Eje X", color='blue')
140     eje.set_ylabel("Eje Y", color='blue')

```

```

141 #Color de fondo
142 fig.patch.set_facecolor("whitesmoke")
143 eje.set_facecolor("whitesmoke")
144 #Color de numeros en los ejes
145 eje.tick_params(axis='x', colors='black')
146 eje.tick_params(axis='y', colors='black')
147
148 for ventana in ventanas:
149     eje.axvline(ventana, color='red', linestyle='--')
150     eje.plot(ventana, 0, 'ro', markersize=8)
151
152 eje.set_xlim([-10, 10])
153 eje.set_ylim([-10, 10])
154 eje.grid(True, linestyle='-', linewidth=0.5)
155 #eje.grid(True)
156 eje.legend()
157
158 for widget in canvasFrame.winfo_children():
159     widget.destroy()
160
161 canvas = FigureCanvasTkAgg(fig, master=canvasFrame)
162 canvas.draw()
163 canvas.get_tk_widget().pack()
164
165 def limpiarFrame():
166     for widget in frameContenido.winfo_children():
167         widget.destroy()
168
169 def Presentacion():
170     global frameContenido, ventanaLabel, infoLabel, canvasFrame
171     limpiarFrame()
172     for widget in canvasFrame.winfo_children():
173         widget.destroy()
174
175     ventanaLabel.config(text="")
176     infoLabel.config(text="")
177
178     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
179     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
180     tk.Label(frameContenido, text="BIENVENIDOS A LA APLICACIÓN", font=("Comic Sans MS", 18, "bold"),
181             fg="#00008B", bg="whitesmoke").pack()
182     tk.Label(frameContenido, text="RAÍCES DE FUNCIONES NO LINEALES",
183             font=("Arial", 14, "bold"), fg="#00008B", bg="whitesmoke").pack()
184     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
185     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
186
187     try:
188         image = Image.open("RootFinder.png")
189         image = image.resize((550, 400))
190         imageTk = ImageTk.PhotoImage(image)
191         imageLabel = tk.Label(frameContenido, image=imageTk, bg="whitesmoke")
192         imageLabel.image = imageTk
193         imageLabel.pack(pady=20)
194     except Exception as e:
195         print(f"Error al cargar la imagen: {e}")
196
197     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
198     tk.Label(frameContenido, text="", font=("Arial", 12, "bold"), fg="#00008B", bg="whitesmoke").pack()
199     tk.Label(frameContenido, text="RootFinder v1.0", font=("Comic Sans MS", 20, "bold"),
200             fg="#00008B", bg="whitesmoke").pack()
201
202 def Ayuda():
203     limpiarFrame()
204     for widget in canvasFrame.winfo_children():
205         widget.destroy()
206
207     ventanaLabel.config(text="")
208     infoLabel.config(text="")
209
210     tk.Label(frameContenido, text="\n\nINSTRUCCIONES DE USO", bg="whitesmoke",

```

```

211     font=("Comic Sans MS", 14,"bold"), fg="#00008B").pack(pady=20)
212 tk.Label(frameContenido, text="RootFinder v1.0:\n\n", bg="whitesmoke",
213     font=("Comic Sans MS", 14,"bold"), fg="#00008B").pack(pady=20)
214 tk.Label(frameContenido, text="\n1. Ingresar la función en el cuadro correspondiente"
215     f"\nEjemplos de Funciones validas\nx**3 - 2*x + 1\nsin(x) - x/2\n"
216     f"2. Hacer un click en el boton 'Encontrar raíces'"
217     f"\n3. Visualización gráfica\n4. Hacer un click en el botón Borrar gráfico",bg="whitesmoke",
218     font=("Comic Sans MS", 14)).pack(pady=20)
219
220 def Acercade():
221     limpiarFrame()
222     for widget in canvasFrame.wininfo_children():
223         widget.destroy()
224
225     ventanaLabel.config(text="")
226     infoLabel.config(text="")
227
228     tk.Label(frameContenido, text="\n\n\n\nCalcula raíces de funciones no lineales."
229         f"\nRootFinder v1.0", bg="whitesmoke", font=("Comic Sans MS", 14,"bold"), fg="#00008B").pack(pady=20)
230
231     textos = [
232         "\nDesarrollado por: Leonel Coyla Idme",
233         "Alfredo Mamani Canqui",
234         "Elqui Yeye Pari Condori",
235         "Juan Reynaldo Paredes Quispe",
236         "José Pánfilo Tito Lipa",
237     ]
238
239     for texto in textos:
240         tk.Label(frameContenido, text=texto, bg="whitesmoke", font=("Comic Sans MS", 14)).pack(pady=4)
241
242     labelAcercade = tk.Label(frameContenido, text= "\n\nLanzamiento : 1 de mayo 2025",
243         font=("Comic Sans MS", 14),fg="#003366",bg="whitesmoke")
244     labelAcercade.pack(pady=(1,10))
245     labelAcercade = tk.Label(frameContenido, text= "Contacto: lcoyla@unap.edu.pe",
246         font=("Comic Sans MS", 14),fg="#003366",bg="whitesmoke")
247     labelAcercade.pack(pady=(1,10))
248
249 def crearInterfaz():
250     global frameContenido, ventanaLabel, infoLabel, canvasFrame
251
252     ventana = tk.Tk()
253     ventana.title("Busca raíces")
254     ventana.geometry("1000x1000")
255     ventana.config(bg="whitesmoke")
256
257     menuBar = tk.Menu(ventana)
258     ventana.config(menu=menuBar)
259
260     menuBar.add_command(label="Presentación", command=Presentacion)
261     menuBar.add_command(label="Encontrar raíces", command=encontrarRaices)
262     menuBar.add_command(label="Ayuda", command=Ayuda)
263     menuBar.add_command(label="Acerca de", command=Acercade)
264
265     mainFrame = tk.Frame(ventana, bg="whitesmoke")
266     mainFrame.pack(pady=10)
267
268     frameContenido = tk.Frame(mainFrame, bg="whitesmoke")
269     frameContenido.pack()
270
271     ventanaLabel = tk.Label(mainFrame, text="", bg="whitesmoke")
272     ventanaLabel.pack()
273
274     infoLabel = tk.Label(mainFrame, text="", fg="blue", bg="whitesmoke")
275     infoLabel.pack()
276
277     canvasFrame = tk.Frame(mainFrame, bg="whitesmoke")
278     canvasFrame.pack()
279
280     encontrarRaices()

```

```
281
282     ventana.mainloop()
283
284 def main():
285     crearInterfaz()
286
287 if __name__ == "__main__":
288     main()
289
```