

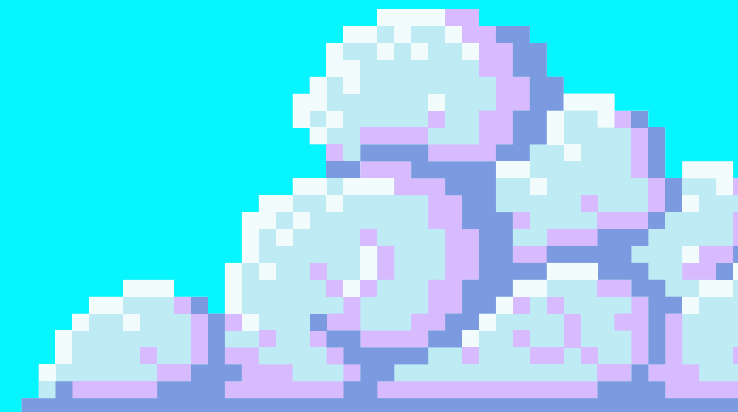
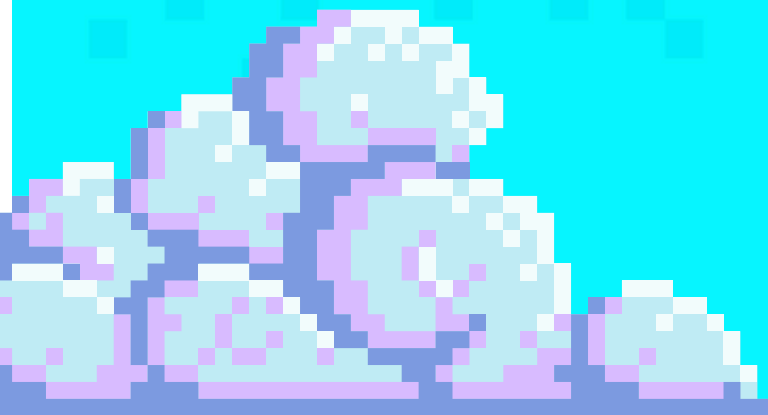
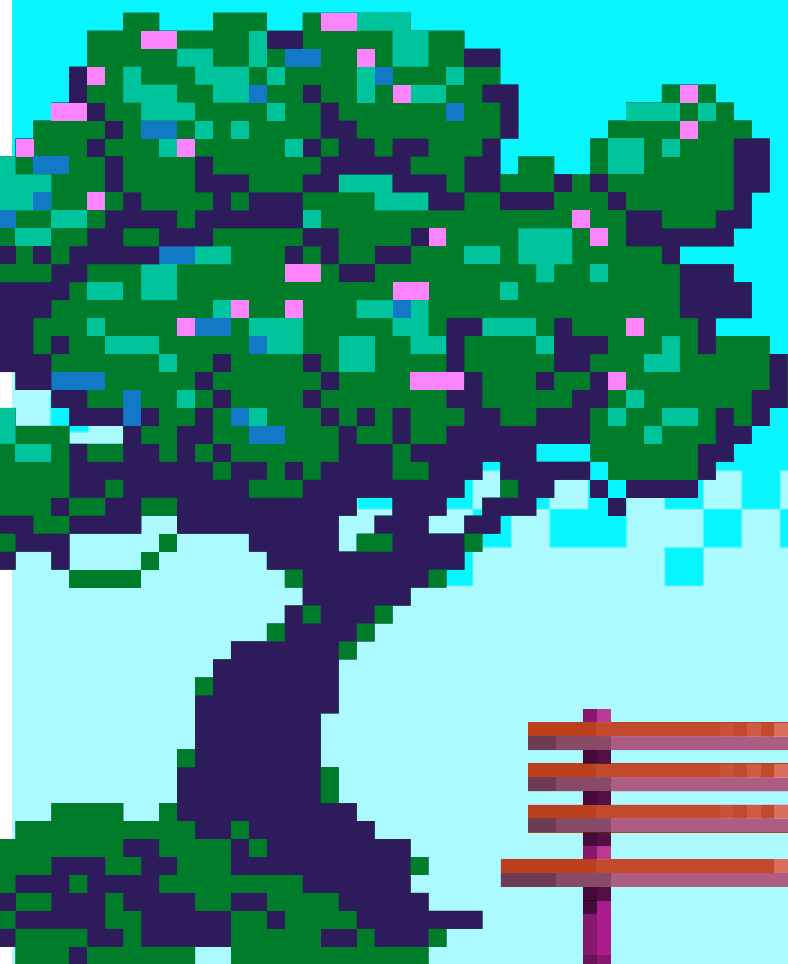
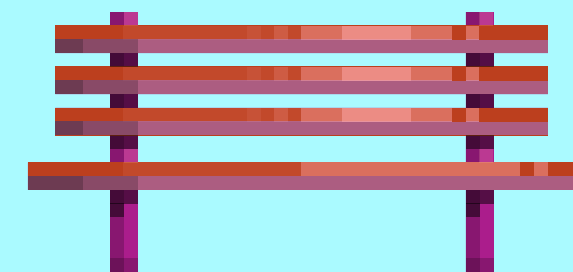
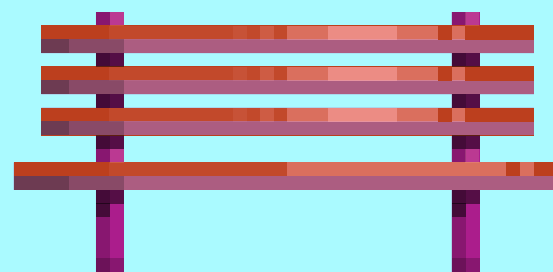
PROYECTO BIMESTRAL

METODOS NUMERICOS



AHORRO CON INTERES COMPUESTO

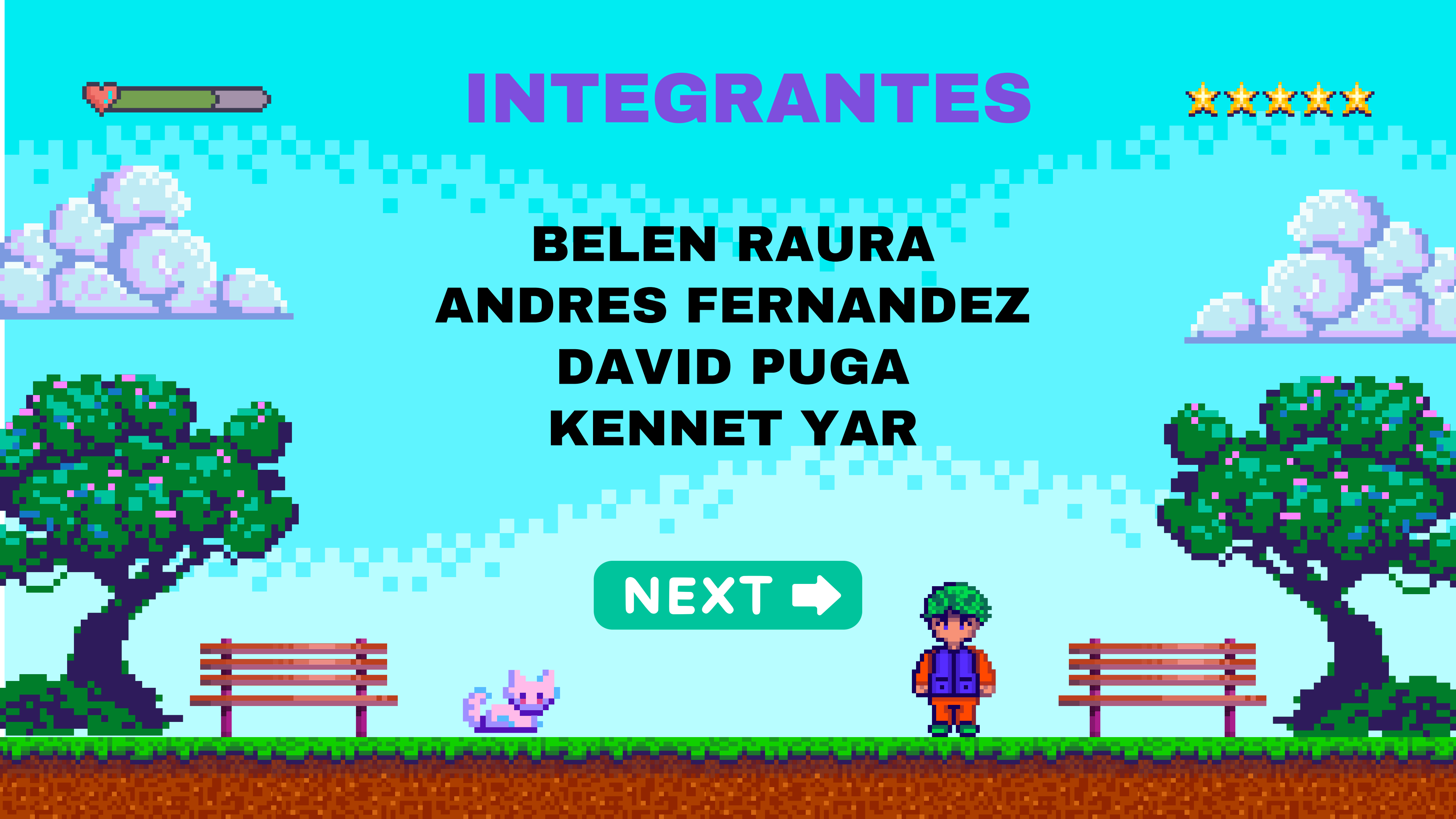
START



INTEGRANTES

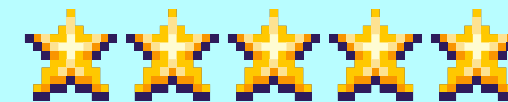
BELEN RAURA
ANDRES FERNANDEZ
DAVID PUGA
KENNET YAR

NEXT ➡





OBJETIVOS

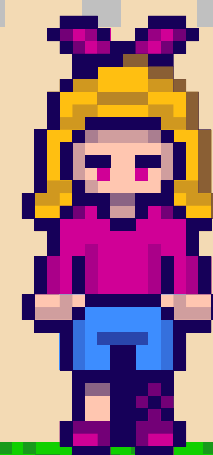
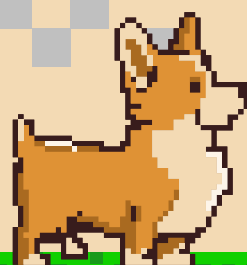


DESARROLLAR UNA HERRAMIENTA FINANCIERA INTUITIVA Y FÁCIL DE USAR QUE PERMITA A CUALQUIER USUARIO CALCULAR TASAS DE INTERÉS DE MANERA PRECISA Y EFICIENTE, UTILIZANDO EL MÉTODO DE LA SECANTE.

FACILITAR LA INTERACCIÓN CON LOS USUARIOS MEDIANTE UNA INTERFAZ GRÁFICA INTUITIVA, PERMITIENDO EL INGRESO DE DATOS Y LA VISUALIZACIÓN DE RESULTADOS.

CREAR UNA HERRAMIENTA QUE PERMITA PERSONALIZAR FRECUENCIAS DE CÁLCULO, COMO APORTES SEMANALES, MENSUALES O TRIMESTRALES, PARA ADAPTARSE A DIFERENTES NECESIDADES.

NEXT ➡



CÓDIGO



```
def f(i, V0, A, n, Vf):  
    return V0 * (1 + i)**n + A * ((1 + i)**n - (1 + i)) / i - Vf  
  
# Método de la secante  
def metodo_secante(V0, A, n, Vf, i0, i1, tol=1e-10, max_iter=100):  
    iteraciones = 0  
    while iteraciones < max_iter:  
        # Calculamos los valores de la función en i0 y i1  
        f_i0 = f(i0, V0, A, n, Vf)  
        f_i1 = f(i1, V0, A, n, Vf)  
  
        # Comprobamos si la diferencia entre las funciones es pequeña para evitar división por 0  
        if abs(f_i1 - f_i0) < tol:  
            return None  
  
        # Calculamos el nuevo valor de i usando la fórmula del método de la secante  
        i_next = i1 - f_i1 * (i1 - i0) / (f_i1 - f_i0)  
  
        # Comprobamos si la diferencia entre i_next y i1 es suficientemente pequeña  
        if abs(i_next - i1) < tol:  
            return i_next  
  
        # Actualizamos i0 e i1 para la siguiente iteración  
        i0, i1 = i1, i_next  
        iteraciones += 1  
  
    return None
```



CÓDIGO



```
def iniciar_simulacion():
    try:
        # Obtenemos los valores introducidos por el usuario
        V0 = float(entry_V0.get())
        A = float(entry_A.get())
        n = int(entry_n.get())
        Vf = float(entry_Vf.get())
        frecuencia = combo_frecuencia.get()

        # Validar que los valores sean positivos
        if V0 < 0 or A < 0 or n <= 0 or Vf < 0:
            messagebox.showerror("Error", "Por favor, ingresa valores positivos para todos los campos.")
            return

        # Inicializamos los valores para el método de la secante
        i0 = 0.05 # Valor inicial 1
        i1 = 0.08 # Valor inicial 2

        # Calculamos la tasa de interés usando el método de la secante
        i_calculado = metodo_secante(V0, A, n, Vf, i0, i1)

        if i_calculado is None:
            messagebox.showerror("Error", "No se pudo encontrar la tasa de interés. Intenta con otros valores iniciales.")
            return
```



CÓDIGO



```
def iniciar_simulacion():  
  
    # Ajuste de la cantidad de periodos dependiendo de la frecuencia de los aportes  
    if frecuencia == 'Mensual':  
        n = n * 4 # Suponemos que la cantidad de periodos es en meses, multiplicamos por 4 (para semanales)  
        A = A * 4 # Convertir aportes semanales a mensuales  
    elif frecuencia == 'Bimestral':  
        n = n * 2 # Convertir de semanas a bimestres  
        A = A * 2 # Convertir aportes semanales a bimestrales  
    elif frecuencia == 'Trimestral':  
        n = n * 4 / 3 # Convertir de semanas a trimestres  
        A = A * 4 / 3 # Convertir aportes semanales a trimestrales
```



CÓDIGO



```
def iniciar_simulacion():

    # Calcular los resultados para cada período
    capital = V0 # Capital inicial (se mantiene igual)
    historial = []
    for t in range(1, n + 1):
        # Calculamos la ganancia en base al capital y la tasa de interés
        ganancia = capital * i_calculado
        total = capital + ganancia # El total es capital + ganancia

        # Sumar el aporte al total, y usar el total como el nuevo capital para el siguiente periodo
        if t > 1: # A partir del segundo periodo, sumamos el aporte
            capital = total + A
            aporte = A
        else:
            capital = total
            aporte = 0 # El aporte es 0 en la primera iteración

        # Guardamos los resultados en el historial
        historial.append((t, aporte, round(capital - ganancia , 2), round(ganancia, 2), round(capital, 2)))
    Interes = float(i_calculado) * n
    # Mostrar la tasa de interés calculada
    label_resultado.config(text=f"Tasa de interés calculada: {Interes:.6f}")
    # Mostrar el historial en una nueva ventana
    mostrar_historial(historial)

except ValueError:
    messagebox.showerror("Error", "Por favor, ingresa valores válidos.")
```



CÓDIGO



```
✓ def limpiar_valores():  
    # Limpiar todos los campos de entrada  
    entry_V0.delete(0, tk.END)  
    entry_A.delete(0, tk.END)  
    entry_n.delete(0, tk.END)  
    entry_Vf.delete(0, tk.END)  
    # Restablecer el valor predeterminado del combobox  
    combo_frecuencia.set("Semanal")  
    # Borrar el texto del resultado  
    label_resultado.config(text="Tasa de interés calculada: ")
```

```
# Función para mostrar el historial en una nueva ventana  
def mostrar_historial(historial):  
    # Crear una nueva ventana  
    ventana_historial = tk.Toplevel(root)  
    ventana_historial.title("Historial de Resultados")
```




SUSTENTO MATEMATICO



1.

$$V_f = V_0(1 + i)^n + A \sum_{k=1}^n (1 + i)^{n-k}$$

Donde:

- V_f = Valor futuro.
- V_0 = Valor inicial o capital de partida.
- A = Aporte periódico.
- i = Tasa de interés por periodo.
- n = Número de periodos.
- k = Índice de cada periodo.



SUSTENTO MATEMATICO



2.

$$V_f = V_0(1 + i)^n + A \frac{(1 + i)^n - 1}{i}$$

Donde:

- V_f = Valor futuro.
- V_0 = Valor inicial o capital de partida.
- A = Aporte periódico.
- i = Tasa de interés por periodo.
- n = Número de periodos.





SUSTENTO MATEMATICO



Dado que los aportes se realizan al final de cada periodo, se debe restar un término que represente el primer aporte, no afectado por los intereses del primer periodo:

$$-A(1 + i)$$

3.

$$V_0(1 + i)^n + A \frac{(1 + i)^n - 1}{i} - A(1 + i) = V_f$$



SUSTENTO MATEMATICO



4.

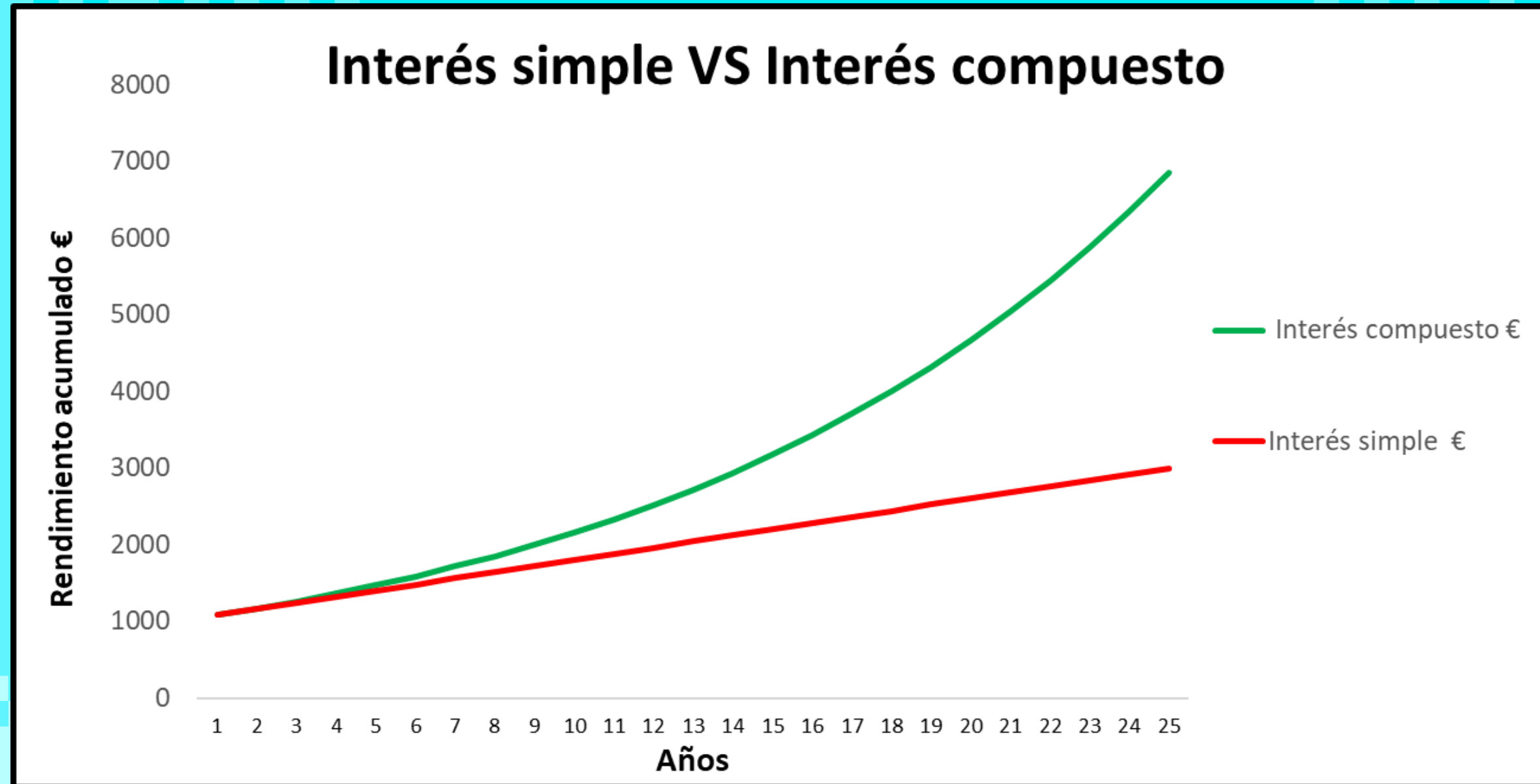
$$V_0(1+i)^n + A \frac{(1+i)^n - 1}{i} - A(1+i) = V_f$$

5.

$$f(i) = V_0(1+i)^n + A \frac{(1+i)^n - (1+i)}{i} - V_f = 0$$

Dado que todos los valores son conocidos excepto i , podemos definir una función en términos de i , de manera que se convierta en el único parámetro desconocido.

Para calcular el interés, utilizamos el método de la secante.



EXPLICACION CODIGO



```
# Método de la secante
def metodo_secante(V0, A, n, Vf, i0, i1, tol=1e-10, max_iter=100):
    iteraciones = 0
    while iteraciones < max_iter:
        # Calculamos los valores de la función en i0 y i1
        f_i0 = f(i0, V0, A, n, Vf)
        f_i1 = f(i1, V0, A, n, Vf)

        # Comprobamos si la diferencia entre las funciones es pequeña para evitar división por 0
        if abs(f_i1 - f_i0) < tol:
            return None

        # Calculamos el nuevo valor de i usando la fórmula del método de la secante
        i_next = i1 - f_i1 * (i1 - i0) / (f_i1 - f_i0)

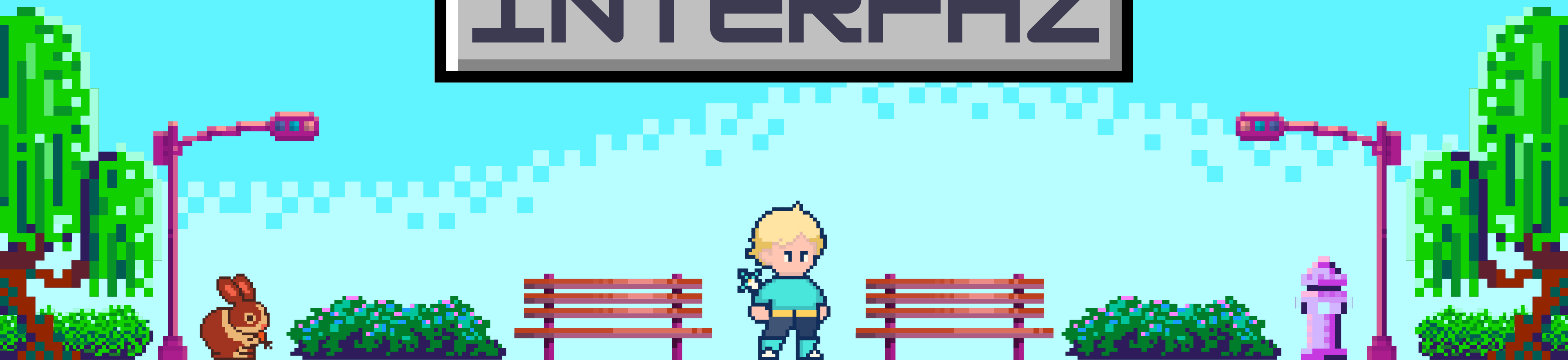
        # Comprobamos si la diferencia entre i_next y i1 es suficientemente pequeña
        if abs(i_next - i1) < tol:
            return i_next

        # Actualizamos i0 e i1 para la siguiente iteración
        i0, i1 = i1, i_next
        iteraciones += 1

    return None
```



INTERFAZ



INTERFAZ



TKINDER

BIBLIOTECA UTILIZADA
PARA CREAR UN GUI
PARA LA SIMULACION.

```
# Crear la ventana principal
root = tk.Tk()
root.title("Simulación de Interés")

center_window(root)

# Crear un marco principal
frame_principal = ttk.Frame(root, padding="10")
frame_principal.grid(row=0, column=0)

bold_font = ("Helvetica", 9, "bold")

# Etiquetas y campos de entrada
ttk.Label(frame_principal, text="Valor Inicial (V0):", font=bold_font).grid(row=0, column=0, sticky="w", pady=5)
entry_V0 = ttk.Entry(frame_principal)
entry_V0.grid(row=0, column=1, pady=5)

ttk.Label(frame_principal, text="Aporte Periódico (A):", font=bold_font).grid(row=1, column=0, sticky="w", pady=5)
entry_A = ttk.Entry(frame_principal)
entry_A.grid(row=1, column=1, pady=5)

ttk.Label(frame_principal, text="Número de Periodos (n):", font=bold_font).grid(row=2, column=0, sticky="w", pady=5)
entry_n = ttk.Entry(frame_principal)
entry_n.grid(row=2, column=1, pady=5)

ttk.Label(frame_principal, text="Valor Final (Vf):", font=bold_font).grid(row=3, column=0, sticky="w", pady=5)
entry_Vf = ttk.Entry(frame_principal)
entry_Vf.grid(row=3, column=1, pady=5)

ttk.Label(frame_principal, text="Frecuencia de Aportes:", font=bold_font).grid(row=4, column=0, sticky="w", pady=5)
combo_frecuencia = ttk.Combobox(frame_principal, values=["Semanal", "Mensual", "Bimestral", "Trimestral"])
combo_frecuencia.set("Semanal") # Valor predeterminado
combo_frecuencia.grid(row=4, column=1, pady=5)
```


INTERFAZ



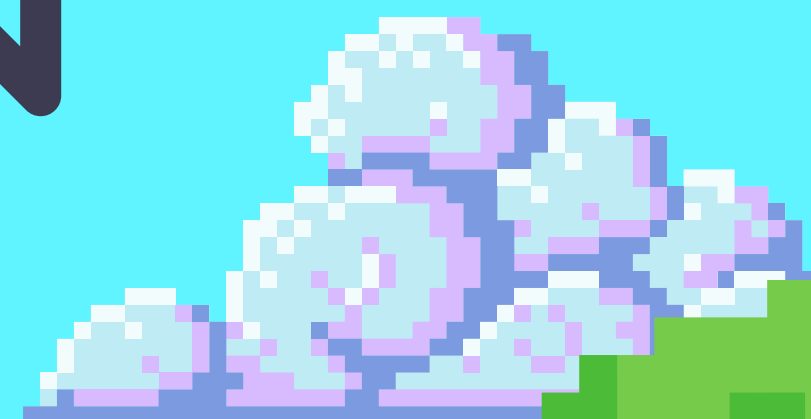
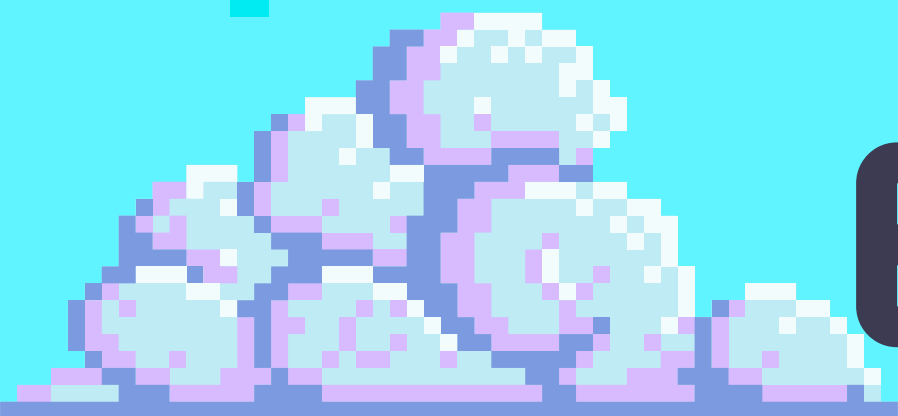
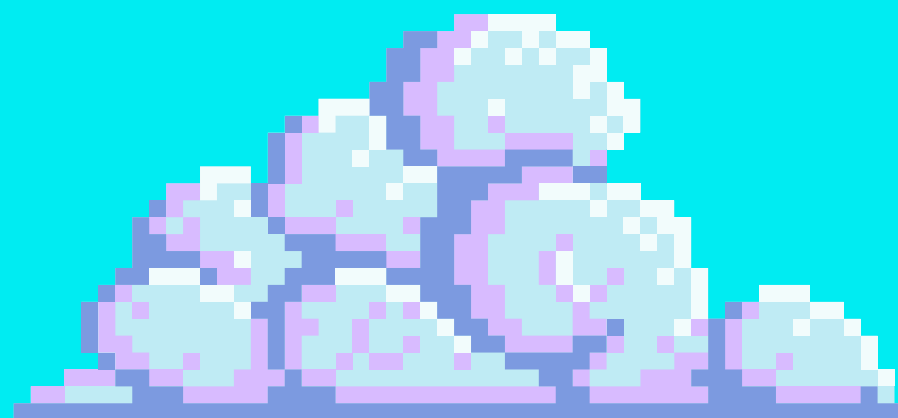
TKINDER

BIBLIOTECA UTILIZADA
PARA CREAR UN GUI
PARA LA SIMULACION.

```
# Botones en la interfaz
ttk.Button(frame_principal, text="Iniciar Simulación", command=iniciar_simulacion).grid(row=5, column=0, pady=10, padx=5)
ttk.Button(frame_principal, text="Limpiar Valores", command=limpiar_valores).grid(row=5, column=1, pady=10, padx=5)

# Etiqueta para mostrar el resultado
label_resultado = ttk.Label(frame_principal, text="Tasa de interés calculada: ", font=bold_font)
label_resultado.grid(row=6, column=0, columnspan=2, pady=10)

root.mainloop()
```



EJECUCION
EJEMPLO:



DATOS:




Ejemplo

Dado un depósito inicial de 100 dólares, aportes semanales de 5 dólares y una tasa de interés anual del 8%, la tabla de cálculo sería la siguiente:

Semana	Aporte (\$)	Capital (\$)	Ganancia (\$)	Total (\$)
1	100	100	0.15	100.15
2	5	105.15	0.16	105.31
3	5	110.31	0.17	110.48
4	5	115.48	0.18	115.66
5	5	120.66	0.19	120.85
...				
51	5	367.65	0.57	368.22
52	5	373.22	0.57	373.79

VALORES DEL EJEMPLO:



 Simulación de Interés — □ ×

Valor Inicial (V_0):	<input type="text" value="100"/>
Aporte Periódico (A):	<input type="text" value="5"/>
Número de Periodos (n):	<input type="text" value="52"/>
Valor Final (V_f):	<input type="text" value="373.79"/>
Frecuencia de Aportes:	<input type="text" value="Semanal"/>

Tasa de interés calculada:



COMPILACIÓN



Simulación de Interés

—

□

×

Valor Inicial (V0):

100

Aporte Periódico (A):

5

Número de Periodos (n):

52

Valor Final (Vf):

373.79

Frecuencia de Aportes:

Semanal

▼

Iniciar Simulación

Limpiar Valores

Tasa de interés calculada: 8.000582

Historial de Resultados

—

□

×

Periodo	Aporte	Capital	Ganancia	Total
43	5.0	323.11	0.49	323.6
44	5.0	328.6	0.5	329.09
45	5.0	334.09	0.51	334.6
46	5.0	339.6	0.51	340.11
47	5.0	345.11	0.52	345.64
48	5.0	350.64	0.53	351.17
49	5.0	356.17	0.54	356.71
50	5.0	361.71	0.55	362.26
51	5.0	367.26	0.56	367.82
52	5.0	372.82	0.57	373.38



GRACIAS

