MODELOS Y SIMULACIÓN

Tomás Licciardi

Ingeniería en Informática

2024

INFORME

1-

A) Material

Poliestireno expandido (ETS), debido a que este material es ligero y económico. También cuenta con buenas propiedades de aislamiento térmico.

B) Forma del recipiente

Recipiente cilíndrico de 1000cc de volumen

C) Propósito

El propósito que se elegirá es calentar el agua para infusiones porque todas las mañanas me tomo 1 taza de té

D) Fluido a calentar

Agua, ya que es el fluido que más utilizo en mi día a día.

E) Tiempo que se espera alcanzar la temperatura

El tiempo que se espera para alcanzar los 80° de temperatura es de 240 segundos (4 minutos)

F) Tensión de alimentación

Se utilizarán 220 Volts

G) Cálculo de la Resistencia

Se calcula la resistencia de la siguiente forma:

Sabiendo que:
Q = m . c . Δt

m = 1kg

c = $\frac{4186\ julios}{kg\ °C}$

Δt = 80°C - 20°C = 60°C

Q= $\frac{1kg\ .4186J\ .\ 60°C}{kg\ °C}$

Q= 251.160J

Potencia:

P= $\frac{Q}{t} = \frac{251.160J}{240s}$ = 1046,5W

Intensidad de Corriente:

I = $\frac{P}{V} = \frac{1046,5W}{220V}$ = 4,75A

Resistencia eléctrica:

R = $\frac{V}{I} = \frac{220V}{4,75A}$ =46,31Ω

H) Temperatura inicial del fluido

La temperatura inicial es de 20° → 293,15° K

I) Temperatura del entorno

La temperatura del entorno es de 20° → 293,15° K

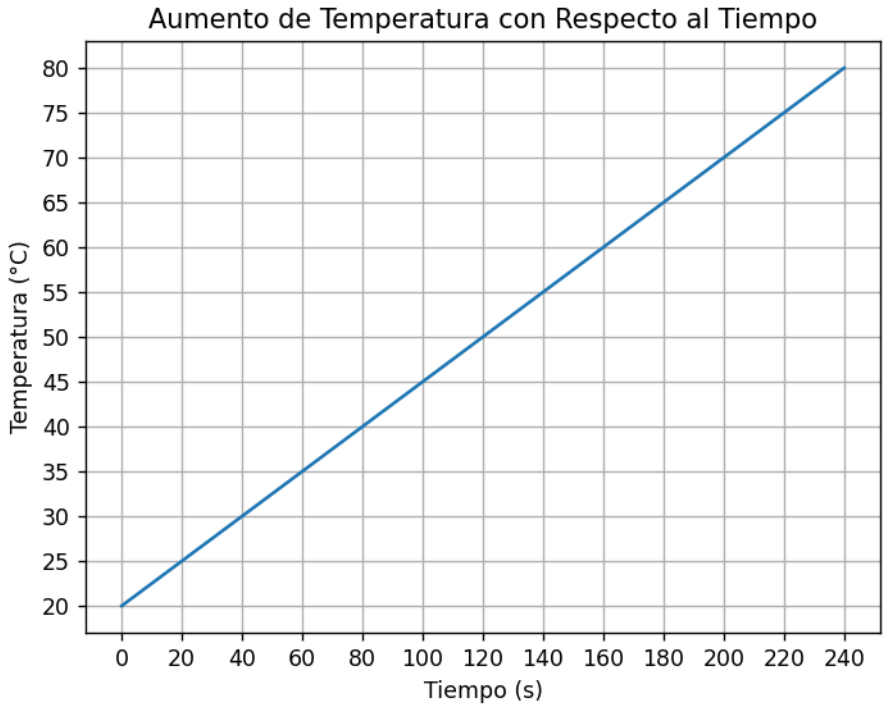
J) Aumentar la temperatura en un segundo

Sabiendo que:

Q = P.t → Q = 1046.5W . 1s= 1046.5J

Δt = $\frac{1046.5J}{1kg \cdot \frac{4186J}{kg°C}}$ = 0, 25007 °C

Como se puede observar se aumentan 0,25007°C por cada segundo que pasa. Logrando así que 0,25007 °C . 240s alcanzan los 80°C respectivamente.



Características del recipiente

3-

Para el material se elige un espesor de 1mm, altura de 35cm y diámetro de 16cm

Teniendo en cuenta que el Coeficiente de Conductividad Térmica del Telgopor es de 0,035 $\frac{W}{k\,.m}$

Superficie = 2 * π $*$ $\frac{diametro}{2}$ $*$ $(\frac{diametro}{2}$ + altura) = 0,2161m^2

Y la fórmula para calcular el calor perdido es:

Perdida de calor = cct .$\frac{superficie}{espesor}$ = 0,035 $\frac{W}{k\,.m}$ $*$ $\frac{0{,}0942m^2}{0{,}002m}$ = 7.5635 $\frac{W}{K}$

Luego tomamos en cuenta temperatura inicial, final y exterior.

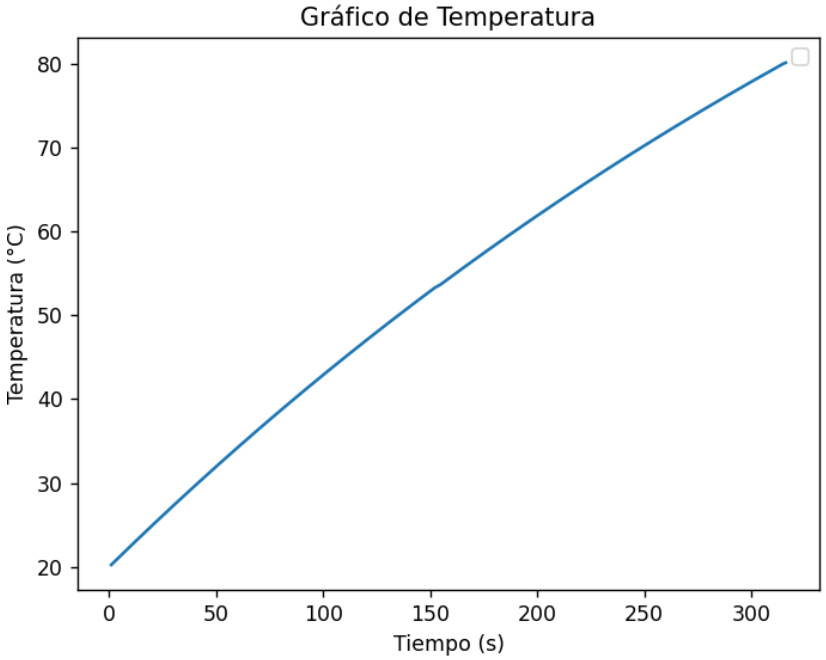
Se utilizaron los siguientes casos:

4-

Temperatura inicial = 20

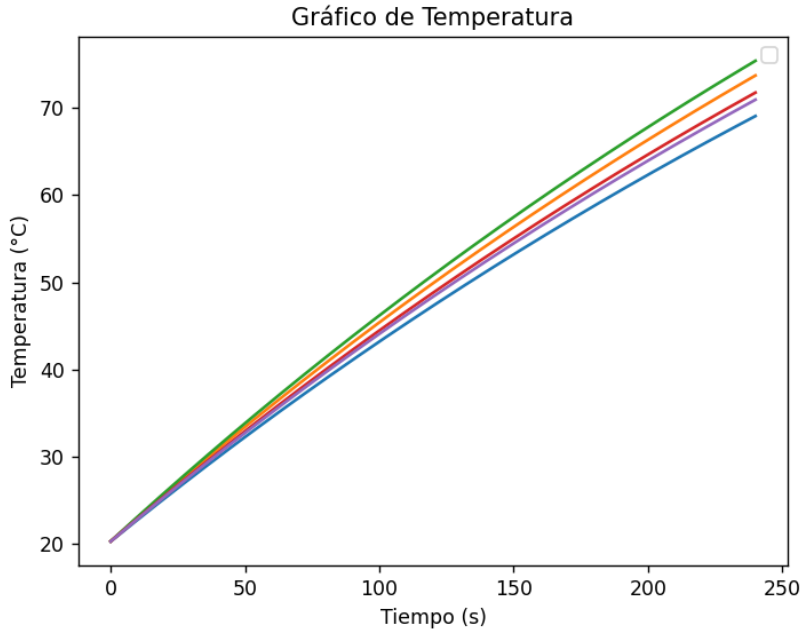
Temperatura final = 80

Temperatura exterior = 20

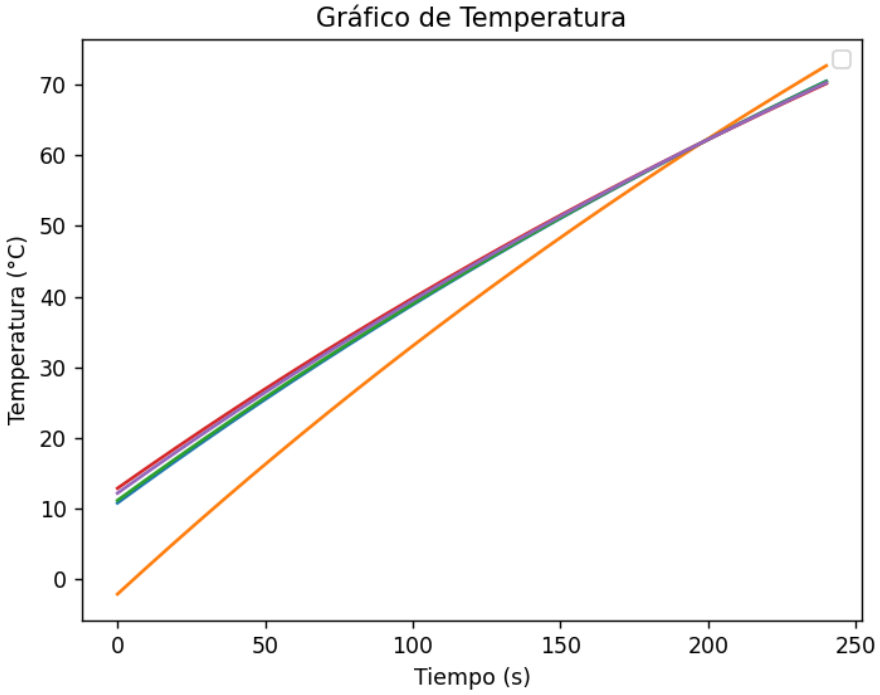


5-

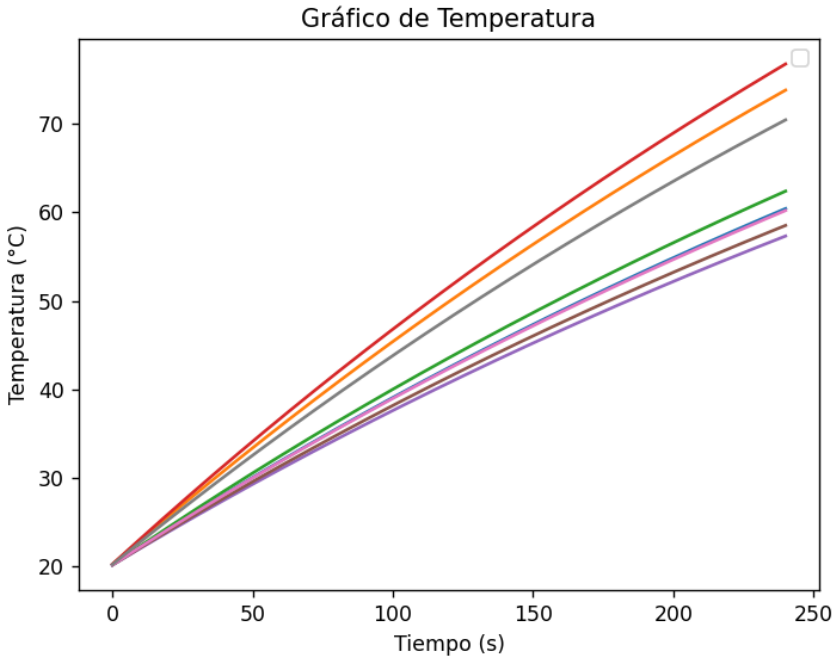
A) Ingresando valores para una distribución uniforme de 5 valores de resistencias cercano en un **rango de 40-50.**



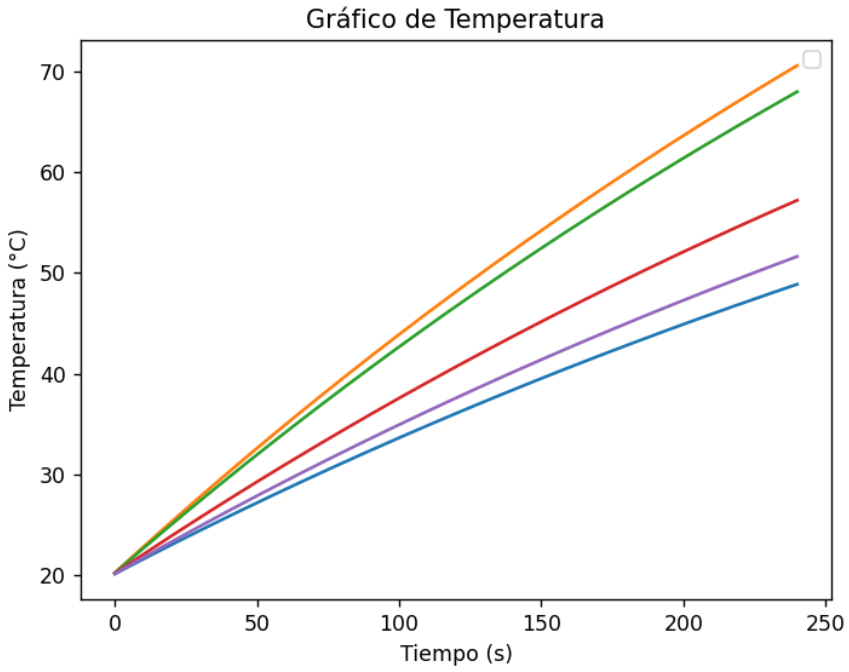
B) Ingresando valores para una distribución normal de 5 valores de temperaturas iniciales del agua con **Media 10, desvío estándar 5.**



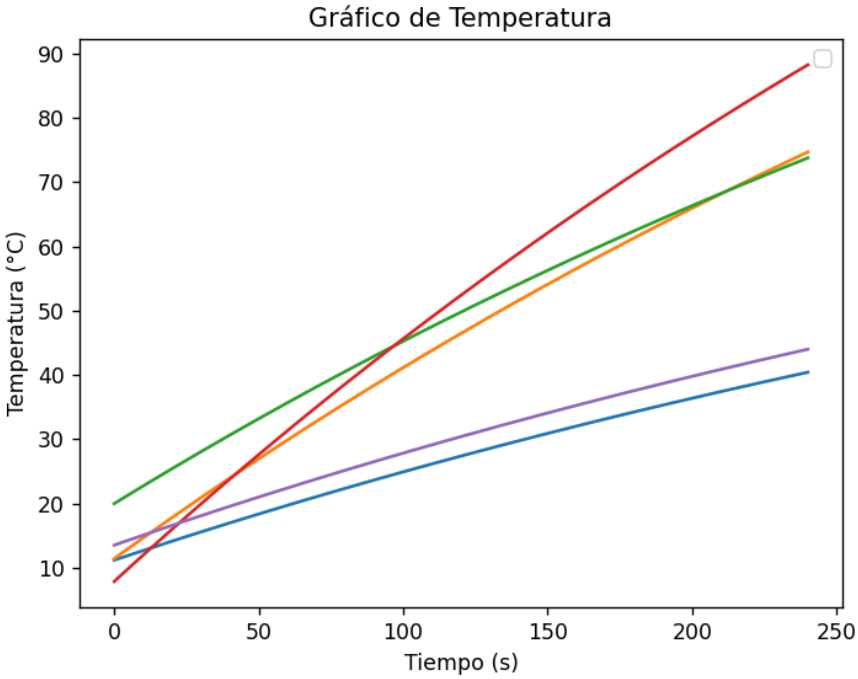
C) Ingresando valores para una distribución uniforme de 8 valores de temperaturas exterior **entre -20 y 50 grados.**



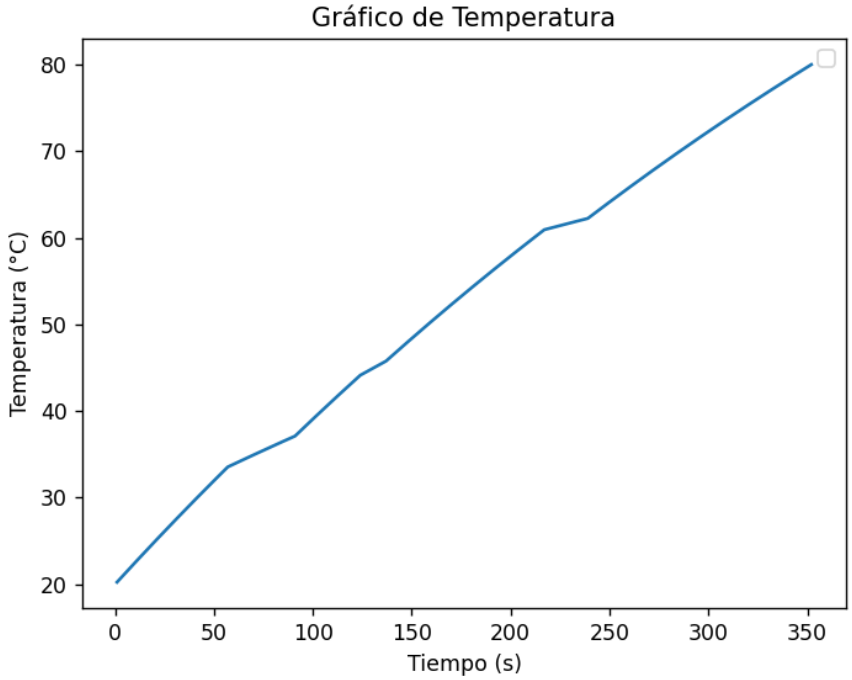
D) Ingresando valores para una distribución normal de 5 valores de tensión de alimentación de **Media 220, Desviación estándar 40**



E) Simulaciones que contengan todas las familias de curvas previas.



6- Simulación de un fenómeno estocástico que tiene una probabilidad de ocurrencia de 1/300 en cada tick de tiempo. Con variables aleatorias, un descenso de temperatura entre -50 y 0 grados con una duración de entre 1 y 40 segundos.



Adjunto Código completo

```
import random
import matplotlib.pyplot as plt
import math
import numpy as np
```

```
# Clase Proyecto y sus métodos como en tu código original...
class Proyecto:
    capacidad_calorifica = 4186  # J/kg°C

    def __init__(self, temperatura_inicial, temperatura_final,
temperatura_exterior, resistencia=None,voltaje=None):
        self.temperatura_inicial = temperatura_inicial  # °C
        self.temperatura_final = temperatura_final  # °C
        self.temperatura_exterior = temperatura_exterior  # °C
        self.capacidad = 1000  # Gramos (volumen de agua en el cilindro)
        self.voltaje = voltaje if voltaje is not None else 220  # Voltios
        self.tiempo = 240  # Segundos
        self.altura = 35  # cm
        self.radio = 8  # cm
        self.espesor = 0.001  # Metros
        self.conductividad_de_telgopor = 0.035  # Conductividad térmica
del telgopor
        self.superficie = (
            (2 * math.pi * (self.radio ** 2) + 2 * math.pi * self.radio *
self.altura) / 10000
        )  # Área de la superficie del cilindro
        self.resistencia = resistencia
        self.voltaje = voltaje
        self.descenso_activo = False
        self.segundos_restantes = 0
        self.temperatura_exterior_normal = temperatura_exterior

    # Métodos para cálculos
    def calcular_calor(self):
        masa = self.capacidad/1000
        variacion_temperatura = self.temperatura_final -
self.temperatura_inicial
        calor = masa * self.capacidad_calorifica * variacion_temperatura
        return calor

    def calcular_potencia(self):
        if self.resistencia:
            potencia = ((self.voltaje ** 2) / self.resistencia)
        else:
            potencia = self.calcular_calor() / self.tiempo
        return potencia

    def temperatura_por_segundo(self):
        potencia = self.calcular_potencia()
        aumento_temperatura = potencia / ((self.capacidad/1000) *
self.capacidad_calorifica)
        return aumento_temperatura

    def obtener_temperaturas_sin_perdida(self):
```

```
        segundos = []
        temperaturas = []
        for seg in range(self.tiempo + 1):
            temperatura = self.temperatura_inicial +
self.temperatura_por_segundo() * seg
            segundos.append(seg)
            temperaturas.append(temperatura)

        return segundos, temperaturas

    def obtener_temperaturas_con_perdida(self):
        segundos = []
        temperaturas = []
        temperatura_actual = self.temperatura_inicial
        tiempo = 0
        while temperatura_actual <= 80:
            if self.descenso_activo:
                if self.segundos_restantes > 0:
                    self.segundos_restantes -= 1
                else:
                    self.temperatura_exterior =
self.temperatura_exterior_normal
                    self.descenso_activo = False
                tiempo += 1
            else:
                num_aleatorio = random.randint(1,300)
                if num_aleatorio == 1:
                    descenso_temperatura = random.randint(-50, 0)
                    duracion= random.randint(1, 40)
                    self.temperatura_exterior = descenso_temperatura
                    self.segundos_restantes = duracion
                    self.descenso_activo = True
                tiempo += 1

            calor_perdido = (self.conductividad_de_telgopor *
self.superficie * (temperatura_actual - self.temperatura_exterior) /
self.espesor)
            variacion_temperatura = self.temperatura_por_segundo() -
(calor_perdido / self.capacidad_calorifica)
            temperatura_actual += variacion_temperatura
            segundos.append(tiempo)
            temperaturas.append(temperatura_actual)

        return segundos, temperaturas

def main():
    temperatura_inicial = 20
    temperatura_final = 80
    temperatura_exterior = 20
```

```
    '''
    resistencias = np.random.uniform(40, 50, 5)
    temperaturas_iniciales = np.random.normal(10, 5, 5)
    temperaturas_exteriores = np.random.uniform(-20,50,5)
    voltajes = np.random.normal(220, 40, 5)
    '''
    opcion = int(input("1 para temperatura sin pérdida o 2 para
temperatura con pérdida de calor: "))

    proyecto = Proyecto(temperatura_inicial,temperatura_final
,temperatura_exterior)
    if opcion == 1:
        segundos, temperaturas =
proyecto.obtener_temperaturas_sin_perdida()
        plt.plot(segundos, temperaturas)
    elif opcion == 2:
        segundos, temperaturas =
proyecto.obtener_temperaturas_con_perdida()
        plt.plot(segundos, temperaturas)

    # Configuración del gráfico
    plt.xlabel("Tiempo (s)")
    plt.ylabel("Temperatura (°C)")
    plt.title("Gráfico de Temperatura")
    plt.legend()
    plt.show()

if __name__ == "__main__":
    main()
```