

# **ESCUELA POLITÉCNICA NACIONAL**

## **FACULTAD DE INGENIERÍA MECÁNICA**

### **DESARROLLO DE UNA HERRAMIENTA INFORMÁTICA PARA EVALUACIÓN DEL COMPORTAMIENTO TERMODINÁMICO DE SISTEMAS DE REFRIGERACIÓN POR COMPRESIÓN DE VAPOR CON REFRIGERANTES NATURALES PROPANO R-290**

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO  
REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO MECÁNICO**

**KEVIN ISMAEL NARANJO CHISAGUANO**

**kevin.naranjo@epn.edu.ec**

**Ing. NARANJO MENDOZA CARLOS ANDRÉS, PhD.**

**carlos.naranjo@epn.edu.ec**

**Quito, Febrero 2023**

## CERTIFICACIONES

Yo, KEVIN ISMAEL NARANJO CHISAGUANO declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.



**KEVIN ISMAEL NARANJO CHISAGUANO**

**kevin.naranjo@epn.edu.ec**

**naranjoisma2@gmail.com**

Certifico que el presente trabajo de integración curricular fue desarrollado por KEVIN ISMAEL NARANJO CHISAGUANO, bajo mi supervisión.



**Ing. CARLOS ANDRÉS NARANJO MENDOZA, PhD.**


**carlos.naranjo@epn.edu.ec**

## DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.



KEVIN ISMAEL NARANJO CHISAGUANO



Ing. CARLOS ANDRÉS NARANJO MENDOZA, PhD

## **DEDICATORIA**

A Dios, por darme la vida y sabiduría necesaria para llegar hasta aquí, quien permitió que cumpla mis metas mediante el esfuerzo y perseverancia.

A mis padres Ligia y Luis, por darme su apoyo incondicional en todo momento y los valores que han hecho en mí un hombre de bien.

A mi amada Mildred por haberme apoyado día tras día sin esperar nada a cambio y haberme acompañado en esta gran aventura.

A mi pequeño hijo Gael, por ser mi fuente de inspiración para culminar la meta que años atrás me propuse.

## **AGRADECIMIENTO**

A Dios, sin él nada de esto sería posible por su amor y sabiduría infinita.

A mis padres Ligia y Luis, por su gran ejemplo de perseverancia y lucha.

A mi amada Mildred por su valiosa compañía en los momentos más importantes.

A Gael, por regalarme el amor más puro e incentivarme a ser un gran padre.

A mi tutor Carlos Naranjo por su guía y apoyo a lo largo de mi carrera.

# ÍNDICE DE CONTENIDO

CERTIFICACIONES.....	I
DECLARACIÓN DE AUTORÍA.....	II
DEDICATORIA.....	III
AGRADECIMIENTO.....	IV
ÍNDICE DE CONTENIDO.....	V
RESUMEN .....	VII
ABSTRACT .....	VIII
1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO .....	1
1.1 Objetivo general .....	2
1.2 Objetivos específicos .....	2
1.3 Alcance .....	2
1.4 Marco teórico .....	3
Sistema de aire acondicionado tipo Split.....	3
Ciclos de refrigeración por compresión de vapor .....	4
Refrigerante naturales .....	6
Refrigerante propano R-290.....	6
Comparación entre R-134a y R-290.....	7
Programación e interfaz de usuario.....	8
Dispositivo de procesamiento.....	9
2 METODOLOGÍA .....	10
2.1 Recopilación de información teórica.....	12
Variables que intervienen en la generación de diagramas de refrigeración por compresión de vapor .....	12
Modelos matemáticos de ciclos de refrigeración por compresión de vapor.....	13
2.2 Codificación de Software.....	14
Resolución numérica con librería Coolprop.....	15
Resolución gráfica con librería Fluprodia .....	17
Interfaz gráfica de usuario con librería Tkinter .....	19
Intercomunicación con Arduino Mega .....	22
Generación de archivos.....	22
2.3 Verificación y ejecución del aplicativo .....	23
3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES.....	24

3.1	Resultados .....	24
	Registro de datos .....	25
	Comparación de resultados con diferentes aplicativos .....	26
	Errores en resultados de aplicativo Python .....	27
	Documentos exportados.....	27
3.2	Conclusiones.....	29
3.3	Recomendaciones.....	30
4	REFERENCIAS BIBLIOGRÁFICAS .....	31
5	ANEXOS.....	33
	ANEXO I .....	33
	ANEXO II .....	39

## RESUMEN

En el presente trabajo de integración curricular se realiza una herramienta informática con el lenguaje de programación de Python, que permite monitorear el funcionamiento termodinámico de un sistema de refrigeración por compresión de vapor o módulo Split ubicado en el centro de perfeccionamiento en el uso de refrigerantes CEPUR, usando el refrigerante natural propano o R-290 como fluido de trabajo.

Las propiedades y diagramas  $\ln(P)$ - $h$  y  $T$ s de los ciclos de refrigeración real e ideal son proporcionados por la codificación asistida por las librerías CoolProp y Fluprodia. Además para el desarrollo de la herramienta informática se utiliza la librería Tkinter para la generación de widgets y ventanas dinámicas.

La interfaz gráfica está diseñada para procesar 8 datos de entrada, 6 son de temperatura ingresados de manera automática desde un módulo Arduino y 2 son de presión ingresados de manera manual por el usuario. Estos datos son suficientes para crear diagramas superpuestos del ciclo de refrigeración y resultados de calor absorbido y rechazado, trabajo del compresor y COP.

Se comprueba la efectividad del software realizado para el uso de prácticas de laboratorio, al comparar los resultados gráficos y numéricos de calor, trabajo y COP en Python con aplicativos como EES y Visual Studio, dando como error máximo de 1% en el COP de refrigeración ideal y real.

Tanto el banco de pruebas como el software de análisis servirá para capacitar a los estudiantes y técnicos de refrigeración sobre el comportamiento termodinámico del R-290 que es la tendencia actual para los sistemas de refrigeración.

**KEYWORDS:** Propano, R-290, Termodinámica, Refrigeración, Programación, Python.



## ABSTRACT

This degree project shows a computer tool with the Python programming language, which allows monitoring the thermodynamic operation of a vapor compression refrigeration system or Split module located in the refrigerant use improvement center or CEPUR, using the natural refrigerant propane or R-290 as working fluid.

The properties and diagrams  $\ln(P)$ - $h$  and  $T_s$  of the real and ideal refrigeration cycles are provided by the coding assisted by the CoolProp and Fluprodia libraries, in addition to the development of the graphical interface the Tkinter library is used for the generation of widgets and dynamic windows.

The graphical interface is designed to process 8 input data, 6 of them are temperature inputted automatically from an Arduino module and 2 of them are pressure inputted manually by the user. These data are sufficient to create overlay diagrams of the refrigeration cycle and results of heat, work and COP.

The effectiveness of the software made for the use of laboratory practices is verified, by comparing the graphic and numerical results of absorbed and rejected heat, compressor work and COP in Python with applications such as EES and Visual Studio, giving a maximum error of 1% in the refrigeration COP perfect and real.

The test bench and software will serve to train students and refrigeration technicians on the thermodynamic behavior of R-290, which is the current trend for refrigeration systems.

**KEYWORDS:** Propane, R-290, Thermodynamics, Refrigeration , Programming, Python.

# 1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

Con el fin de monitorear el funcionamiento del módulo Split ubicado en el centro de perfeccionamiento en el uso de refrigerantes CEPUR, se realizó la herramienta informática correspondiente. El diseño y desarrollo del aplicativo se programó en software libre con lenguaje de programación en Python para el análisis del funcionamiento termodinámico en tiempo real de sistemas de enfriamiento que operan bajo ciclos de refrigeración por compresión de vapor con R-290.

El aplicativo desarrollado permite calcular diagramas termodinámicos interactivos de Presión – Entalpía y Temperatura – Entropía con resultados numéricos de calores y coeficientes de desempeño que muestran el comportamiento real e ideal del R-290 en un ciclo de refrigeración simple. Las variables de entrada son 6 datos de temperatura que se obtiene del Arduino de manera automática y 2 datos de presiones que se debe ingresar de manera manual al Frame de la interfaz, estos 8 datos son suficientes para obtener las respuestas numéricas y gráficas. El software permite además exportar los diagramas de refrigeración real e ideal superpuestos y las propiedades termodinámicas con extensión png y txt respectivamente para futuros estudios y análisis independientes del proyecto.

Para llevar a cabo la interfaz gráfica se realizó el estado del arte del comportamiento del R-290 en otros sistemas de refrigeración, así como la sintaxis de codificación en Python para obtener los resultados numéricos y gráficos. El pilar fundamental de la programación se basa en las librerías que permitió un fácil desarrollo del software, estos son Coolprop, Fluprodia y Tkinter. La función principal de Coolprop y Fluprodia es determinar todas las propiedades termodinámicas que se desee en el diagrama de Mollier del refrigerante R-290, partiendo de 2 propiedades intensivas independientes, de tal forma se obtiene una base de datos de propiedades que se usan para cálculos de calor absorbido y rechazado, trabajo del compresor y COP. Por otra parte, Tkinter permite diseñar la interfaz gráfica amigable para el usuario por medio de widgets que permite la intercomunicación constante entre funciones, cálculos y gráficos.

Para comprobar la factibilidad y confianza de los resultados obtenidos por el software diseñado en Python, se comparan sus resultados numéricos y gráficos con otros aplicativos de programación como EES que dispone su propia base de datos del R-290 y un script de Visual Studio cuyas propiedades se obtuvieron tras la interpolación de las tablas de saturación del R-290. El aplicativo dispondrán de un manual de usuario que permitirá el uso correcto de la interfaz así como el método de descarga paso a paso con extensión exe.

## **1.1 Objetivo general**

Realizar una herramienta informática para evaluar el comportamiento termodinámico de un sistema de refrigeración por compresión de vapor con refrigerante R-290.

## **1.2 Objetivos específicos**

1. Recopilar información acerca del comportamiento del R-290 como refrigerante en ciclos de refrigeración y sintaxis del lenguaje de programación de Python.
2. Determinar los puntos que proporciona el registro de datos experimentales del módulo Split y su procesamiento de información.
3. Codificar una interfaz gráfica para obtener los cálculos y gráficos que describan el funcionamiento del R-290 en el ciclo de refrigeración.
4. Genera pruebas de funcionamiento entre el módulo Split y aplicativo y comparar sus resultados con otros software de programación.

## **1.3 Alcance**

Se realizará el código informático que permita comprender el funcionamiento termodinámico de un banco de pruebas de refrigeración por compresión de vapor con propano. Para lo cual se realizará la lectura y registro de datos térmicos y de presión de cada componente en formato .txt para su posterior procesamiento en el software libre Python, asistido con la librería de CoolProp y Fluprodia. Para una buena interpretación de datos, se desarrollará una interfaz agradable para el usuario en donde se podrá visualizar los diagramas T-S y P-H junto los datos registrados en cada componente, la toma de datos es secuencial y la interfaz se actualizará automáticamente. Además, la interfaz podrá comparar gráficamente el comportamiento real e ideal del refrigerante R-290, mostrando ambos diagramas superpuestos.

Los puntos para la lectura de datos provienen del evaporador, válvula de expansión, compresor y unidad condensadora. Dichos puntos proporcionarán los datos suficientes para la generación de procesos matemáticos y el desarrollo de los diagramas termodinámicos, cálculos del COP, rendimientos, cargas térmicas, potencia e interpretación de resultados. Las interfaces se dividen en subinterfaces donde indican los diagramas, cálculos y resultados por separado, además podrán ser monitoreadas desde un computador externo asistido por un cable de datos USB.

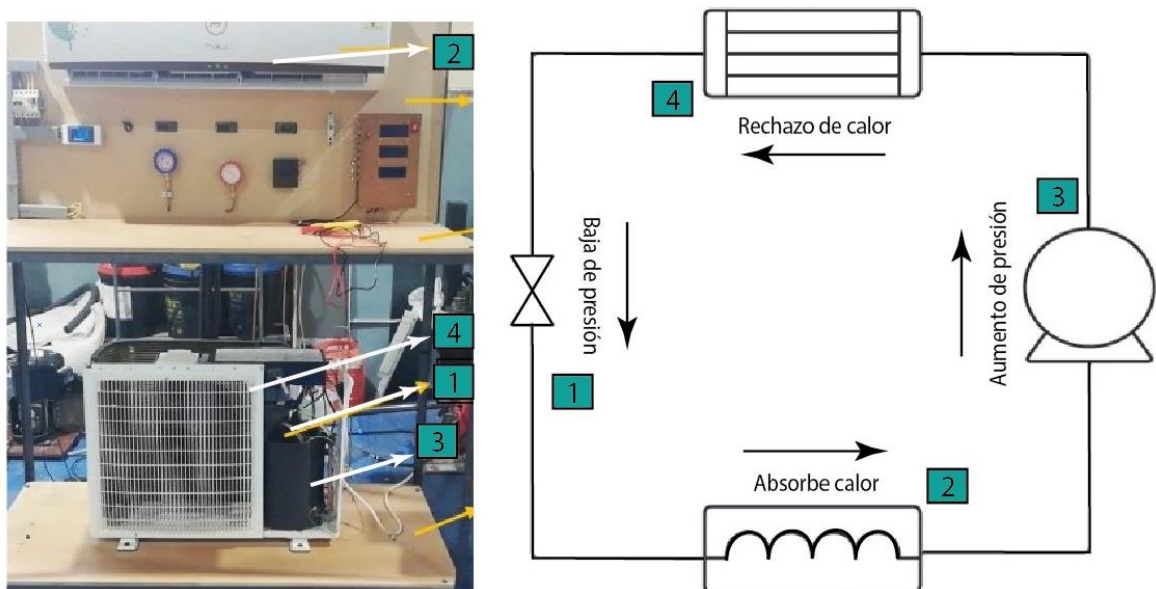
## 1.4 Marco teórico

El comportamiento y análisis termodinámico del R-290 dentro de un sistema de refrigeración por compresión de vapor conlleva varios pasos que se deben realizar de manera minuciosa para obtener gráficos y resultados confiables. Por ende, esta sección contiene varios conceptos importantes para la ejecución de un adecuado código de programación.

### Sistema de aire acondicionado tipo Split

El ciclo de refrigeración consiste en que un fluido de trabajo con bajo punto de ebullición y químicamente estable pueda absorber y rechazar calor cambiando sus estados, de fase líquida a gaseosa y viceversa, de tal manera pueda refrigerar un espacio determinado [1]. A diferencia de un sistema de aire acondicionado que posee todos sus componentes compactados, un módulo Split se conforma por dos dispositivos principales por separado, la unidad evaporadora y condensadora ubicada al interior y exterior del espacio a refrigerar respectivamente [2].

Los principales elementos del Split están interconectados por tubos de cobre junto a los componentes auxiliares, además un fluido de trabajo considerado como refrigerante ingresa por cada uno de ellos, describiendo así un ciclo de refrigeración simple por compresión de vapor tal como se muestra en la figura 1.1.



**Figura 1.1.** Componentes del ciclo de refrigeración en el módulo Split [3].

La figura 1.1 indica la trayectoria que cumple el refrigerante en todo el ciclo, ingresando por los principales componentes del sistema Split, los cuales son:

1. Válvula de estrangulamiento o de expansión
2. Unidad evaporadora
3. Compresor
4. Unidad condensadora

Se puede definir al ciclo de refrigeración como ideal para comprender con mayor facilidad el comportamiento termodinámico del fluido y las ecuaciones físicas que lo gobiernan al pasar por cada componente [3].

El fluido se comprime en un proceso isentrópico en el compresor para ingresar al condensador y rechazar calor isotérmicamente, de tal forma que su estado cambia de vapor a líquido saturado a la presión de alta. Secuencialmente, el fluido ingresa a la válvula de expansión para expandirse con entalpía constante y terminar como mezcla antes de ingresar al evaporador, que es donde absorbe calor isotérmicamente. Finalmente, al salir del evaporador su estado cambia a vapor saturado a la presión de baja, mismo que ingresa de nuevo al compresor para reiniciar el ciclo.

Tanto como las tuberías de cobre y dispositivos auxiliares que interconectan a los elementos primarios, deben estar aislados y protegidos contra factores externos para evitar un mal funcionamiento del módulo [4]. Estos factores pueden provocar pérdidas de temperatura en los tubos por falta de aislantes térmicos, alteración de lectura de datos en sensores por impulsos magnéticos provocada por el compresor, daño a equipos eléctricos a causa de sobretensiones, bajo rendimiento por fugas en el sistema, entre otras.

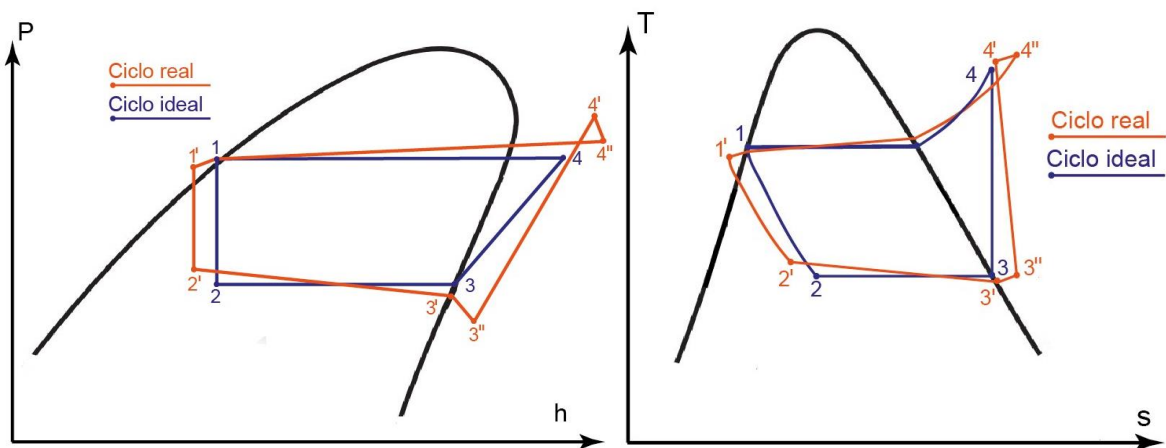
### **Ciclos de refrigeración por compresión de vapor**

Las irreversibilidades en los ciclos de refrigeración son notables por dos causas principales, la caída de presión a causa de la fricción que existe entre los componentes y el fluido de trabajo y la transferencia de calor entre los componentes y ambiente con fluido de trabajo [4]. El ciclo ideal de refrigeración omite estas irreversibilidades para su facilidad de comprensión comportándose como un sistema adiabático y si perdidas de presión, mientras que el ciclo real considera que hay una transferencia de calor permanente en todos tubos de conexión así como cambios de presión a través de los dispositivos.

Los puntos que definen a los ciclos de refrigeración se consideran como estados del fluido de trabajo y se fijan por medio de dos propiedades termodinámicas [5]. En la práctica para

monitorear un sistema de refrigeración, si se desea conocer las propiedades como entalpia y entropía de un estado fuera de la campana termodinámica, se necesita saber la temperatura y presión en dicho punto, mientras si está dentro de la campana termodinámica se necesita conocer la calidad y temperatura o presión de saturación.

La comparación entre el ciclo real e ideal del sistema de refrigeración y los ciclos termodinámicos de Presión – Entalpia “Ph” y Temperatura Entropía “Ts”, permiten localizar específicamente las irreversibilidades y comprender las condiciones de operación del fluido de trabajo. En la figura 1.2 se puede visualizar las irreversibilidades y comparar los ciclos reales e ideales



**Figura 1.2.** Comparación de diagramas Ph (izquierdo) y Ts (derecho) [6].

De la figura 1.2 se pueden resaltar las principales características dentro del ciclo real:

- a. 1-1': Subenfriamiento y caída de presión entre condensador y línea de entrada a la válvula de expansión.
- b. 1'-2': Proceso de estrangulación real – no es completamente isoentálpico.
- c. 2'-3': Proceso evaporativo con caída de presión en toda la línea del evaporador.
- d. 3'-3'': Sobrecalentamiento y caída de presión entre evaporador y compresor.
- e. 3''-4': Proceso de compresión real – no es completamente isentrópico.
- f. 4'-4'': Caída de presión obligatoria para forzar la descarga por las válvulas.
- g. 4''-1: Proceso de condensación con caída de presión en toda la línea del condensador.

La eficiencia del ciclo real e ideal están determinadas por el coeficiente de desempeño, este valor es la razón entre el calor que absorbe el evaporador y el trabajo que desempeña el compresor, por tanto obedeciendo a la primera ley de la termodinámica se obtiene la ecuación 1.1 y 1.2.

$$COP_{real} = \frac{q_{abs}}{W_{comp}} = \frac{h_{C'} - h_{B'}}{h_{D''} - h_{C''}}$$

**Ecuación 1.1.** COP real

$$COP_{ideal} = \frac{q_{abs}}{W_{comp}} = \frac{h_C - h_B}{h_D - h_C}$$

**Ecuación 1.2.** COP ideal

Dentro de los sistemas de refrigeración, es muy común observar en los diagramas que el refrigerante se sobrecalienta antes del compresor y se subenfía después del condensador. Esto se debe a que ningún equipo es adiabático, también que el compresor no es isentrópico y existe pérdidas de presión en los conductos del refrigerante. Por lo tanto en la mayoría de los equipos de refrigeración, la tasa de calor absorbido en el evaporador aumenta en el ciclo real, resultando que el COP real sea mayor que el COP ideal [7].

### Refrigerantes naturales

Los refrigerantes naturales son aquellos que no provocan una influencia negativa al medio ambiente y brindan un buen rendimiento termodinámico en máquinas refrigeradoras, por lo tanto la transición hacia la fabricación y uso de equipos con estos refrigerantes tiene un mayor impacto en la actualidad [8]. Además de poseer excelentes propiedades termodinámicas, los refrigerantes HC tienen cero potencial de destrucción de la capa de ozono (ODP) y potencial de calentamiento global (GWP) cercano de cero [9].

**Tabla 1.1.** Comparación de diagramas Ph (izquierdo) y Ts (derecho) [8].

Refrigerante	Categoría	ODP	Años de vida en atmósfera	GWP 1000 años CO <sub>2</sub> =1
R-11	CFC	1	45	4700
R-12	CFC	1	100	10700
R-113	CFC	0.8	85	6000
R-22	HCFC	0.05	12	1800
R-123	HCFC	0.02	1.3	76
R-32	HFC	0	4.9	670
R-134a	HFC	0	14	1400
R-290	HC	0	3	3
R-600	HC	0	-	4
R-717	NH <sub>3</sub>	0	1	0
R-744	CO <sub>2</sub>	0	100	1
R-1234yf	HFO	0	11 days	4
R-1234ze	HFO	0	-	6

HFC: hidrofluorocarburos CFC: clorofluorocarburos HCFC: hidroclorofluorocarburos  
HFC: hidrofluorocarburos HC: hidrocarburos HFO: hidrofluoroolefinas

Según la tabla 1.1 los refrigerantes HFC como R-134a poseen un valor de GWP muy alto en comparación a los refrigerantes HC como el R-290, además existe una comparativa muy notable de 1400 GWP a 3 GWP respectivamente.

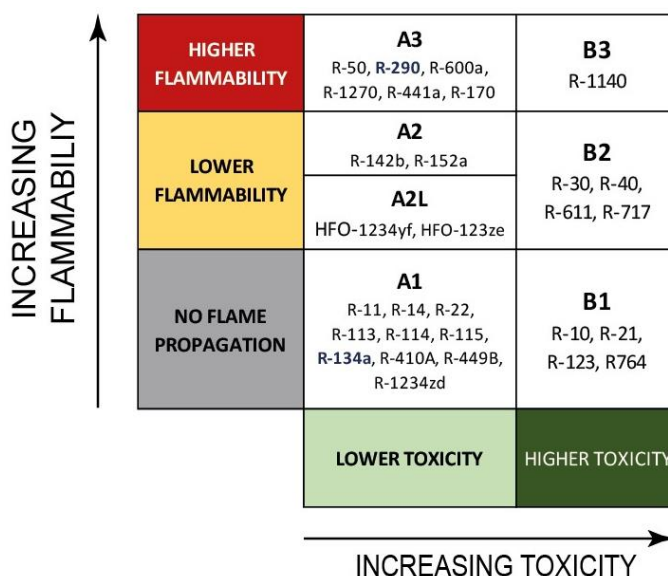
Por otra parte, los refrigerantes naturales tienen un tiempo de vida en la atmosfera muy baja, generando una expectativa agradable para desarrollar equipos adecuados para el R-290, R-600a y amoníaco.

La normativa de regulación europea de gas [9] ha restringido la comercialización de refrigerantes HCF (hidrofluorocarburos) como R134a o R410a en la mayoría de los países de Europa y dan apertura a los refrigerantes naturales HC (hidrocarburos) como el propano (R-290) e Isobutano (R-600a).

### Refrigerante propano R-290

El propano se obtiene por el refinado del petróleo dentro del proceso de fraccionamiento del gas natural, por lo tanto se considera como un gas inflamable. Su composición química es  $CH_3CH_2CH_3$  y a causa de su estructura molecular y bajo punto de ebullición tiene varias aplicaciones como refrigerante o carburante para refrigeración, bombas de calor, aire acondicionado, entre otros [10].

La ASHRAE [11] califica al propano como un refrigerante tipo A3, considerado como un refrigerante amigable con el medio ambiente y con bajos niveles de toxicidad, además cuenta con características térmicas superiores a las del R-134a y R-404a. La desventaja de los HC es que son categorizados como inflamables y es indispensable considerar la inflamabilidad para el diseño y mantenimiento de equipos de refrigeración con propano [9].



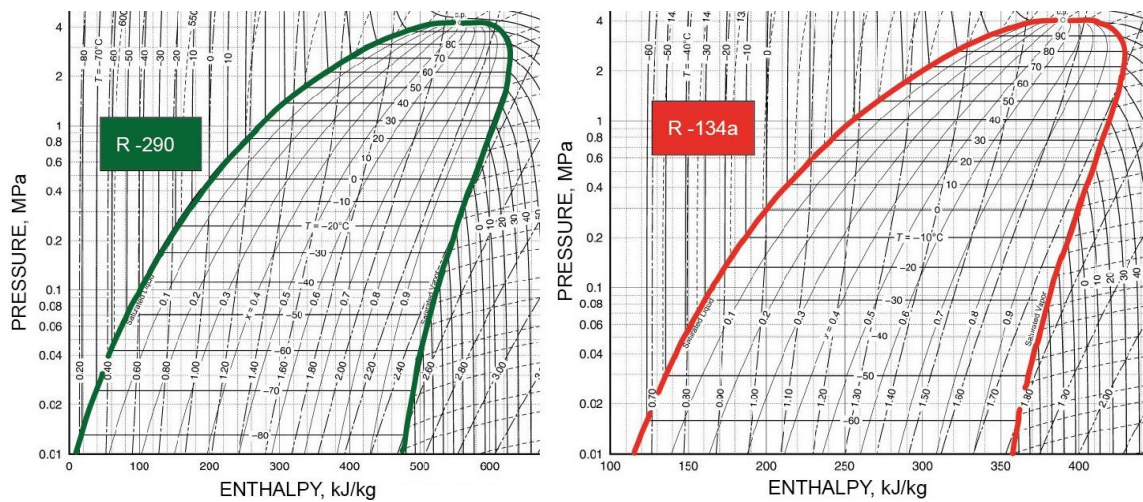
**Figura 1.3.** Clasificación de la seguridad de refrigerantes [11].



La figura 1.3 indica que los refrigerantes de hidrocarburos, en especial el R-290, presentan propiedades amigables con el medio ambiente, las cuales pueden ser aprovechadas en los sistemas de refrigeración, teniendo un desempeño equiparable a los refrigerantes fluorocarbonados como el R-134a.

### Comparación entre R-134a y R-290

El R-290 posee una campana más ancha en el cambio de fase en comparación al R-134a, lo que indica que el calor latente en el cambio de fase del sistema es mayor para el propano, esto se puede visualizar en la figura 1.4, donde los diagramas de R-290 y R-134a corresponde al de la izquierda y derecha respectivamente.



**Figura 1.4.** Campanas termodinámicas Ph del R-290 y R-134a [11].

Al considerar un mismo flujo másico para ambos refrigerantes en un sistema de refrigeración, la capacidad de absorber calor a lo largo del evaporador y rechazarlo en el condensador es mayor en el R-290 que el R-134a [6]. Sin embargo se requiere mayor energía para comprimir 1 kg de propano que el R134a, provocando la modificación en el compresor para el uso correcto. Ciertas características termodinámicas se comparan en la tabla 1.2.

**Tabla 1.2.** Presiones de condensación y evaporación para el R-134a y R-290 [6].

Refrigerante	Presión alta kPa ( $T_{sat}=40^{\circ}\text{C}$ )	Presión baja kPa ( $T_{sat}=-15^{\circ}\text{C}$ )	Variación de presiones kPa	Porcentaje %
R - 134a	1018	170	853	100%
R - 290	1366	289	1077	126%

La tabla 1.2 indica que el propano tiene una presión de evaporación y condensación más alta que el R-134a. Si se considera un mismo compresor que trabaja con ambos fluidos de trabajo, la temperatura de evaporación será más alta y la de condensación será más baja en el R-290. Los principales elementos que se cambia para la adaptación de ambos refrigerantes en un mismo sistema de refrigeración es la válvula de expansión y el compresor [12].

## Programación e interfaz de usuario

Se consideran herramientas computacionales a los lenguajes de programación que permiten desarrollar algoritmos para resolver desde básicos problemas numéricos hasta complejos procesos o fenómenos [13]. Los fabricantes de equipos como los sistemas de refrigeración usan múltiples plataformas para el control de los componentes, por medio de interfaces gráficas programadas en lenguajes como C, C++, Matlab, Fortran, R, Python y Visual Estudio.

Una GUI se considera como una interfaz donde el usuario logra comunicarse con el ordenador por medio de un lenguaje de programación y un diseño visual conformado por frames, botones, bloques, campos de entrada y ventanas [14]. Estas características deben permitir que la GUI sea satisfactoria, entendible para el usuario, efectiva y eficiente al desarrollar las tareas programadas.

Uno de los lenguajes de programación más fáciles y potentes que permitan no solo crear interfaces de usuario, sino también usar librerías que facilitan al desarrollo de programas para analizar los sistemas de refrigeración es Python. Además posee un amigable sistema de programación orientada a objetos con librerías orientadas al estudio de sistemas termodinámicos como Tkinter, Coolprop y Fluprodia [14].

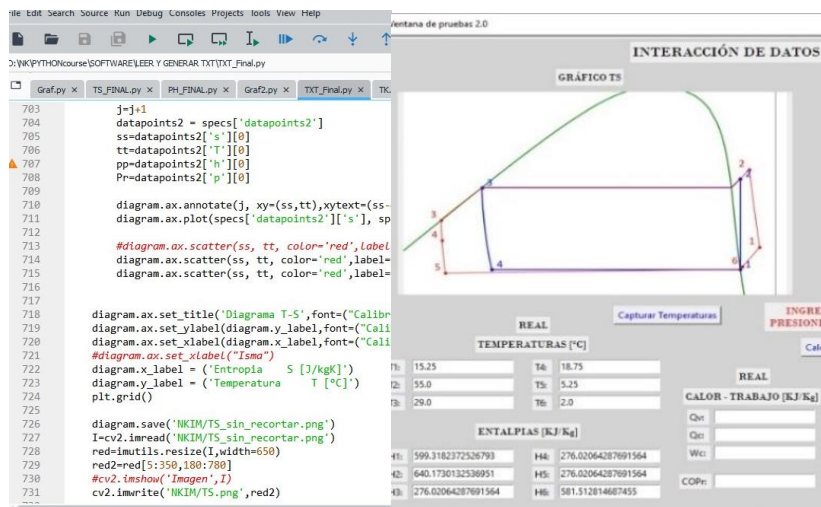


Figura 1.5. Interfaz desarrollada en Python

## Dispositivo de procesamiento

Los dispositivos por excelencia de procesamiento de datos se consideran como microcontroladores, ya que disponen de un sistema complejo de recopilación de datos programable, los más usados en el mercado son Intel, Microchip y Arduino [15]. Para el análisis a nivel de ingeniería, los microcontroladores Arduino son los más usados, en especial para el monitoreo de bancos de prueba como los equipos de aire acondicionado.

Arduino se conforma por un software y hardware que permite la programación libre del usuario a través de un ordenador. El Software se conforma por una interfaz llamada Arduino IDE que permite comunicarse con la tarjeta por medio de un lenguaje de programación Processing/Wiring, mientras el Hardware es una placa con pines y puertos periféricos de entrada y salida de datos gobernados por un microcontrolador.

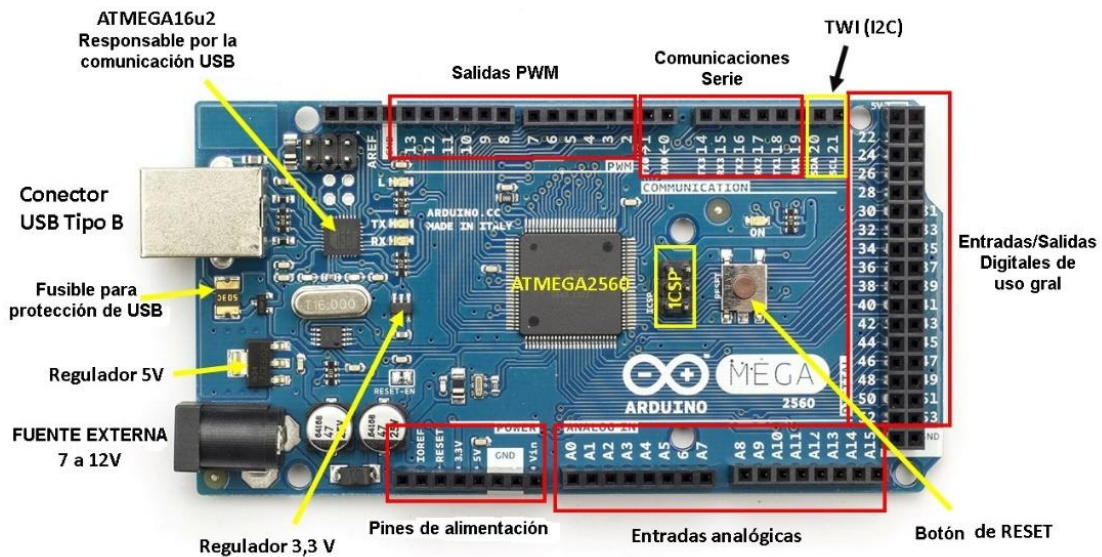


Figura 1.6. Arduino Mega [15].

El Arduino Mega de la figura 1.6 se considera como el microcontrolador más potente de la familia de los Arduino, ya que presenta las siguientes características:

1. 54 pines digitales y 31 analógicos con alta frecuencia
2. 4 puertos de comunicación serial
3. Memoria FLASH con 256 Kb y 8 Kb de memoria SRAM

## 2 METODOLOGÍA

El presente proyecto de integración curricular se lleva a cabo bajo la metodología de una investigación con línea proyectiva. Según [16], este tipo de investigación se plantea para proponer soluciones de alto potencial que no se habían aprovechado para un problema diagnosticado. En este caso, la problemática se basa en la falta de plataformas educativas que permitan comprender las condiciones de operación del R-290 en el módulo Split ubicado en el CEPUR. De esta forma se propone la creación de una interfaz gráfica que indica los ciclos reales e ideales de refrigeración de los principales diagramas termodinámicos, así como propiedades y resultados de operación en tiempo real del sistema Split.

Para llevar a cabo la interfaz gráfica, se propone tres etapas a lo largo de la realización del proyecto:

- Recopilación de información teórica
- Codificación del software
- Verificación y ejecución del aplicativo.

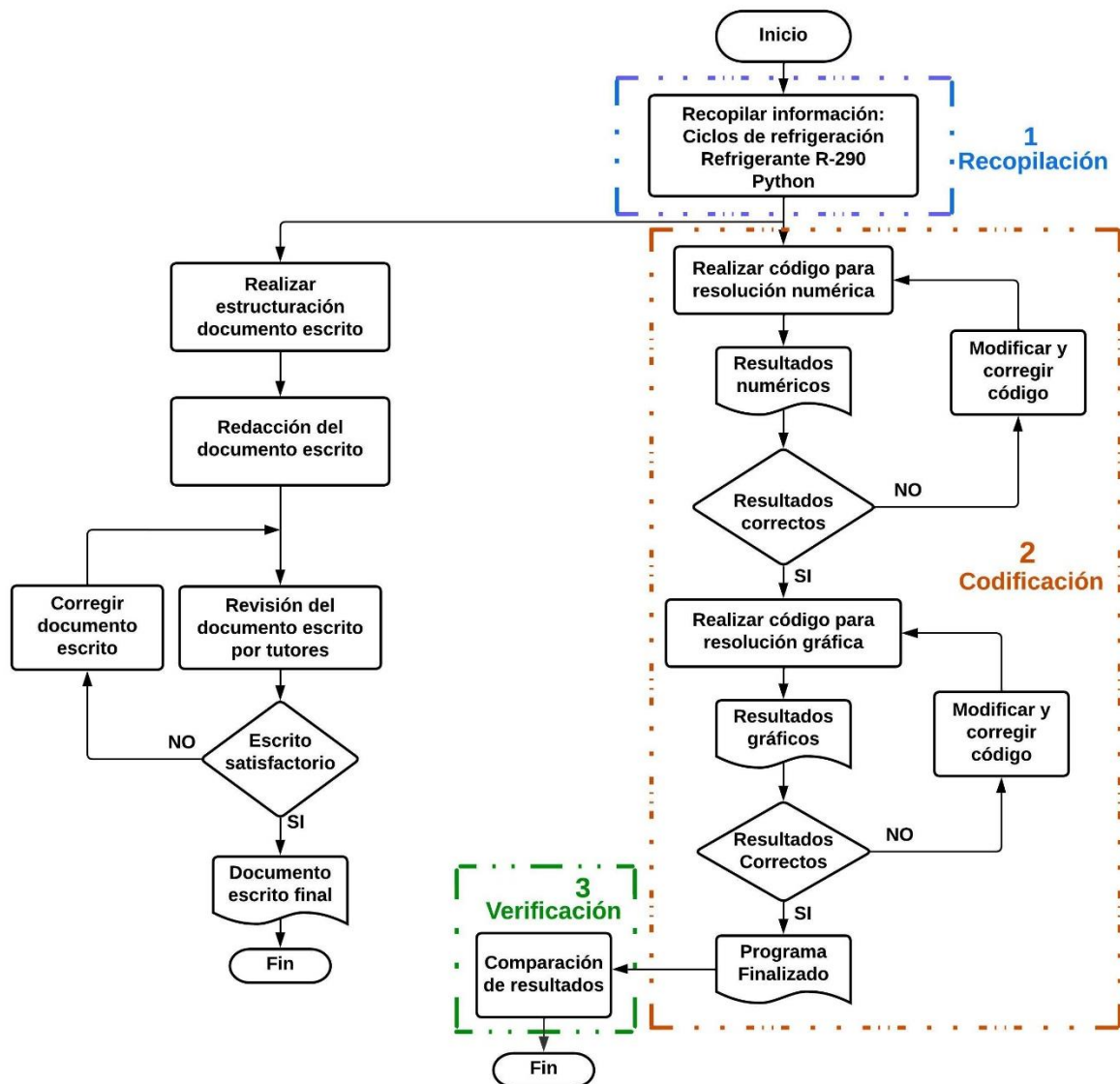
Estas etapas están divididas por otras actividades que se indican en el diagrama de flujo de la figura 2.1.

La Recopilación de información teórica consiste en extraer los fundamentos teóricos y ecuaciones físicas que describan el comportamiento termodinámico de los ciclos de un sistema de refrigeración real e ideal. Además, se debe considerar las propiedades y comportamiento del R-290 dentro de un sistema de refrigeración simple. Adicional, se debe recopilar información acerca de la sintaxis del lenguaje de programación de Python, así como las librerías que permitirá realizar una interfaz de usuario adecuada.

La Codificación del software consiste en generar los múltiples scripts que permitirá obtener respuestas numéricas al aplicar las librerías y ecuaciones con el lenguaje de programación de Python. Posterior se realiza el desarrollo del código fuente la cual permitirá obtener los valores numéricos en forma de diagramas y gráficos, se entiende por diagramas a los ciclos de refrigeración Real e Ideal. Además, todo el código está incluido en la interfaz principal en donde se indica los principales resultados numéricos y diagramas.

La Verificación del aplicativo es la manipulación del código en un archivo ejecutable portable y comparar sus resultados numéricos y gráficos con otros programas especializados como EES y Visual Studio para comprobar su confiabilidad y efectividad.

Además, la ejecución permitirá al usuario conectarse al sistema Split de manera segura, por medio de un manual de usuario incluido en el anexo I. El manual de usuario contiene información del adecuado funcionamiento del aplicativo y su interpretación de los gráficos y documentos que ofrece el software al iniciar la práctica.



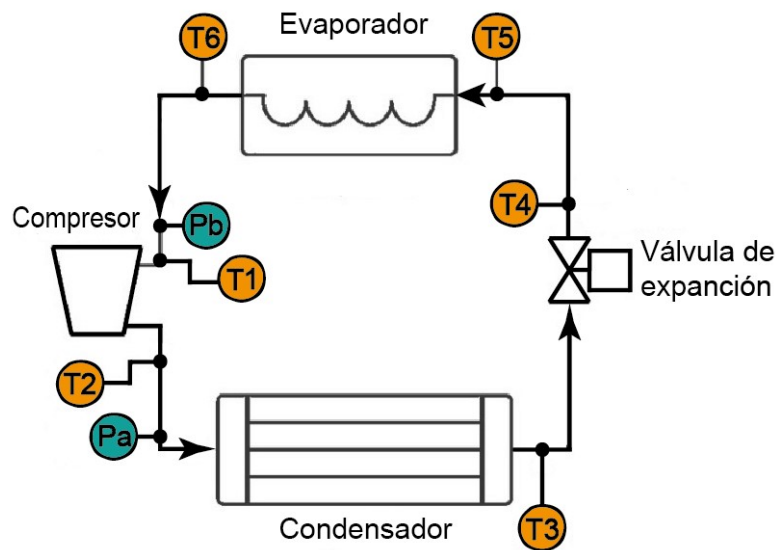
**Figura 2.1.** Metodología aplicada a TIC.

En la figura 2.1 se puede visualizar que el escrito de la documentación se trabaja en paralelo al desarrollo de la herramienta informática, de tal forma que el proyecto de integración curricular tendrá un 2 productos finales, el documento escrito y aplicativo para el sistema Split.

## 2.1 Recopilación de información teórica

### Variables que intervienen en la generación de diagramas de refrigeración por compresión de vapor

El sistema Split ubicado en el CEPUR nos ofrece varios valores que son indispensables para el diseño de la herramienta informática, se obtienen a través de conexiones de termopares a un módulo Arduino para las temperaturas y transductores para las presiones. Cada sensor está ubicado estratégicamente en el Split para obtener las variables adecuadas, tal como se indica en la figura 2.2.



**Figura 2.2.** Principales Transductores y sensores en Split.

Se requieren 6 temperaturas y 2 presiones para calcular 6 puntos con sus propiedades termodinámicas y trazar los diagramas Ts y Ph.

**Tabla 2.1.** Ubicación de sensores y transductores

Punto	Descripción	Ubicación
$P_b$	Presión de baja	Entrada al compresor
$P_a$	Presión de alta	Salida del compresor
$T_1$	Temperatura	Entrada al compresor
$T_2$	Temperatura	Salida del compresor
$T_3$	Temperatura	Salida del condensador
$T_4$	Temperatura	Salida de la válvula de expansión
$T_5$	Temperatura	Entrada al evaporador
$T_6$	Temperatura	Salida del evaporador

Para comprender de mejor manera las próximas ecuaciones y las variables en los códigos de programación, se presenta la siguiente nomenclatura.

**Tabla 2.2.** Nomenclatura de propiedades termodinámicas

<b>Punto</b>	<b>Propiedad</b>	<b>Descripción</b>
$h_1$	Entalpía 1 real	Entrada al compresor
$h_2$	Entalpía 2 real	Salida del compresor
$h_3$	Entalpía 3 real	Salida del condensador
$h_4$	Entalpía 4 real	Salida de la válvula de expansión
$h_5$	Entalpía 5 real	Entrada al evaporador
$h_6$	Entalpía 6 real	Salida del evaporador
$h_{1i}$	Entalpía 1 ideal	Presión de baja
$h_{2i}$	Entalpía 2 ideal	Presión de alta
$h_{3i}$	Entalpía 3 ideal	Presión de alta
$h_{4i}$	Entalpía 4 ideal	Presión de baja

### Modelos matemáticos usados en ciclos de refrigeración

Se aplica la primera ley de la termodinámica a los 4 componentes del ciclo de refrigeración, considerando que son dispositivos de flujo estacionario y despreciando la variación de energía potencial y cinética se obtiene la ecuación 2.1.

$$q_{entrada} - q_{salida} = w_{salida} - w_{entrada}$$

**Ecuación 2.1.** Balance de energías

De la ecuación 2.1, calor de entrada es el calor absorbido por la unidad evaporadora, calor de salida es el calor rechazo en el condensador y trabajo de entrada es la potencia del compresor para circular el refrigerante. Considerando que no hay trabajo de salida y reajustando la ecuación 2.1 al sistema Split, se tiene:

$$q_{abs} + w_{comp} = q_{rech}$$

**Ecuación 2.2.** Balance de energía del sistema Split

En términos de entalpía en el ciclo real se obtiene la ecuación 2.3.

$$(h_6 - h_5) + (h_2 - h_1) = (h_3 - h_2)$$

**Ecuación 2.3.** Balance de energía en función de entalpías reales

En términos de entalpía en el ciclo ideal se obtiene la ecuación 2.4.

$$(h_{1i} - h_{4i}) + (h_{2i} - h_{1i}) = (h_{3i} - h_{2i})$$

**Ecuación 2.4.** Balance de energía en función de entalpías ideales

El coeficiente de desempeño del ciclo real de refrigeración se define en la ecuación 2.5.

$$COP_{real} = \frac{q_{abs}}{w_{comp}} = \frac{h_6 - h_5}{h_2 - h_1}$$

**Ecuación 2.5.** COP real

El coeficiente de desempeño del ciclo ideal de refrigeración se define en la ecuación 2.6.

$$COP_{ideal} = \frac{q_{abs}}{w_{comp}} = \frac{h_{1i} - h_{4i}}{h_{2i} - h_{1i}}$$

**Ecuación 2.6.** COP ideal

## 2.2 Codificación de Software

El sistema Split dispone de 6 termocuplas que ofrecen 6 datos de temperatura conectadas al módulo Arduino y 2 datos de presión que provienen de transductores Full Gauge conectados a un display para su visualización, todos ellos realizan mediciones en tiempo real. Por lo tanto, se dispone de 6 estados, conformada por 6 variables de temperatura y 2 variables de presión.

La interfaz gráfica procesa los datos de temperatura y presión, mismos que permiten realizar las operaciones numéricas y determinar los valores de las propiedades termodinámicas en los 6 estados, calor absorbido, rechazado y COP del sistema. Por otra parte, la interfaz realiza la solución gráfica que se compone por los diagramas de Presión – Entalpía (Ph) y Temperatura – Entropía (Ts) de manera iterativa, eso significa que el usuario puede leer los datos y ejecutar las acciones del aplicativo las veces que desee.

El aplicativo permite generar el último gráfico de la iteración y exportarlo en formato PNG, tanto para el ciclo real e ideal. También permite generar una lista con todas las propiedades termodinámicas que se obtiene en cada iteración en formato TXT.

En resumen, para diseñar y codificar el software se plantea un diagrama de flujo en la figura 2.3 donde indica la metodología del funcionamiento y las acciones que lleva a cabo la interfaz gráfica durante su ejecución. La codificación se realiza con lenguaje de programación de Python por las librerías que ofrece y permite realizar una codificación más sencilla.



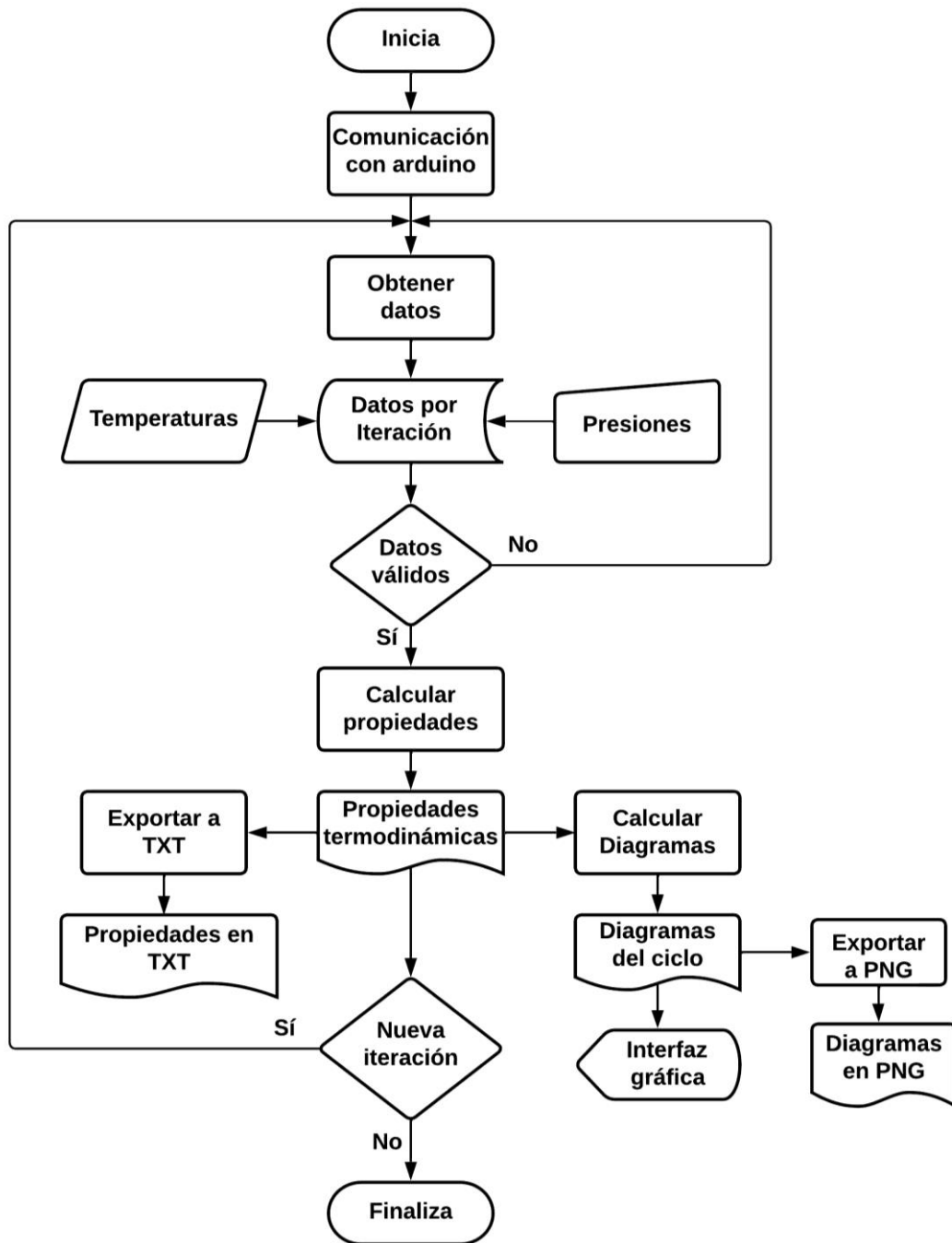


Figura 2.3. Diagrama de flujo de funcionalidad de aplicativo

### Resolución numérica con librería Coolprop

La librería CoolProp permite realizar el computo numérico para determinar todas las propiedades termodinámicas de cualquier refrigerante en un estado específico [10]. La ventaja de utilizar CoolProp es que dispone de una gran lista de fluidos, siendo el R-290 el fluido de trabajo para el sistema de refrigeración.

Para el uso correcto de la librería se tiene que tomar en cuenta 3 aspectos importantes:

- Establecer el nombre correcto del fluido de trabajo
- Identificar la propiedad con las unidades que se desea
- Ingresar 2 propiedades independientes del estado que se busca

Considerando el punto anterior, se ejecuta la línea de código con la estructura que se indica en la figura 2.4.

```
from CoolProp.CoolProp import PropsSI           #Librería
h1r=PropsSI('H','T',300,'P',6000000,'R290');    #Entalpía
```

**Figura 2.4.** Línea de código usando CoolProp

La propiedad termodinámica que se ocupa en la gran mayoría de los cálculos numéricos es la entalpía, por tanto se debe ingresar 'h' como la propiedad que se desea, seguido por las propiedades intensivas, 'T' con un valor de 300 K y 'P' con un valor de 6000000 Pa. La respuesta de entalpía depende del fluido de trabajo, por tanto para el 'R290' es 617929.4 J/kg.

Dentro de la codificación con Coolprop se define las propiedades con sus respectivas unidades por defecto en la tabla 2.3.

**Tabla 2.3.** Propiedades termodinámicas de CoolProp

Símbolo	Propiedad	Unidad
P	Presión	Pa
T	Temperatura	K
h	Entalpía	J/kg
s	Entropía	J/kg.K
Q	Calidad	-

El sistema de unidades de los datos de entrada y salida se definen en la tabla 2.1, sin embargo si se desea trabajar con otras unidades, se debe realizar el cálculo respectivo para la transformación de unidades posterior al cálculo numérico obtenido en CoolProp.

Este proceso de cambio de unidades y determinación de propiedades se realiza las veces que sea necesario hasta obtener 4 datos de salida para el ciclo ideal y 6 datos de salida ciclo real, siendo entalpías y entropías las propiedades de salida más importantes.

```

from CoolProp.CoolProp import PropsSI
Patm=72
Palt=float(P1n.get())*6.89476+Patm
Pbaj=float(P2n.get()) *6.89476+Patm
h1r=PropsSI('H','P',Pbaj*1000,'T',T1r,'R290');
h2r=PropsSI('H','P',Palt*1000,'T',T2r,'R290');
h3r=PropsSI('H','P',Palt*1000,'T',T3r,'R290');
h4r=h3r;h5r=h4r
p3r=PropsSI('P','T',T3r,'Q',0,'R290');
p4r=PropsSI('P','T',T4r,'Q',0.5,'R290');
p5r=PropsSI('P','T',T5r,'Q',0.5,'R290');
T6rp=PropsSI('T','P',Pbaj*1000,'Q',0.5,'R290');
if T6rp>=T6r:
    T6r=T6rp+0.5
h6r=PropsSI('H','P',Pbaj*1000,'T',T6r,'R290');

```

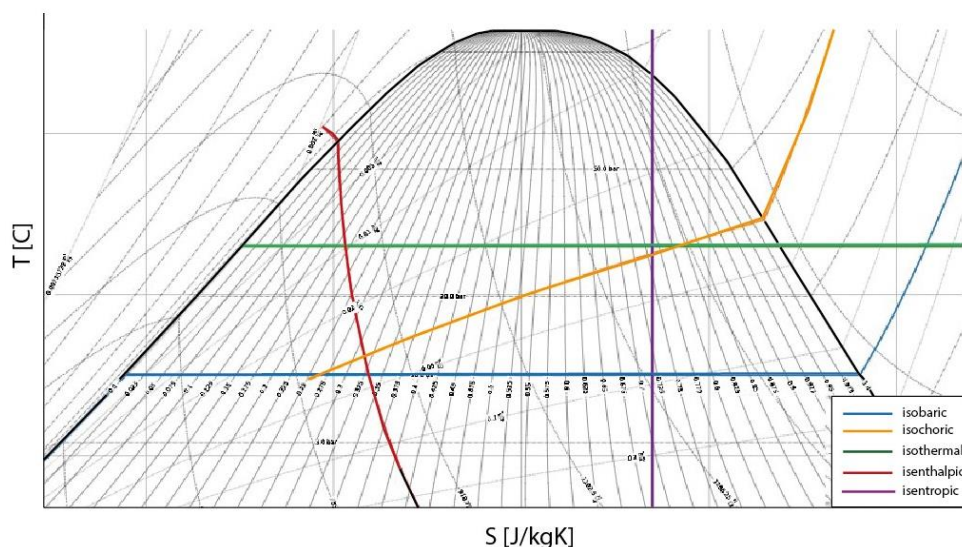
**#Librerías**  
**#Presión atmmosferica**  
**#Presión de alta de psig a kPa**  
**#Presión de baja de psig a kPa**  
**#Entalpías**  
  
**#Condición de ciclo real**  
**#Presiones auxiliares**  
  
**#Temperaturas auxiliares**  
**#Condiciones por error**

**Figura 2.5.** Codificación numérica usando CoolProp

En la figura 2.5 se muestra una sección de la codificación para obtener las entalpías y otras propiedades de cada ciclo, además se utiliza la sintaxis de Python para obtener los valores de calor y COP utilizando las ecuaciones del apartado de recopilación de información. Las presiones que son digitadas manualmente por el usuario en psig  $Palt$  y  $Pbaj$ , se transforma automáticamente a kPa para su uso correcto en CoolProp.

### Resolución gráfica con librería Fluprodia

La librería Fluprodia permite realizar el computo gráfico para obtener los diagramas Ph y Ts de los fluidos que pertenecen a CoolProp, en este caso del R-290. Fluprodia es la herramienta que permite dibujar isóneas independientes de cualquier propiedad termodinámica, describiendo un estado inicial y final, de esta forma se pueden crear isóneas ideales y reales [14].



**Figura 2.6.** Isóneas en campana termodinámica usando Fluprodia [14]

Para el uso correcto de la librería se tiene que tomar en cuenta 3 aspectos importantes:

- Seleccionar el tipo de diagrama, unidades y fluido que se desea
- Establecer 4 propiedades que serán los límites de la isolínea, 2 límites verticales y 2 límites horizontales
- Enlazar las isolíneas dentro de una base de datos

Considerando el punto anterior, se ejecuta la línea de código con la estructura que se indica en la figura 2.7.

```

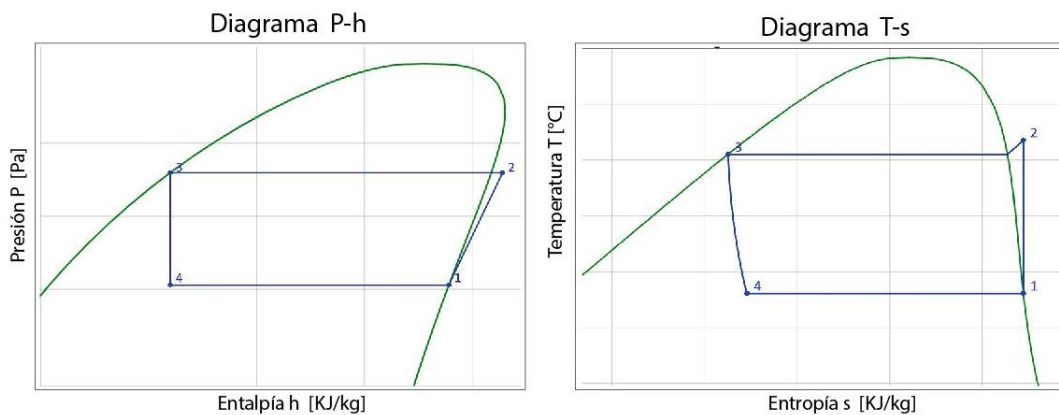
from fluprodia import FluidPropertyDiagram as FPD
from CoolProp.CoolProp import PropsSI
diagram = FPD('R290')
diagram.set_unit_system(T='°C', p='bar', h='kJ/kg', v='m^3/kg')
data = {
    'isentropic': {
        'isoline_property': 's',
        'isoline_value': 2444,
        'isoline_value_end': 2371,
        'starting_point_property': 'p',
        'starting_point_value': 534,
        'ending_point_property': 'p',
        'ending_point_value': 1368}}
for name, specs in data.items():
    data[name]['datapoints'] =
        diagram.calc_individual_isoline(**specs)
diagram.calc_isolines()
diagram.draw_isolines('Ts', isoline_data=mydata)
diagram.save('TS_DIAGRAM.svg')

```

*#Librería de Fluprodia*  
*#Selección de refrigerante*  
*#Unidades del diagrama*  
*#Datos de isolíneas*  
  
*#Puntos verticales*  
  
*#puntos horizontales*  
  
*#Enlace a base de datos*  
  
*#Enlace de datos a diagrama*  
*#Selección del tipo de diagrama*  
*#Guardamos diagrama*

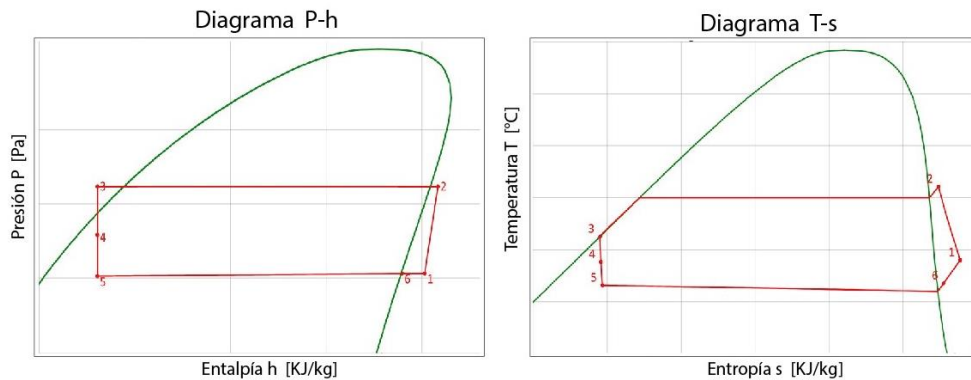
**Figura 2.7.** Línea de código usando Fluprodia

La figura indica una isolínea de ciclo real, estableciendo los límites verticales como la entropía inicial 's' de 2444 kJ/kg.K y entropía final 's' de 2371 kJ/kg.K, posterior se ingresa las propiedades que son los límites horizontales, siendo la presión de alta 'p' de 534 bar y presión de baja 'p' de 1368 bar. Se requieren 4 isolíneas para definir el diagrama de ciclo real Ts y Ph como se detalla en la figura 2.8.



**Figura 2.8.** Diagramas ideales usando Fluprodia

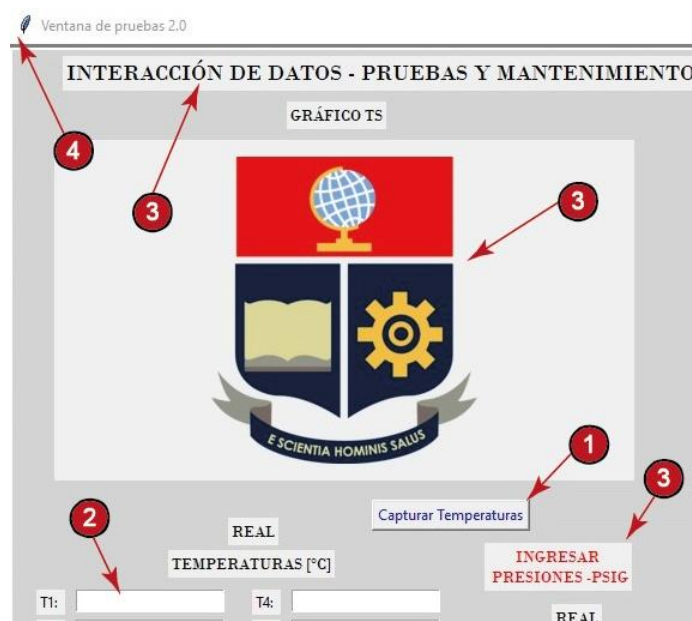
Se requiere 6 isóneas para definir el diagrama de ciclo real Ts y Ph como se detalla en la figura 2.9.



**Figura 2.9.** Diagramas reales usando Fluprodia

### Interfaz gráfica de usuario con librería Tkinter

Python es capaz de diseñar interfaces gráficas (GUI) por medio del uso de librerías como PyQt5 y Tkinter. PyQt5 permite crear los widgets con una metodología visual (similar a visual estudio) mientras genera un código de programación, esto es gracias al programa Qt designer instalado por defecto. Tkinter crea los widgets configurando cada uno un código de programación facilitando la comprensión y configuración que puede llevar interfaz con múltiples funciones [14]. Para la creación de la GUI se elige Tkinter, ya que PyQt5 genera códigos complejos y automáticos, dificultando la comprensión y el enlace con clases y funciones. Los principales comandos que se utiliza para la GUI se indican en la figura 2.10.



**Figura 2.10.** Widgets principales para interfaz gráfica

La interfaz gráfica debe contener la menor cantidad de elementos y ser amigable con el usuario, por tanto de la figura 2.10 se ubican los principales comandos, tales como:

1. **'Button'** ejecuta y enlaza a diferentes funciones.
2. **'Entry'** es una caja de entrada y salida de datos.
3. **'Label'** es un campo de texto e imagen.
4. **'Frame'** es el marco principal en donde se inserta los widgets.

El código de la ventana principal está acompañado con un Frame base, donde se ingresa las dimensiones, posición, título e ícono, un ejemplo se indica en la figura 2.11.

```

from tkinter import*
raiz=Tk()                                #Raiz de todo
raiz.title('Ventana de pruebas 2.0')     #Titulo de ventana
raiz.geometry("1100x680")                #Dimensiones: ancho y alto
raiz.resizable(0,0)                       #Condicion de ventana
raiz.config(bg="lightgray")              #Fondo de ventana

miFrame=Frame()                           #Construir el frame
miFrame.pack()                             #Empaquetar
miFrame.config(bg="lightgray")            #Fondo de frame
miFrame.config(width="1100",height="680") #Dimensiones de frame
miFrame.config(bd=4.5)                     #Grosor del relief
miFrame.config(relief="groove")           #Estilo de borde
raiz.mainloop()

```

**Figura 2.11.** Codificación de ventana principal

Para la codificación de los comandos de la figura 2.10, Tkinter normaliza una estructura en donde llama cada comando y se añade los parámetros que se desee dentro del paréntesis. Posterior se empaqueta cada widget con su ubicación y dimensión dentro del Frame principal que según la figura 2.12 es **'miFrame'**.

```

Button(miFrame,text="Boton 1",           #Texto de boton
       command=lambda:[Funcion_1()]).place(x=300,y=375) #Anexa funciones

Label(miFrame,text="Texto 1",.place(x=485,y=370)        #Texto y posición

P1=StringVar()                                         #Dato de entrada
Label(miFrame,text='Dato de entrada: ').place(x=620,y=366) #Texto y posición
Entry(miFrame,textvariable=P1).place(x=700,y=366)      #Parametro de entrada

T1=StringVar()                                         #Dato de salida
Label(miFrame,text='Dato de salida: ').place(x=620,y=366) #Texto y posición
Entry(miFrame,textvariable=T1).place(x=850,y=395)     #Parametro de salida

```

**Figura 2.12.** Codificación de widgets

Como se visualiza en la figura 2.12, el comando *Button*, permite enlazarse a otras funciones o clases creadas por el usuario por medio de la sintaxis *command=lambda:[ ]*, los parámetros ingresados a este comando son el texto que contenga y la posición dentro del Frame. Además, para asignar un dato de entrada y salida se debe asignar la sintaxis *StringVar()* asignada a una variable, misma que se ocupa como parámetro en el comando *Entry*.

Para ingresar las imágenes al Frame principal, es necesario cargar la imagen con la librería *PIL*, esta va a permitir recortar los diagramas que resulten del análisis gráfico. Primero se carga la imagen, luego se inserta como un *Label* y finalmente se indica la posición respecto al Frame principal, tal como se indica en la figura 2.13.

```

from tkinter import*
from PIL import Image, ImageTk
X=Image.open("NKIM/PH.png")
NI=ImageTk.PhotoImage(X)
lk=Label(miFrame,image=NI,width="480",height="280")
lk.image=NI
lk.place(x=580,y=75)
#Importar librerías
#Obtener imagen
#Comando para abrir
#Parametro en Label
#Ingresar imagen a Frame
#Posición

```

**Figura 2.13.** Codificación para insertar imágenes

Para llevar a cabo una codificación global más sencilla se hace el uso de funciones enlazadas a los botones del Frame principal, cada función debe cumplir las actividades que se detalla en la tabla 2.3.

**Tabla 2.3.** Funciones de botones en el Frame principal

Funciones	Características
Temperatura( )	Permite obtener los 6 datos de temperatura dados por el Arduino.
Calcular( )	Enlaza al método numérico comandado por CoolProp.
Arduino( )	La interfaz permite comunicarse al Arduino de manera constante.
Imágenes( )	Permite crear los diagramas reales e ideales en formato png.
TXT( ).	Genera un archivo en txt de las propiedades termodinámicas.

Se entiende que la codificación de la función *Temperatura()* y *Arduino()* es la intercomunicación con Arduino Mega y la impresión de sus valores en los Entry de la interfaz, *Calcular()* es la resolución numérica y gráfica usado la librería CoolProp y Fluprodia, *Imágenes()* y *TXT()* pertenecen a la codificación de generación de archivos.

## Intercomunicación con Arduino Mega

El módulo Arduino y Split están conectados con 6 termocuplas que ofrecen datos de temperatura de manera constante. Para la intercomunicación con el Arduino se deben utilizar las librerías *serial*, *time* y *serial.tools.list\_ports*, las cuales permiten crear una comunicación constante, recibir y enviar datos en tiempo real. El aplicativo inicia conexión con el Arduino, cada vez que se oprime el botón de Capturar Temperaturas enlazado con la función *Temperatura()*.

El Arduino mantiene un código de ejecución propuesto por los tesisistas de la TIC de construcción de un módulo para Split [14], se debe ingresar la letra 'g' para guardar una línea de datos y la letra 'p' para visualizarla. La intercomunicación debe iniciarse digitando 'g' y 'p' automáticamente cada vez que el usuario desee los datos de temperatura.

```
def ARDUINO(): #Función
    ports = serial.tools.list_ports.comports() #Habilita los puertos
    List1 = [] #Vector vacio
    for port in sorted(ports) #Itera conexión con los puertos
        List1.append(port) #Almacena el puerto con vector
    KK=serial.Serial(z,9600) #Inicia conexión
    time.sleep(1) #1 segundo de tiempo de espera
    g='G'; KK.write(g.encode('ascii')) #Envia letra G de guardar
    g='P'; KK.write(g.encode('ascii')) #Envia letra P de almacenar
    for i in range(9+c): #Almacena cada valor como vector
        val=KK.readline()
        val=val.decode('ascii')
        L.append(val) #Vector L con datos del arduino
```

**Figura 2.14.** Codificación para comunicación con Arduino

El vector L posee los datos de temperatura y es utilizado por las funciones de *Calcular()* y *TXT()* para la generación de los diagramas reales y documentación de la lista de propiedades en formato txt.

### Generación de archivos

Además de los resultados de calor, COP y diagramas, el aplicativo genera 2 archivos adicionales a la ubicación en donde se encuentra el aplicativo. El primero son los diagramas de Ts y Ph ideales y reales superpuestos en formato PNG y el segundo son las propiedades registradas en cada iteración en formato TXT.



```

def TXT():
for i in range (12):
    tt=L[i+12]
    T.append(float(tt))
V12=L[0:9]+['P1', 'P2']+L[9:12]
ValP=T[0:9]+[Pa,Pb]+T[9:12]
for g in range(14):
    I=str(Ij)+str(V12[g])+str('\t')
    M=str(Mj)+str(ValP[g])+str('\t')
    Ij=I;Mj=M
Regf=(Ij+'\n'+Mj)
f = open("Nuevo1F.txt", "w")
i=0
for i in Regf:
    f.write(i)
f.close()

```

*#Transformar cada string a float*  
*#Almacenar en T*  
*#Plantilla con presiones*  
*#Vector con T y presiones*  
*#Unimos Plantilla y valores con espacios*  
*#Plantilla + espacio*  
*#Valores + espacio*  
*#Unimos Plantilla + valores*  
*#Creamos nuevo TXT*  
*#Ingresamos valores al TXT*  
*#Cerramos archivo*

**Figura 2.15.** Codificación para generación de archivo TXT

De la figura 2.15 se puede analizar que la codificación empieza con el vector  $L$ , vector en donde se almacenó la lectura en la comunicación con Arduino. La codificación permite transformar cada valor de carácter simbólico a numérico flotante, además se debe incorporar una plantilla con los valores de presión que se ingresan manualmente. El resultado es el archivo *Nuevo1F.txt* con las propiedades de entalpía, entropía, temperatura y presión en cada estado del ciclo real e ideal.

La codificación para la exportación de los diagramas reales e ideales se detalla como ejemplo las líneas de código de la figura 2.16.

```

def Imagenes():
    diagram.ax.legend(prop={'size':13},loc='upper left')
    diagram.save('Ciclo_TS.png')
    plt.grid()
    MI3=PhotoImage(file='Ciclo_TS.png')
    lk=Label(miFrame,image=MI3,
            width="480",height="280").place(x=580,y=75)

```

*#Ubicación de Leyenda*  
*#Guardar imagen*  
*#Insertamos imagen*  
*#Label para imagen*  
*#Posicion en el frame*

**Figura 2.16.** Codificación para guardar imagen en PNG

La función continua desde la codificación para los diagramas del ciclo, se deben guardar en el formato deseado y ubicarlos en el Frame principal.

## 2.3 Verificación y ejecución del aplicativo

Para comprobar la factibilidad y confianza de los resultados obtenidos por el software diseñado en Python, se comparan sus resultados numéricos y gráficos con otros aplicativos de programación como EES que dispone su propia base de datos del R-290 y un script de Visual Studio cuyas propiedades se obtuvieron tras la interpolación de las tablas de saturación del R-290.

Los errores al comparar sus resultados deben muy inferiores para concluir que el software diseñado es confiable y efectivo para las prácticas de laboratorio y uso didáctico para educación.

### 3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

En este apartado se muestran los resultados procesados por la interfaz gráfica al conectarse al módulo Split. Los resultados numéricos obtenidos se van a comparar con otros aplicativos ya que disponen su propia base de datos de propiedades para el R-290, estos son Visual Studio y EES. Los resultados numéricos y gráficos se van a llevar a cabo con 2 toma de datos distintos.

#### 3.1 Resultados

Una vez finalizada la codificación de la GUI siguiendo la metodología del apartado anterior, se debe convertir el código en un aplicativo ejecutable por medio de la librería *py\_to\_exe* en la consola del ordenador. El producto final se detalla en la figura 3.1.



Figura 3.1. Interfaz Gráfica de usuario

## Registro de datos

Una vez encendido el módulo Split y cargado con el refrigerante R-290, se conecta el Arduino al ordenador de la interfaz gráfica y se registra los siguientes datos de manera automática y manual:

- Primera toma de datos a Temperatura ambiente de 25°C y 15 minutos de estabilización en Split.

**Tabla 3.1.** Registro de datos 1

<b>Propiedad</b>	<b>Valor</b>	<b>Método</b>
Temperatura T1	15.25 °C	Automático - Arduino
Temperatura T2	55°C	Automático - Arduino
Temperatura T3	29°C	Automático - Arduino
Temperatura T4	18.75°C	Automático - Arduino
Temperatura T5	5.25°C	Automático - Arduino
Temperatura T6	2°C	Automático - Arduino
Presión alta	216 psig	Manual – Interfaz
Presión baja	67 psig	Manual – Interfaz

Segunda toma de datos a Temperatura ambiente de 19°C y 5 minutos de estabilización.

**Tabla 3.2.** Registro de datos 2

<b>Propiedad</b>	<b>Valor</b>	<b>Método</b>
Temperatura T1	10.35 °C	Automático - Arduino
Temperatura T2	42.75°C	Automático - Arduino
Temperatura T3	34.75°C	Automático - Arduino
Temperatura T4	15.25°C	Automático - Arduino
Temperatura T5	6.25°C	Automático - Arduino
Temperatura T6	6.5°C	Automático - Arduino
Presión alta	180 psig	Manual – Interfaz
Presión baja	66 psig	Manual – Interfaz

Los datos obtenidos en las tablas se registran a diferentes temperaturas ambientales para que exista una clara variación entre las presiones de los registros, de tal forma se pueda tener resultados más dispersos para una comparación más ilustrativa. Cabe mencionar

que el módulo Split requiere un tiempo de espera de 5 a 10 minutos para que el sistema pueda estabilizarse y tomar los datos correctos.

### Comparación de resultados con diferentes aplicativos

Los registros de datos son procesados por el aplicativo y se obtienen los valores de calor, trabajo y COP. Además, se exportan sus propiedades en un documento formato txt para su análisis con el software EES y Visual Studio, los resultados son los siguientes:

**Tabla 3.3.** Resultados del registro de datos 1

Variable	Software		
	Aplicativo Python	EES	Aplicativo Visual Studio
Calor absorbido ideal	255.06	254.1	256.02
Calor rechazado ideal	-305.15	-304.1	-305.25
Trabajo compresor ideal	50.09	50.02	50.06
COP ideal	5.09	5.08	5.11
Calor absorbido real	305.49	304.9	305.66
Calor rechazado real	-364.15	-364.2	-364.12
Trabajo compresor real	40.85	41.15	41.02
COP real	7.47	7.40	7.43

*Las unidades obtenidas de calor y trabajo están en kJ/kg*

**Tabla 3.4.** Resultados del registro de datos 2

Variable	Software		
	Aplicativo Python	EES	Aplicativo Visual Studio
Calor absorbido ideal	276.75	275.9	276.12
Calor rechazado ideal	-319.39	-318.5	-318.01
Trabajo compresor ideal	42.63	42.57	42.64
COP ideal	6.49	6.48	6.47
Calor absorbido real	292.01	290.8	293.12
Calor rechazado real	-330.69	-330.4	-330.2
Trabajo compresor real	31.93	32.15	32.18
COP real	9.14	9.04	9.10

*Las unidades obtenidas de calor y trabajo están en kJ/kg*

De los datos obtenidos en la tabla 3.3 y 3.4 se puede analizar que la variación entre los valores es muy baja para los 3 software, eso indica que las entalpías obtenidas y el uso de las ecuaciones termodinámicas en el aplicativo de Python son las correctas.

### Errores en resultados de aplicativo Python

Para un análisis más preciso, se obtienen los errores relativos porcentuales, comparando los resultados del aplicativo de Python con EES y Visual Studio en las variables más importantes que son el COP real e ideal.

**Tabla 3.5.** Error en resultados

Registro	Variable	Software	
		EES	Aplicativo Visual Studio
1	Error en COP ideal	0.19	0.39
	Error en COP real	0.94	0.53
2	Error en COP ideal	0.15	0.30
	Error en COP real	1.10	0.43
<i>Los errores obtenidos son porcentuales %</i>			

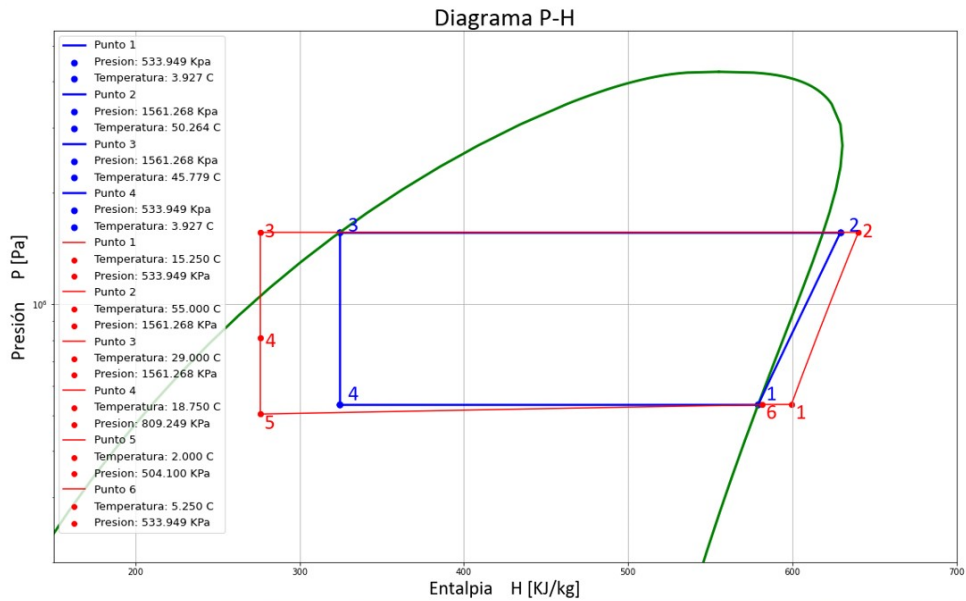
Los resultados obtenidos en la tabla 3.5 muestran que los valores que obtiene el software diseñado en Python tienen un error máximo en el COP real comparado con el software EES de 1.1% en el primer registro y 0.94% en el segundo. Sin embargo todos los demás errores son menores que 1% y se concluye que el aplicativo en Python es un software confiable para su uso en prácticas de laboratorio con el Split.

### Documentos exportados

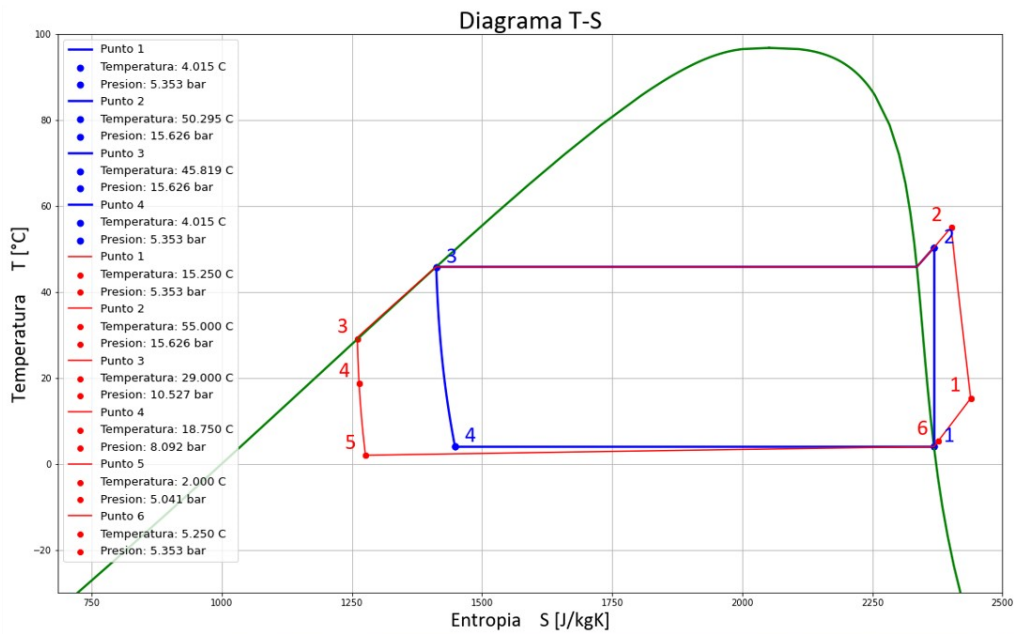
Acerca de la programación de los datos en EES y Visual Studio se pueden visualizar en los anexos II y III. Por otra parte, los diagramas obtenidos del registro 1 se muestran en la parte superior de la interfaz y exportados en formato png. De los diagramas de ciclo de refrigeración se puede constatar el subenfriamiento que tiene el refrigerante R-290 después del condensador y el recalentamiento luego del evaporador, esto permite alterar considerablemente la diferencia entre el COP ideal y real.

**Tabla 3.5.** Temperatura de enfriamiento y recalentamiento

Temperatura	Valor (°C)	
	Registro 1	Registro 2
Recalentamiento	10.3	8.7
Subenfriamiento	15.5	12.9



**Figura 3.2.** Diagrama P-h del registro de datos 1



**Figura 3.3.** Diagrama T-s del registro de datos 1

Los registros de datos se almacenan en el documento *DATOS\_REGISTRADOS.txt* en donde se evidencia las propiedades más importantes de los ciclos reales e ideales, estos son: Temperatura (°C), Presión (psig), Entalpía (kJ/kg), Entropía (J/kgK), Voltaje (V), Corriente (A) y Potencia (W).

ID	T1	T2	T3	T4	T5	T6	P1	P2	V	I	P	H1	H2	H3	H4	H5
1.0	15.25	55.0	29.0	18.75	2.0	5.25	216	67	237.78	4.55	1082.8	599.5	640.7	276.5	276.5	276.5
2.0	10.35	42.75	34.75	15.25	6.25	6.5	180	66	210.25	4.08	985.25	590.9	623.1	292.7	292.7	292.7

**Figura 3.4.** Registro de datos en formato TXT

Por cada iteración que se registre se almacena los datos, donde los primeros conjuntos de propiedades corresponden al ciclo Ideal, y el segundo grupo son del ciclo real.

El presente proyecto dispone de un manual de ejecución que contiene valiosa información recopilada para el uso correcto del aplicativo al momento de conectar el ordenador con el módulo Split, el manual se halla en el Anexo I.

## 3.2 Conclusiones

Por medio de la codificación en Python, se logró obtener una interfaz amigable y de fácil uso para el usuario la cual permite obtener los resultados de calor, trabajo, coeficientes de desempeño y diagramas del comportamiento del refrigerante R-290 en el sistema de refrigeración del módulo Split.

El aplicativo desarrollado se apoyó en la fundamentación teórica y metodología de sintaxis de programación que ofrecen las librerías CoolProp y Fluprodia para determinar las propiedades termodinámicas y la construcción de diagramas termodinámicos Ts y Ph de los ciclos de refrigeración ideal y real.

La depuración del código y los resultados obtenidos dependen del registro de datos experimentales que ofrece el módulo Split, tales como las 6 temperaturas conectadas por termocuplas al Arduino y las 2 presiones conectadas por transductores al display, por lo que si existiera resultados erróneos se debe verificar estas conexiones.

Para comprobar las propiedades, diagramas termodinámicos y resultados de calor, trabajo y coeficientes de desempeño, se comparó los valores obtenidos en Python con resultados obtenidos con la programación de EES y tabulación de tablas con código de ejecución en Visual Studio, mostrando que la interfaz diseñada es confiable para el uso en prácticas de laboratorio.

Existe una gran diferencia entre el COP real e ideal en los registros de datos en las pruebas de funcionamiento, esto se debe a que el módulo Split dispone de tubos de conexión muy largos con poco aislamiento térmico, perdiendo considerablemente calor a lo largo del ciclo de refrigeración.

El coeficiente de desempeño del R-290 es más alto que el R-134a según la recopilación de información, el comportamiento y los diagramas termodinámicos del R-290 son semejantes al R-134a al comparar los resultados con la interfaz gráfica diseñada en Python con otros aplicativos como EES o Visual Studio, confirmando que el R-290 es un refrigerante óptimo y eficiente para su uso en sistemas de refrigeración.

### **3.3 Recomendaciones**

Previo a la manipulación del software, se recomienda para el registro de datos, obtener las temperaturas en unidades relativas de manera automática por medio de la interfaz seguido del ingreso manual de las presiones en unidades de psig, este proceso debe ser secuencial para el registro de nuevos valores.

El módulo Split posee un compresor con auto pagado, por lo tanto es recomendable empezar con la práctica después de 5 a 10 minutos y comprobar que la diferencia de presiones de alta y baja sea notable, una vez que el Split se encuentre estabilizado.

Para la depuración de los códigos mostrados en la metodología y replicar la interfaz gráfica, se debe instalar un ambiente de Python con la versión 3.8, ya que las librerías auxiliares como Coolprop y Fluprodia trabajan de manera conjunta con la sintaxis de programación de Python 3.8.

Después de comprobar el funcionamiento de la interfaz diseñada y comparar sus resultados con otros software, se recomienda comprobar únicamente el estado de la conexión de las termocuplas y del módulo Arduino al existir datos y resultados erróneos en la interfaz gráfica.

Se abren nuevas líneas de investigación para futuros proyectos de titulación que estén relacionados en el perfeccionamiento del método de recopilación de datos de manera automática para las presiones de alta y baja, además para la generación de óptimas interfaces usando el lenguaje de programación de Python y otros refrigerantes.



## 4 REFERENCIAS BIBLIOGRÁFICAS

- [1] C. Romero, J. Rojas, E. Carbajal y J. Rodriguez, «Análisis de irreversibilidades en un sistema de refrigeración por compresión,» *Revista de Sistemas Experimentales*, vol. 4, nº 11, pp. 42-52, 2017.
- [2] T. Legutko y M. Taylor, «Split Systems,» de *Split systems Basic*, New York, Carrier Corporation, 2000, pp. 3-5.
- [3] G. Bejarano, M. Ortega y F. Rubio, Modelado simplificado y orientado al control de sistemas de refrigeracion, Terrassa: Universidad de Sevilla, 2013, p. 506.
- [4] A. Kilicarslan, «Irreversibility analysis of a compression refrigeration cycle using natural refrigerants for a sustainable future,» *Int. J. Exergy*, vol. 10, nº 2, p. 155, 2012.
- [5] T. Kivevele, «Propane (HC – 290) as an Alternative Refrigerant in the Food Transport,» *Journal Unimma - Automotive Experiences*, vol. 5, nº 1, pp. 75-89, 2022.
- [6] B. E. Hermosa y H. Z. Puebla, «Estudio del empleo de hidrocarburos como refrigerantes naturales ecológicos en refrigeración doméstica,» Escuela Politécnica Nacional, Quito, 2012.
- [7] N. J. Yasin, K. A. Jehhef y A. S. Abedalh, «Experimental and theoretical study of heat pump performance enhancement by using a nanorefrigerants,» *International Journal of Mechanical Engineering and Technology*, vol. 10, nº 01, pp. 25-42, 2019.
- [8] S. N. R. P. D. Committee, ASHRAE Handbook - Natural Refrigerants, Atlanta: ASHRAE , 2014.
- [9] E. P. O. T. C. EU, « fluorinated greenhouse gases and repealing Regulation (EC) No 842/2006,» *EU legislation to control F-gases*, 2022.
- [10] V. G. I. Laínez, «Estudio de factibilidad para el uso de hidrocarburos en equipos de refrigeración de vapor,» UNIVERSIDAD DE EL SALVADOR - ESCUELA DE INGENIERÍA MECÁNICA, El Salvador, 2015.
- [11] A. H. Fundamentals, Update on New Refrigerants Designations and Safety Classifications, ASHRAE, 2020.
- [12] K. S. K. K. Ramesha D Ka, «An Overview of Propane Based Domestic Refrigeration Systems,» *PMME* , vol. Proceedings 5, nº University Visvesvaraya College of Engineering, Bangalore University, K R Circle, Bengaluru, p. 1599–1606, 2016.

- [13] B. Ray, D. Posnett y V. Filkov, *A Large Scale Study of Programming Languages and Code Quality in Github*, California: University of California, 2018.
- [14] R. P. Akshit J. Dhruv y N. Doshi, «Python: The Most Advanced Programming Language for Computer Science Applications,» *Proceedings of the International Conference on Culture Heritage, Education, Sustainable Tourism, and Innovation Technologies*, pp. 292-299, 2020.
- [15] Y. Shaheen, H. Alkafrawi y I. Elkafrawi, «Arduino Mega Based Smart Traffic Control System,» *Asian Journal of Advanced Research and Reports*, vol. AJARR.84793, pp. 43-52, 2021.
- [16] S. Bhushan y S. Alok, *Handbook of research methodology*, India: Educreation Publishing, 2017.

## 5 ANEXOS

### ANEXO I



## ESCUELA POLITÉCNICA NACIONAL FACULTAD DE INGENIERÍA MECÁNICA



## CENTRO DE PERFECCIONAMIENTO EN EL USO DE REFRIGERANTES CEPUR

SOFTWARE PROPANE REFRIGERATION:

MANUAL DE USUARIO



KEVIN ISMAEL NARANJO CHISAGUANO

kevin.naranjo@epn.edu.ec

DIRECTOR:

Ing. NARANJO MENDOZA CARLOS ANDRÉS, PhD.

carlos.naranjo@epn.edu.ec

Quito, Febrero 2022

## MÓDULO SPLIT

El módulo Split es un dispositivo que permite el funcionamiento de un sistema de refrigeración por compresión de vapor usando R-290. Sus principales elementos se ubican en la figura 5.1.



**Figura 5.1.** Módulo Split

Para el uso correcto del aplicativo se debe entender los componentes mostrados de la figura 5.1 en donde a y b son la presión de alta y baja respectivamente y t es el módulo Arduino donde muestran las temperaturas.

## SOFTWARE – INSTALACIÓN

El software está diseñado en Python y se puede descargar su código final escaneando el QR de la figura 5.2. Es indispensable que el código se encuentre con la carpeta NKIM, ya que la carpeta dispone de los documentos txt, imágenes e icono que permiten el óptimo funcionamiento del código llamado `Aplicativo_Propano_Split.py`.



**Figura 5.2.** Código QR de software.py

Además el software tiene un archivo ejecutable que se puede descargar escaneando el QR de la figura 5.3. No es necesario una instalación, ya que el software es un documento portable y solo es necesario dar doble clic para empezar. Es necesario que el aplicativo se encuentre en la misma carpeta con la carpeta *NKIM*.



**Figura 5.3.** Código QR de propane refrigerator.exe

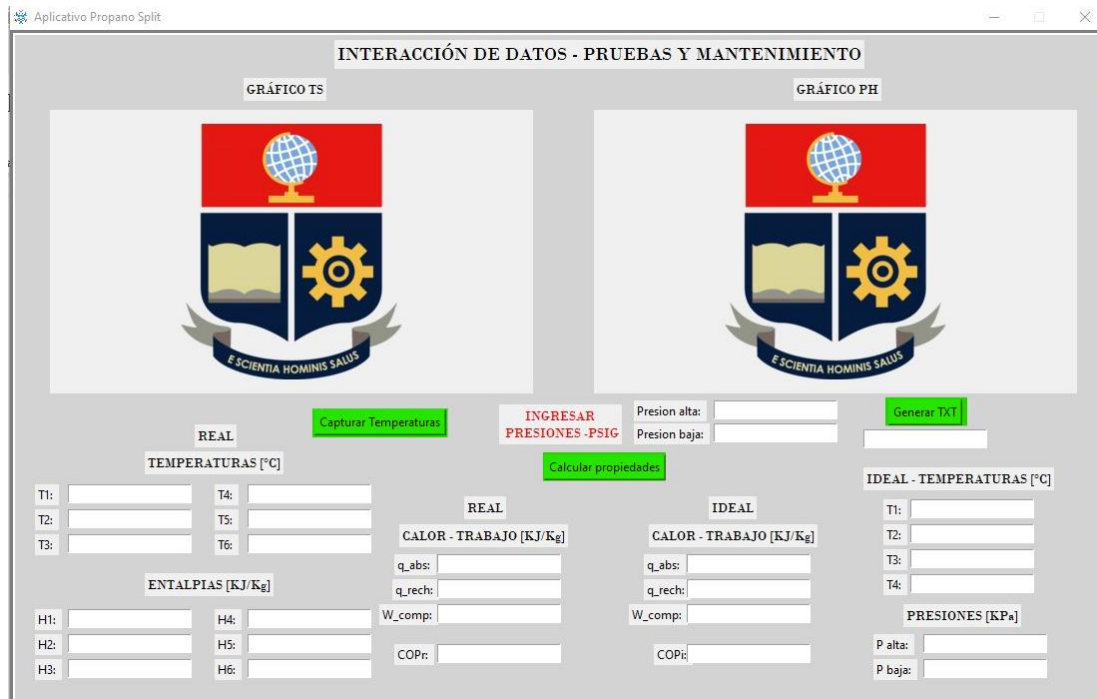
## **FUNCIONES PRINCIPALES**

De la figura 5.4 los botones cumplen las siguientes funciones:

**Capturar Temperaturas:** Permite obtener las temperaturas del módulo Arduino, las mismas que se ubican en los casilleros de *REAL TEMPERATURAS [°C]* de manera automática.

**Calcular Propiedades:** Permite calcular todas las propiedades necesarias para el cálculo de calor, trabajo y COP, además crea los diagramas Ph y Ts de manera automática y las ubica en los laterales superiores. La función permite exportar las imágenes en extensión png a la carpeta *NKIM*.

**Generar TXT:** Permite obtener todas las propiedades en extensión TXT del ciclo de refrigeración real e ideal, el documento se llama *Nuevo1F* y se ubica en la carpeta *NKIM*.



**Figura 5.4.** Interfaz principal

## USO DE SOFTWARE Y SPLIT

1. Encender el módulo Split y fijar una temperatura en el evaporador con el mando a distancia (control).
2. Comprobar que las temperaturas en el display del módulo Arduino tengan valores coherentes y que todos testeen de manera automática.
3. Esperar un lapso de 8 minutos hasta que el Split se estabilice.
4. Se debe conectar el cable de datos - USB al puerto del ordenador antes de iniciar el aplicativo.
5. Iniciar el aplicativo y esperar 15 segundos en promedio para que carguen todas las funciones e imágenes.
6. Capturar las temperaturas desde el Arduino con el botón *Capturar Temperaturas* y esperar que se ubiquen en los casilleros de *REAL TEMPERATURAS [°C]* de manera automática.

7. Si no se obtiene las temperaturas, verificar conexión con el cable USB o revisar puertos.
8. Una vez con las temperaturas obtenidas, se ingresa de manera instantánea las presiones de alta y baja que corresponde al registro de temperaturas del paso F. Si hay mala conexión se debe reiniciar el aplicativo.
9. Se calcula las propiedades y diagramas con el botón Calcular Propiedades, se debe esperar 5 segundos y comprobar si los diagramas tienen la forma adecuada de los ciclos presentados en la metodología.
10. Si los diagramas no son los adecuados, comprobar la coordinación entre el paso F y H, ya que los pasos deben ser secuenciales y los datos coherentes y dependientes uno del otro. Repetir si es necesario.
11. Una vez con las propiedades y resultados correctos, se procede a exportar los resultados con el botón Generar txt.
12. Repetir los pasos F,I y K para un nuevo registro de datos.

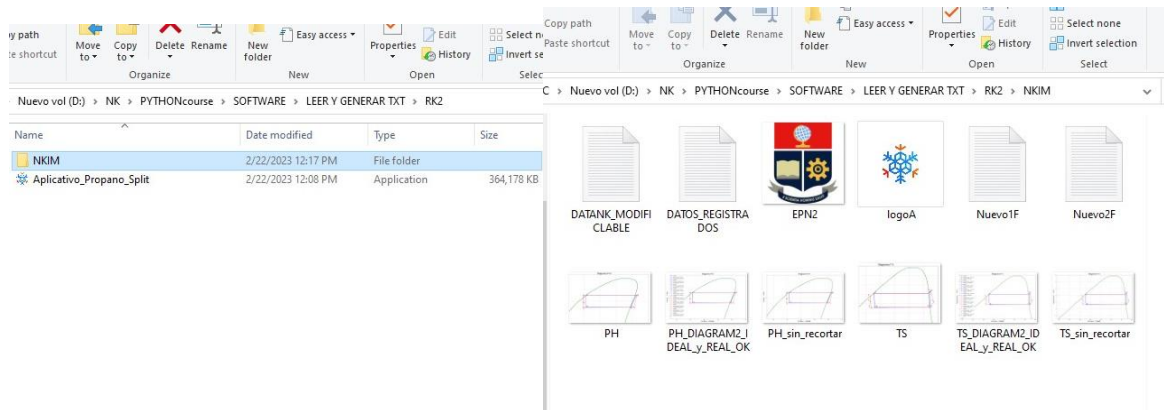
Los pasos también se pueden revisar por medio de un video didáctico para el uso correcto entre la interfaz y el aplicativo, el mismo se ubica en el código de la figura 5.5.



**Figura 5.5.** Código QR de video demostrativo

## **CÓDIGOS EXPORTADOS**

Se obtiene las gráficas del último registro realizado en la carpeta *NKIM*, tal como se muestra en la figura 5.6.



**Figura 5.6.** Carpeta de almacenamiento

Las gráficas PH y TS son las que se muestran en el software cada vez que se ingresa un registro de datos y las gráficas *PH\_DIAGRAMA\_OK* y *TS\_DIAGRAMA\_OK* son las imágenes que se recomienda para su análisis, ya que además de las imágenes existe una tabla que muestra las propiedades en cada punto, tal como indica la figura 5.6.

El documento *Nuevo1F* muestran todas las propiedades termodinámicas necesarias para la construcción de los ciclos termodinámicos, además se almacenan todos los registros de manera vertical. Si se reinicia el aplicativo el documento borra todos los registros de manera automática. La ubicación del documento se halla en la figura 5.6.



# ANEXO II

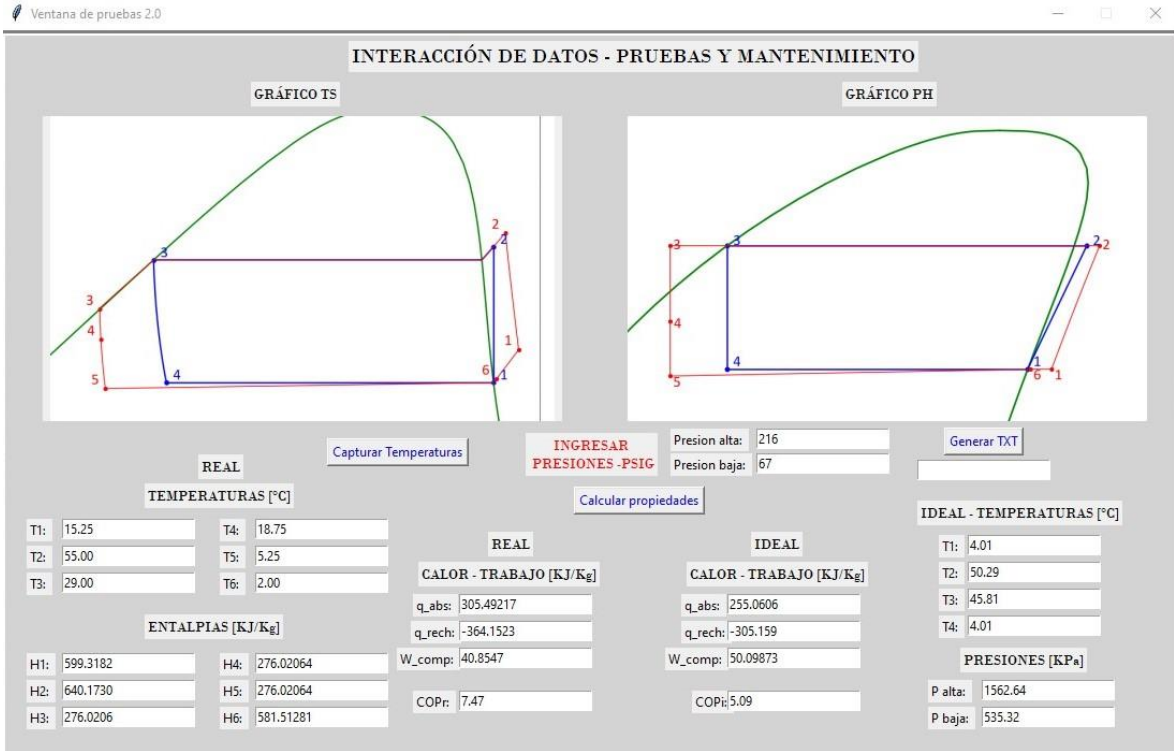


Figura 5.7. Registro de datos 1 con Python

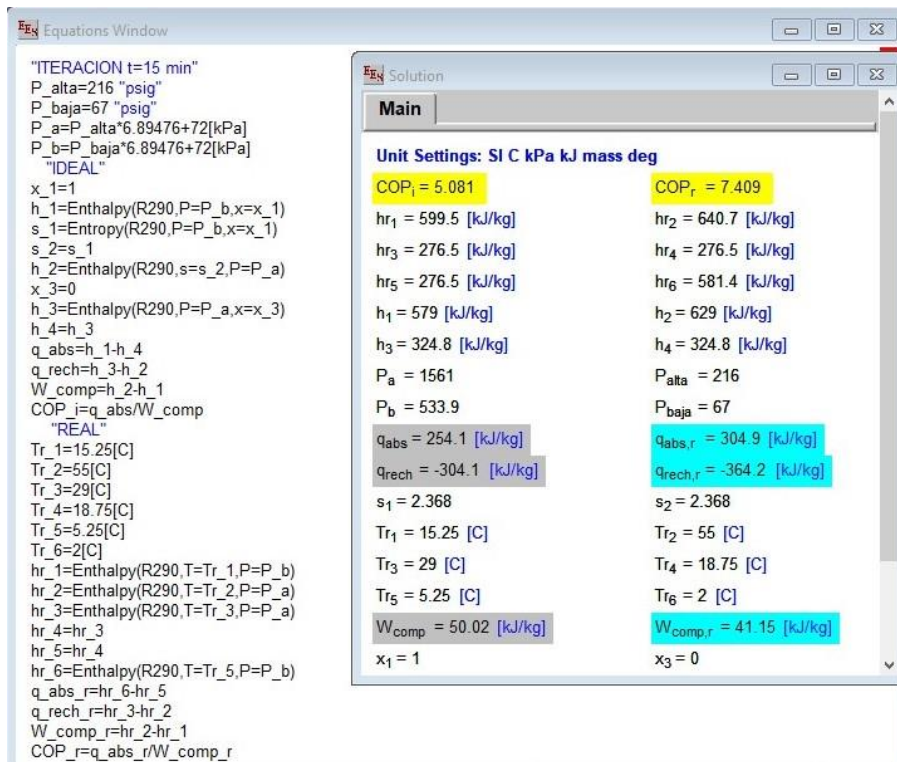


Figura 5.8. Registro de datos 1 con EES

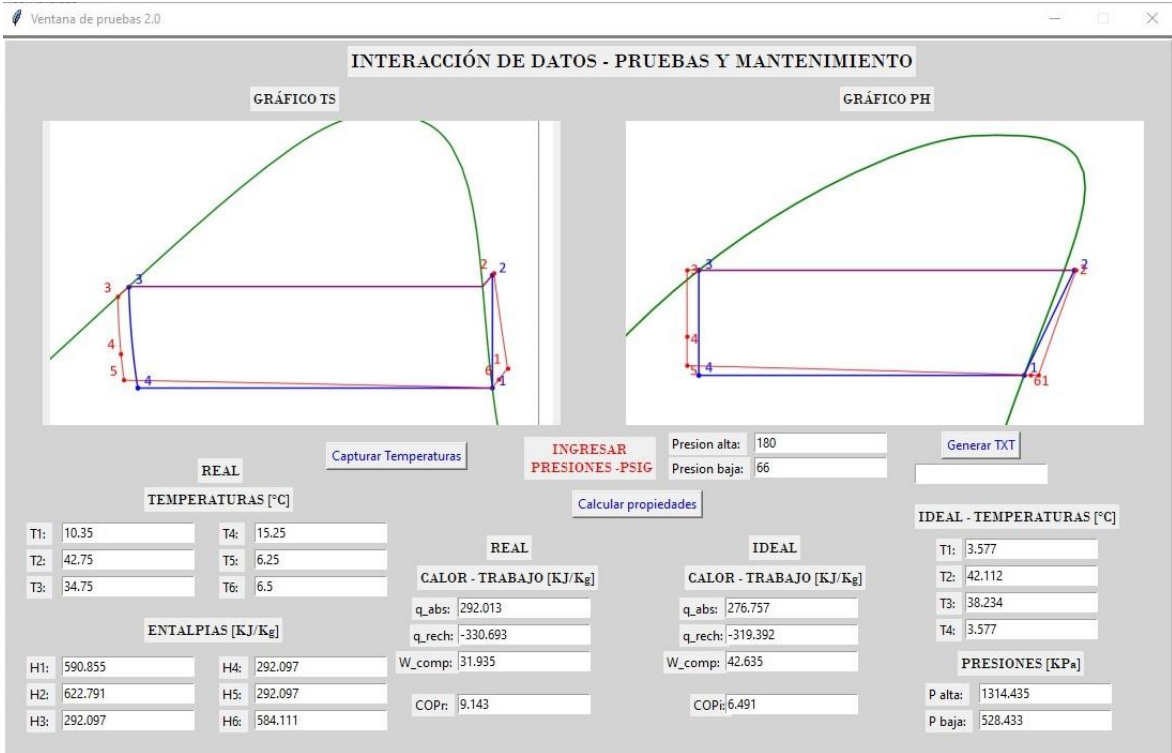


Figura 5.9. Registro de datos 2 con Python

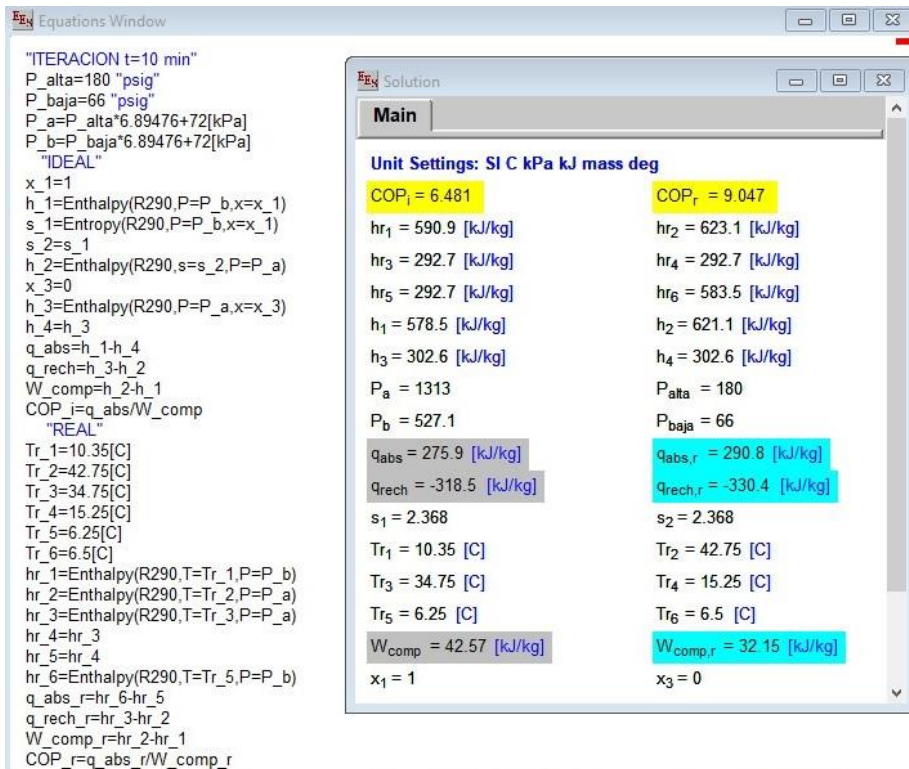


Figura 5.10. Registro de datos 2 con EES