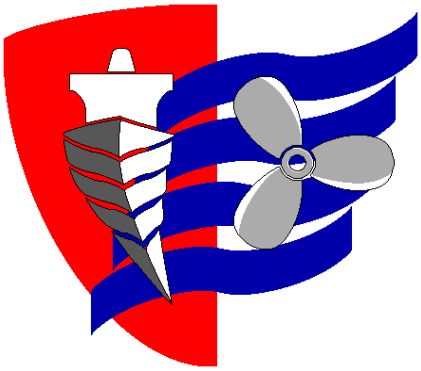
ESCUELA TÉCNICA SUPERIOR DE NÁUTICA

UNIVERSIDAD DE CANTABRIA



Trabajo Fin de Grado

DISEÑO DE UN PROGRAMA PARA EL CÁLCULO DE LOS PARÁMETROS DE FUNCIONAMIENTO DE UNA PLANTA PROPULSORA DE UN BUQUE LNG

DESIGN OF SOFTWARE FOR CALCULATION PARAMETERS OF LNG CARRIER PROPULSION PLANT

Para acceder al Título de Grado en

INGENIERÍA MARÍTIMA

Autor: Agustín Pedraja Crespo
Director: Sergio García Gómez
Septiembre – 2021

ESCUELA TÉCNICA SUPERIOR DE NÁUTICA

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

DISEÑO DE UN PROGRAMA PARA EL CÁLCULO DE LOS PARÁMETROS DE FUNCIONAMIENTO DE UNA PLANTA PROPULSORA DE UN BUQUE LNG

Para acceder al Título de Grado en

INGENIERÍA MARÍTIMA

DESIGN OF SOFTWARE FOR CALCULATION PARAMETERS OF LNG CARRIER PROPULSION PLANT

Septiembre – 2021

ÍNDICE

RESUMEN

El propósito del presente trabajo es el diseño y desarrollo de un software de cálculo de parámetros operativos y de funcionamiento de la planta propulsora de un buque LNG.

Todo el trabajo realizado desde el estudio termodinámico de la planta hasta la construcción del software de cálculo se expone aquí en forma de proyecto técnico abordando todos los aspectos implicados en el mismo y siguiendo la estructura típica de este tipo de documento [3].

ABSTRACT

The purpose of the present work is the design and development of a software for the calculation of operating parameters and performance of a propulsion plant of an LNG ship.

All the work carried out from the thermodynamic study of the plant to the construction of the calculation software is presented here in the form of a technical project addressing all the aspects involved in it and following the typical structure of this type of document [3].

KEYWORDS

LNG, propulsion plant, Rankine cycle, combined cycle, software

PALABRAS CLAVE

LNG, planta propulsora, ciclo de Rankine, ciclo combinado, software

AGRADECIMIENTOS

Al tutor de este TFG por darme la oportunidad de realizar este proyecto y guiarme en la realización del mismo.

MEMORIA

1. GLOSARIO

LNG	Gas natural licuado
HFO	Fuel oil pesado
HRSG	Generador de vapor de recuperación de calor
PCI	Poder calorífico inferior
COGES	Combinado Gas y vapor

2. INTRODUCCIÓN

El estudio del rendimiento de una planta propulsora de vapor o de ciclo combinado depende de muchas variables interconectadas entre sí y que pueden variar a lo largo de su ciclo de funcionamiento y en función de sus condiciones de operación.

Por ello, y para tener una herramienta capaz de mostrar los valores de manera instantánea, obteniendo los resultados pertinentes a las condiciones reales de funcionamiento, en el presente proyecto abordaremos el diseño y desarrollo de un software para realizar el cálculo de los parámetros de mayor relevancia en el funcionamiento de la planta.

3. OBJETO Y ALCANCE DEL PROYECTO

El objeto del presente proyecto consiste en el diseño y desarrollo de un programa informático para realizar el cálculo de los parámetros operativos y

de funcionamiento de la planta propulsora de un buque LNG, a partir de unas condiciones establecidos por el usuario.

Para ello, y dado que el armador cuenta con dos tipos de plantas propulsoras, se contemplarán dos casos distintos permitiendo en la misma aplicación informática realizar el cálculo tanto de una planta propulsora de vapor convencional capaz de funcionar con fuel oil y gas natural, como de una planta de ciclo combinado con turbina de gas y vapor.

4. FUNDAMENTOS TEÓRICOS

En el siguiente apartado, y dado el objeto y alcance del proyecto expuestos, empezaremos por un repaso de los conceptos teóricos claves para entender el funcionamiento de una planta propulsora de vapor y de ciclo combinado, haciendo especial enfoque en los fundamentos termodinámicos [4] [5] [6] que rigen los ciclos de potencia asociados a las máquinas térmicas empleadas como fuente de energía mecánica para la propulsión en este tipo de buque, es decir, las turbinas de gas y vapor [7].

4.1 FUNDAMENTOS DE LA TERMODINAMICA.

La termodinámica es la rama de la física que estudia de forma macroscópica las trasformaciones e intercambios de energía de los cuerpos. Esta ciencia constituye el fundamento básico y principio fundamental en base al cual es posible la utilización de máquinas térmicas que permitan la obtención de trabajo y energía mecánica a partir del calor, permitiéndonos así entre otras aplicaciones su empleo como máquina de propulsión el los buques.

4.1.1 PRINCIPIOS FUNDAMENTALES

Como principales leyes físicas universales establecidas por esta ciencia encontramos cuatro principios universales que regirán cualquier proceso en el que se produzca una transformación de energía, y por tanto condicionarán el funcionamiento y rendimiento de cualquier máquina térmica.

Principio cero: Manifiesta la tendencia de los sistemas al equilibrio térmico. Si un sistema A se encuentra en equilibrio térmico con un sistema B y un tercer sistema C, tanto B como C se encuentran también en equilibrio térmico entre sí.

Primer principio: Manifiesta la conservación la energía, no pudiendo destruirse ni crearse. El incremento de energía interna experimentado por un sistema será siempre igual a la diferencia entre la energía recibida y la energía extraída del mismo independientemente de que estos intercambios se hagan en forma de calor o trabajo.

$$\Delta u = Q + W \tag{1}$$

Segundo principio: Manifiesta la tendencia natural de la energía de fluir siempre desde los cuerpos a temperatura mayor hacia cuerpos a una temperatura inferior. El proceso contrario jamás ocurre de forma espontánea en la naturaleza.

De este principio se deriva la magnitud conocida como **entropía**, la cual permite medir el grado de desorden y degradación de la energía y con ello la pérdida de capacidad para realizar un trabajo útil. Matemáticamente, el incremento de entropía de un sistema se define como la relación entre el calor transferido y la temperatura a la cual se transfiere.

De esta forma, cada vez que el calor se transfiere a una temperatura inferior se está creando entropía. Dado que esta es la tendencia natural de todos

los cuerpos, la entropía es una magnitud que siempre crece o idealmente y en el mejor de los casos se mantiene constante.

Tercer principio: Manifiesta la existencia de una temperatura mínima universal conocida como cero absoluto, la cual es inalcanzable en un número de procesos finito.

4.1.2 SISTEMAS TERMODINÁMICOS

Cuando aplicamos el término “sistema” está haciendo referencia a aquella región a la cual aplicamos el estudio termodinámico para comprender los cambios que se producen en su interior y los intercambios de energía que se producen con su entorno.

Cabe distinguir entre los **sistemas cerrados**, como puede ser el interior de un cilindro, en los cuales la masa del sistema no varía, y los **sistemas abiertos**, los cuales se encuentran continuamente atravesados por una masa como puede ser el caso de una turbina.

4.1.3 EL ESTADO TERMODINÁMICO DE UN SISTEMA

Cualquier sistema termodinámico en equilibrio se encuentra definido por una serie de propiedades macroscópicas concretas que responden a dicho estado. Dichas propiedades macroscópicas se corresponden con las variables de presión, volumen y temperatura.

En el caso de los gases perfectos esas variables quedan interrelacionadas por la fórmula $P \cdot V = nRT$, donde P es la presión, V es el volumen, n es el número de moles, R la constante de los gases ideales y T es la temperatura.

No obstante, de estas variables que de por sí permiten definir el estado del sistema pueden derivarse otras que pueden ser de utilidad a la hora de desarrollar determinados cálculos o estudiar propiedades concretas como son la energía interna, la entalpía o la entropía.

El estado de un sistema puede representarse en un diagrama a partir de las variables que lo definen, típicamente mediante el uso de los diagramas Presión-Volumen, Presión-Temperatura y Temperatura-Presión.

4.1.2 CAMBIOS DE ESTADO, TRANFORMACIONES Y CICLOS CERRADOS

Se denomina **cambio de estado** cualquier transición en la que un sistema termodinámico pasa de un estado inicial a otro final diferente del primero.

Por otra parte, utilizaremos el término **transformación** para referirnos a proceso por el cual se pasa de un estado inicial a otro final pasando por una sucesión continua de estados de intermedios. Cuando se considera que todos estos estados intermedios alcanzan el equilibrio, si representaremos la transformación en el diagrama, utilizaremos un trazo continuo.

Dentro de las infinitas transformaciones que pueden llevarse a cabo en un sistema cualquiera cabe destacar una serie de ellas que son de especial relevancia debido a sus propiedades concretas. Aunque estas trasformaciones en la vida real son difíciles de replicar, nos permiten aproximar los procesos reales que llevan a cabo en las máquinas térmicas para llevar a cabo su estudio. A continuación, se enumeran:

Transformación isócora: Es aquella transformación en la que el volumen del sistema se mantiene constante. Esto implica que el sistema no realiza ni recibe ningún trabajo. El intercambio de energía con el entorno se realiza en forma de calor que va destinado a incrementar o disminuir la energía interna del sistema.

Transformación isóbara: Es aquella transformación en la que la presión del sistema se mantiene constante. El sistema lleva a cabo un intercambio tanto de calor como de trabajo con el medio circundante. Parte del calor absorbido por el sistema se convierte en trabajo y el resto va destinado a incrementar su energía interna.

Transformación adiabática: Es aquella transformación en la que el intercambio de calor con el entorno es cero. Esto implica una situación ideal con un aislamiento perfecto y sin rozamiento de ningún tipo. Reuniendo esas condiciones un proceso adiabático es también isoentrópico, dado que no implica transferencia de calor a una temperatura inferior y, por tanto, sería además un proceso reversible.

Transformación isoterma: Es aquella transformación en la que la temperatura del sistema se mantiene constante. Esto implica que el intercambio de calor se realice entre el sistema y un foco a la misma temperatura que este y con una capacidad ilimitada de absorber o ceder calor.

Al encontrarse el foco externo a la misma temperatura que el sistema la entropía ganada por el sistema será igual a la entropía perdida por el entorno y por tanto atendiendo a la definición ideal de este tipo de transformación, nos encontraríamos ante un proceso reversible.

Por último, una vez introducido el concepto de transformación podemos hablar de la definición de **ciclo cerrado**. Dicho concepto hace referencia a una serie de transformaciones que partiendo de un estado inicial regresan al mismo.

4.2 CICLOS DE POTENCIA

Se denomina ciclo de potencia a todo aquel ciclo termodinámico que opera con un fluido llevando a cabo una serie de transformaciones en el mismo de tal forma que se logra la conversión de calor en trabajo. En este proceso parte del calor aportado inicialmente al ciclo debe ser cedido un foco a una temperatura inferior.

4.2.1 EL CICLO DE CARNOT

Ideado por el ingeniero francés Sadi Carnot, este es un ciclo de trabajo de particular importancia dado sus especiales propiedades de reversibilidad, las cuales hacen de este ciclo el de mayor rendimiento que permiten las leyes físicas.

El ciclo de Carnot tiene la particularidad de que el calor el absorbido y cedido por el sistema siguiendo procesos isotermos, entre los cuales se llevan a cabo procesos adiabáticos para alcanzar las temperaturas correspondientes a cada uno de los focos. Para ilustrarlo de forma más clara puede verse el siguiente diagrama (Figura 1).

Tal como se puede apreciar, el ciclo comienza en el punto A donde se realiza un aporte de calor de forma isoterma a medida que el gas expande hasta alcanzar el punto B. En ese momento se realiza una expansión adiabática hasta alcanzar la temperatura del foco frío T2 en el punto C. A continuación, se extrae calor del sistema de forma isoterma mientras el gas se comprime hasta alcanzar el punto D. Una vez en el punto D, el gas es comprimido de forma adiabática hasta alcanzar el punto inicial, cerrándose así el ciclo.

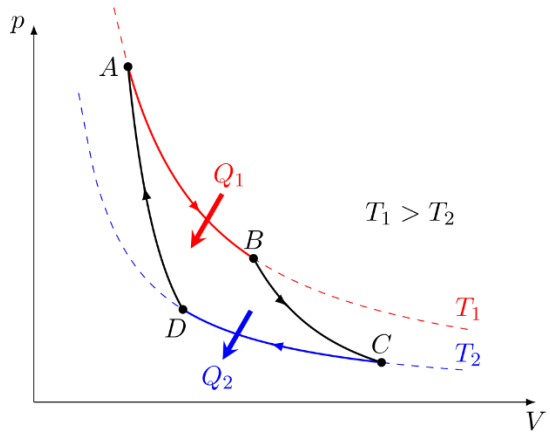


Figura 1: Diagrama de Presión-Volumen del ciclo de Carnot. Fuente: [14]

Entre las particularidades del ciclo de Carnot, y aquel hecho por el que es de tan especial relevancia es que todos los procesos y transformaciones implicados en él son ideales, isostáticos y reversibles, haciendo de él el ciclo de mayor rendimiento que para dos focos de temperatura dados que permite la naturaleza.

Esto quiere decir que cualquier ciclo de potencia que podamos idear tendrá siempre un rendimiento inferior al de una máquina de Carnot que trabaje entre esos dos mismos focos de temperatura.

Cabe destacar que la máquina de Carnot es una máquina ideal, y en por tanto en la práctica irrealizable. No obstante, nos muestra el límite superior de rendimiento que no puede superarse cualquiera sea la máquina térmica que ideemos. Desde este punto de vista el ciclo de Carnot tiene unas implicaciones muy importantes dado que nos demuestra que en todo proceso en el que se pretenda la conversión de calor en trabajo es necesario ceder parte del calor aportado al entono sin posibilidad de convertirlo en trabajo útil.

En el ciclo de Carnot, al igual que en cualquier otro ciclo su rendimiento puede obtenerse mediante la relación entre el trabajo obtenido y el calor aportado mediante la fórmula (ecuación 2).

$$\eta = \frac{W}{Q} = \frac{Q_1 - Q_2}{Q_1} = 1 - \frac{Q_2}{Q_1} \tag{2}$$

La cual, si tenemos en cuenta la particularidad de que el aporte y cesión del calor se realiza siguiendo procesos isotermos, desarrollándola de la forma adecuada nos llevará a la conclusión de que el rendimiento de una máquina de Carnot puede expresarse en función de las temperaturas de sus focos de la forma (ecuación 3):

$$\eta = 1 - \frac{T_2}{T_1} \tag{3}$$

4.2.2 EL CICLO DE RANKINE

Inventado por el ingeniero escoces William John Macquorn Rankine, el ciclo de Rankine es uno de los ciclos termodinámicos de mayor relevancia y aplicación en la actualidad en el campo de generación eléctrica, así como también en el de la propulsión naval, tema que particularmente nos atañe en el presente proyecto.

El ciclo en cuestión se compone de 4 etapas comprendidas a lo largo del transcurso del fluido por el circuito ilustrado en la figura 2:

-Una **compresión isoentrópica** del fluido mediante el uso de una bomba hasta alcanzarse una presión determinada.

-Un **calentamiento a presión constante** en una caldera hasta alcanza una temperatura determinada.

-Una **expansión isoentrópica** en una turbina hasta alcanzarse de nuevo la presión inicial.

-Un **enfriamiento a presión constante** en un condensador hasta alcanzar la temperatura del punto inicial.

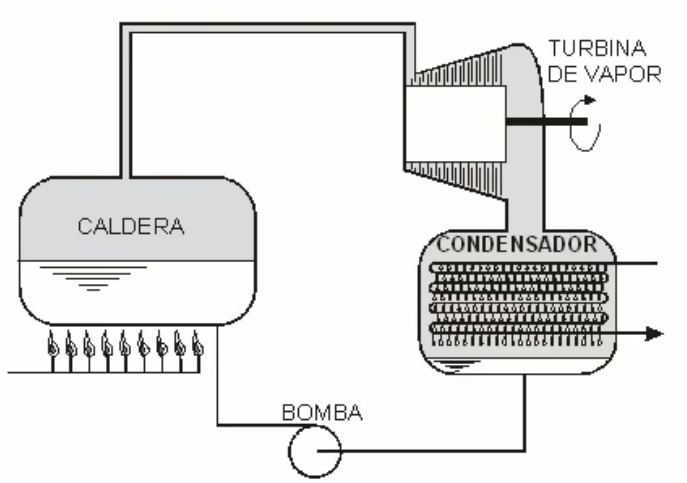


Figura 2: Esquema de una planta de vapor basada en el ciclo de Rankine. Fuente: [15]

El ciclo de Rankine presenta la particularidad de trabajar con un fluido (el agua) que presenta un cambio de fase entre su estado de líquido y vapor a lo largo del ciclo. Esto en un primer momento se puede considerar una ventaja a la hora de buscar una aplicación directa del ciclo de Carnot considerando la facilidad de mantener una temperatura constante cuando el aporte de calor al fluido de trabajo se realiza en el interior de la campana de saturación.

De esta forma podríamos pensar que en la práctica el ciclo de Carnot podría aplicarse al vapor situando todas las transformaciones del ciclo en el interior de la campana de saturación, tal como se muestra en la figura 3.

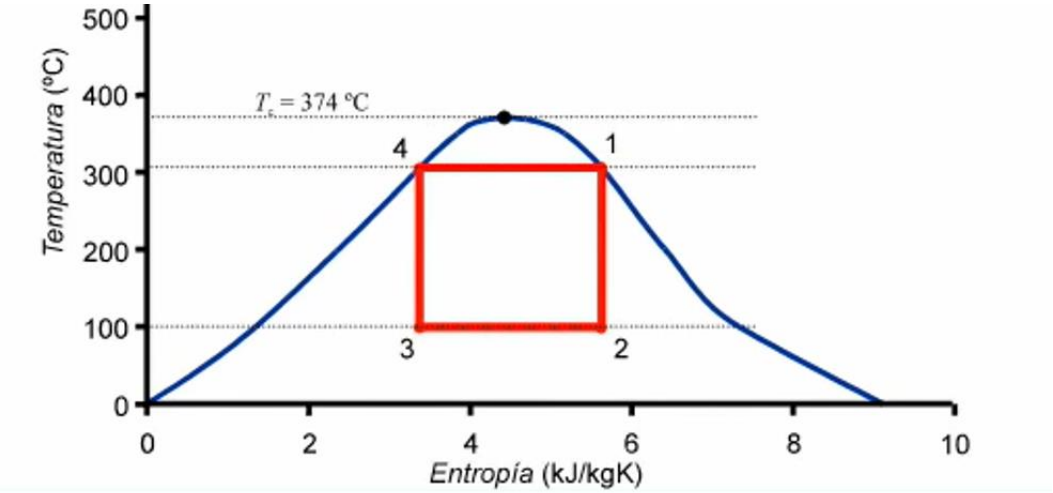


Figura 3: Diagrama de Temperatura-Entroía del ciclo de carnot aplicado al vapor de agua. Fuente: [16]

No obstante, en la práctica esta idea presenta una serie de inconvenientes.

El primero de ellos es el pobre rendimiento que presenta la bomba al tener que trabajar con vapor. Es por este motivo que, en la práctica, en el enfriamiento se alcanza la zona de líquido saturado para garantizar que la bomba no tenga que trabajar con vapor.

Por otra parte, la turbina podría presentar daños en los álabes si la expansión del vapor se realiza en la zona de vapor saturado. Es por eso que en la practica se realiza un sobrecalentamiento del vapor para garantizar en la medida de lo posible que la turbina no tenga que trabajar en esa zona.

De la aplicación de estas dos modificaciones surge que el ciclo de Rankine tal como lo conocemos y se ilustra en la figura 4.

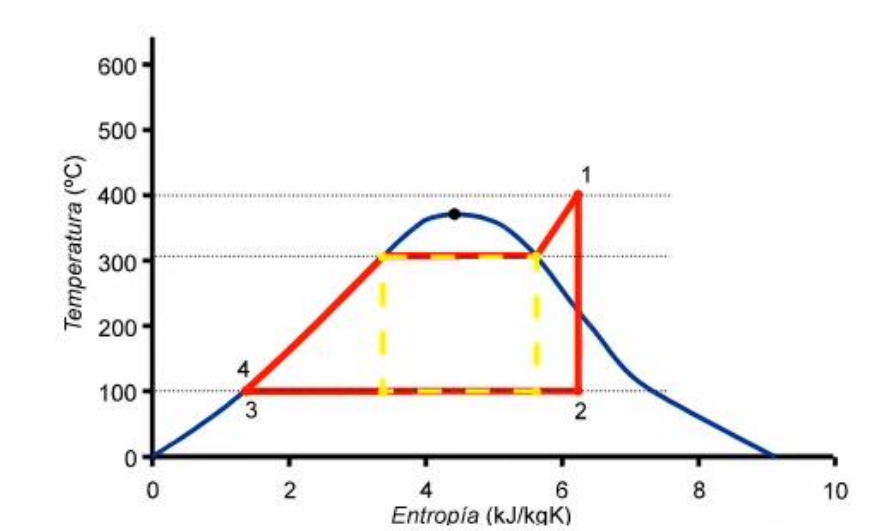


Figura 4: Diagrama de Temperatura-Entropía del Ciclo de Rankine básico. Fuente: [15]

4.2.2.1 BALANCE ENERGÉTICO Y RENDIMIENTO DEL CICLO DE RANKINE

Al igual que en cualquier otro ciclo de potencia, en base al principio de conservación de la energía, la suma de la energía entregada al sistema en forma de calor o trabajo será igual a la suma del trabajo y calor extraídos. (ecuación 4)

Es decir, se cumplirá la igualdad siguiente:

$$Qcaldera + Pbomba = Pturbina + Qcondensador \quad (4)$$

Por su parte, cada uno de estos términos podrá obtenerse en base al salto entálpico entre la entrada y salida de cada uno de ellos y el caudal másico que los atraviesa de la forma siguiente: (ecuaciones 5, 6,7 y 8)

$$Qcaldera = m \cdot (h1 - h4) \quad (5)$$

$$Qcondensador = m \cdot (h2 - h3) \quad (6)$$

$$Pturbina = m \cdot (h1 - h2) \quad (7)$$

$$Pbomba = m \cdot (h4 - h3) \quad (8)$$

En lo referente al rendimiento se corresponderá al igual que en cualquier otro ciclo de potencia con la relación entre la potencia neta y el calor aportado en la caldera. (ecuación 9)

$$\eta = \frac{Pturbina - Pbomba}{\text{Qcaldera}} = \frac{m(h1-h2) - m(h4-h3)}{m(h1-h4)} = \frac{h1-h2-h4+h3}{h1-h4} \quad (9)$$

4.2.2.2 CICLO DE RANKINE CON RECALENTAMIENTO

Una de las modificaciones más habituales respecto al ciclo de Rankine básico es el uso de recalentamientos intermedios durante el proceso de expansión del vapor. De esta forma, es habitual tener una turbina de alta presión que expande el vapor hasta una presión intermedia. A continuación, se eleva la temperatura de ese vapor de nuevo, y por último termina de expandirse en una turbina de baja presión hasta alcanzar la presión del condensador.

Con este tipo de modificación se consigue una mejora tanto del rendimiento del ciclo como de la potencia obtenida.

En las figuras 5 y 6 podemos observar cómo varía el esquema del circuito respecto al ciclo de Rankine básico, así como el nuevo diagrama t-s representativo del mismo.

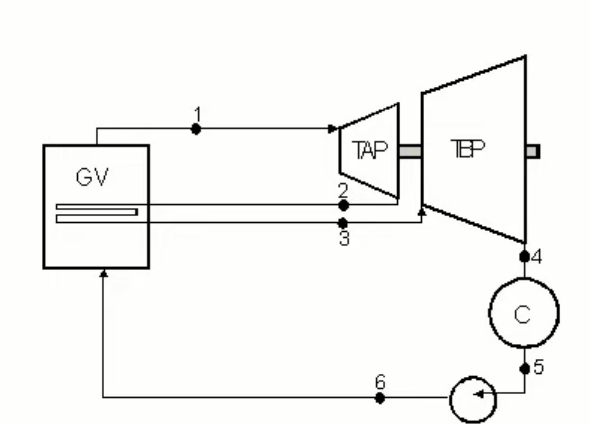


Figura 5: Esquema de una planta de vapor basada en el ciclo de Rankine con un recalentameinto intermedio. Fuente: [17]

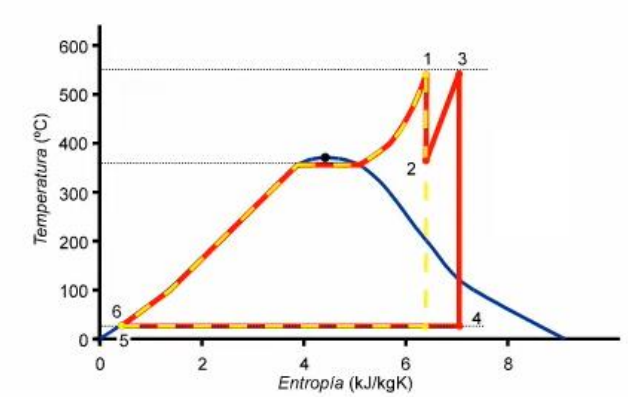


Figura 6: Diagrama de Temperatura-Entropía del ciclo de Rankine con un recalentmiento intermedio. Fuente: [17]

A diferencia del ciclo Rankine simple, ahora el aporte de calor se realizará en dos etapas, al igual que la expansión del mismo, teniendo que considerarse la potencia producida por cada una de las turbinas.

4.2.2.3 CICLO DE RANKINE CON REGENERACIÓN

Esta mejora del ciclo Rankine básico se emplea con el objetivo de lograr un mayor rendimiento. Consiste en realizar una o más extracciones de vapor en puntos intermedios de su expansión y utilizar el calor del mismo para el precalentamiento del agua de alimentación antes de entrar en la caldera.

Este intercambio de calor se realiza mediante el uso de un calentador, existiendo dos tipos, el de superficie, en el cual no existe contacto directo entre el agua y el vapor de extracción, y el de mezcla en el cual sí se produce. Este último es el caso ilustrado en la figura 7, cuyo diagrama temperatura entropía podemos ver en la figura 8.

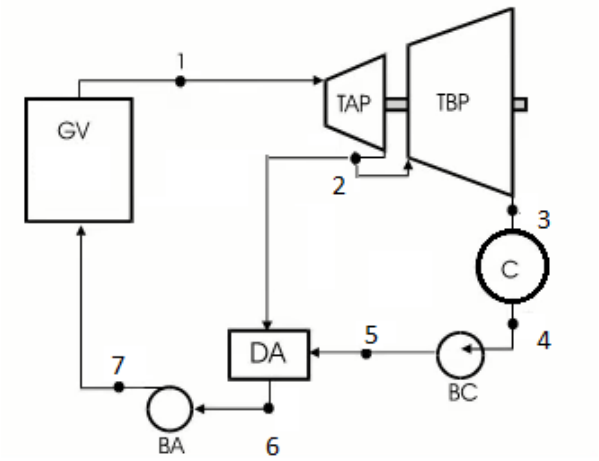


Figura 7: Esquema de una planta de vapor basada en ciclo de Rankine con una regenración mediante intercambiador de mezcla. Fuente [18]

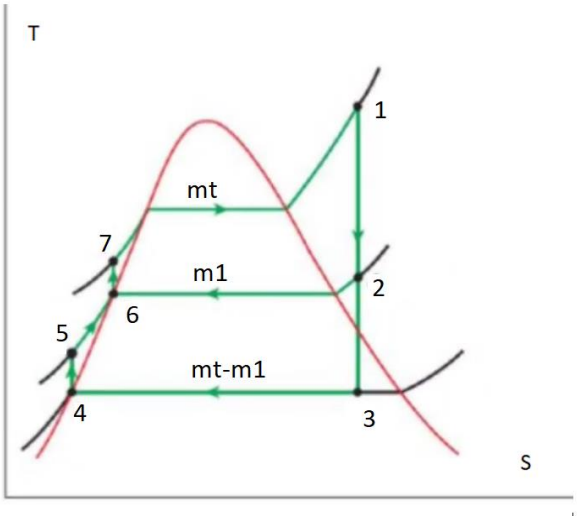


Figura 8: Diagrama de Temperatura-Entropía del ciclo de rankine con una regeneración mediante intercambiador de mezcla. Fuente: [20]

Partiendo del caso ilustrado en el diagrama de la figura 8, para el cálculo del caudal de extracción será necesario hacer un balance de entalpía en el punto 3, así como un balance de los caudales másicos en ese punto cumpliéndose lo siguiente: (ecuación 10)

$$(mt - m1) \cdot h5 + m1 \cdot h2 = mt \cdot h6 \tag{10}$$

En relación a la potencia de la turbina habrá que tener en cuenta que el caudal másico que circula por la misma será diferente en cada una de las 2 etapas. Es decir, la potencia de la turbina será: (ecuación 11)

$$Pturbina = (h1 - h2)mt + (h2 - h3)(mt - m1) \tag{11}$$

En lo referente al rendimiento, el cálculo será igual que en el caso del ciclo Rankine simple, considerando la relación entre la potencia neta del ciclo y el calor aportado en la caldera. (ecuación 12)

$$\eta = \frac{Pturbina - Pbombas}{\text{Qcaldera}} \tag{12}$$

4.2.3 EL CICLO BRAYTON

El ciclo Brayton es un ciclo de potencia de amplia aplicación en el contexto de la producción de energía y también de la propulsión naval, especialmente es muy frecuente su uso conjuntamente con el ciclo de Rankine en las llamadas plantas de ciclo combinado, las cuales explicaremos más adelante.

El ciclo Brayton teórico consta en sí mismo de cuatro etapas de acuerdo a lo ilustrado en el diagrama de la figura 9:

-**Compresión isoentrópica** por medio de un compresor de aire hasta alcanzar una presión determinada.

-**Aporte de calor a presión constante** hasta alcanzar una determinada temperatura.

-**Expansión isoentrópica** en una turbina de gas hasta alcanzar la presión inicial.

-**Enfriamiento isóbaro** hasta alcanzar la temperatura inicial.

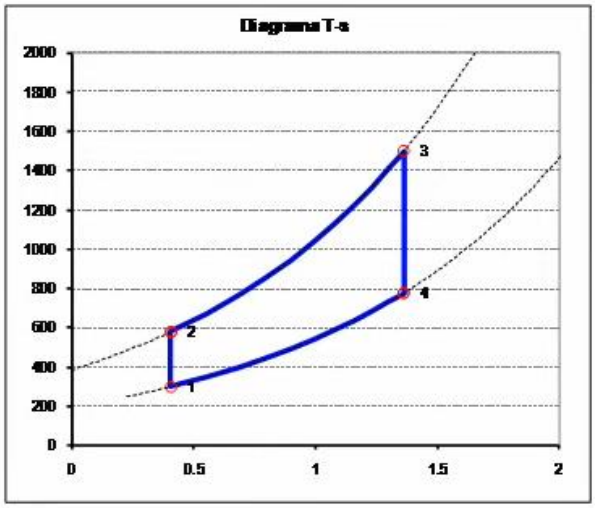


Figura 9: Diagrama de Temperatura-Entropía del ciclo Brayton básico. Fuente: [19]

A continuación, en la figura 10 podemos ver el esquema de lo que sería una configuración típica de una planta que funcione de acuerdo al ciclo Brayton, en la cual encontramos una turbina de gas y un compresor, ambos montados de forma solidaria en un mismo eje, así como una cámara de combustión, donde se quema el combustible y se realiza el aporte de calor a los gases.

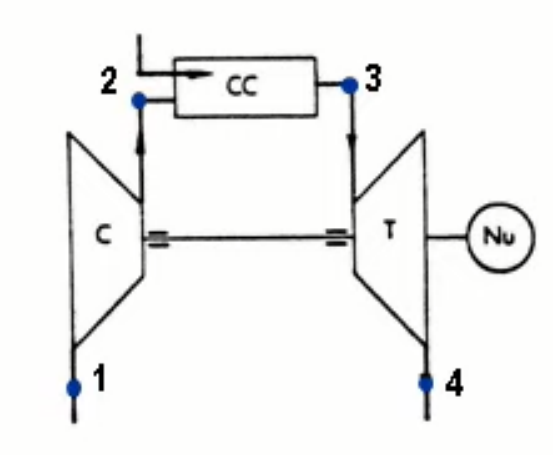


Figura 10: Esquema de una planta de gas basada en ciclo Brayton. Fuente: [19]

En lo referente al cálculo de rendimiento del ciclo Brayton, al igual que en el caso de cualquier otra máquina térmica consideraremos la relación entre la potencia neta (la aportada por la turbina restándole la absorbida por el compresor) y el calor aportado en la cámara de combustión. (ecuación 13)

$$\eta = \frac{Pturbina - Pcompresor}{\text{Qcombustión}} \tag{13}$$

4.2.4 CICLOS COMBINADOS

El ciclo Brayton por sí mismo y utilizado en su forma simple sin aplicar mejoras o modificaciones de ningún tipo, presenta el inconveniente de tener un rendimiento muy bajo dada la elevada temperatura con la que los gases de escape deben ser cedidos al foco frío tras su expansión en la turbina.

Una forma de aprovechar este calor que de otra forma desecharía sin realizar trabajo útil es emplearlo en una caldera de recuperación tal como se ilustra en la figura 11, de forma que sirva de aporte de energía para la producción de vapor que pueda ser expandido en una turbina.

De esta forma, mediante la utilización combinada del ciclo Brayton y el ciclo de Rankine obtenemos una mejora en la potencia y el rendimiento que ambos tendrían por separado.

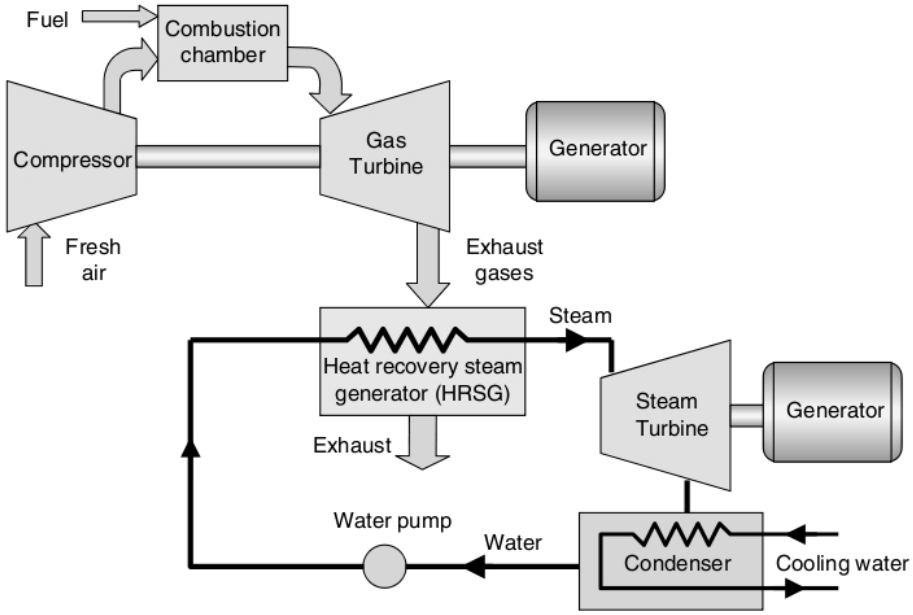


Figura 11: Esquema de una planta de ciclo combinado. Fuente: [21]

5. DESCRIPCIÓN DE LA PLANTA PROPULSORA DE VAPOR

En este apartado abordaremos la composición de la planta propulsora de vapor que se utilizará como referencia para el desarrollo del programa. Trataremos sus distintos elementos desde un punto de vista descriptivo y mostraremos sus diferentes características técnicas y valores de sus parámetros de funcionamiento, los cuales tendrán su relevancia de cara a la construcción del software de cálculo objeto de este proyecto.

5.1. CARACERÍSTICAS TÉCNICAS DE LA PLANTA PROPULSORA

Para el desarrollo del proyecto se ha tomado como base la planta propulsora cuyo esquema simplificado podemos ver en la figura 1. Se trata de la planta propulsora de vapor un buque LNG [1].

Consta de una turbina compuesta transversal con turbina de alta y baja presión, así como turbina de ciar. En el esquema podemos apreciar los distintos elementos que compone la planta: turbina principal, condensador principal, calentador de baja presión, desaireador, caldera y bombas de alimentación.

De la misma forma también observamos la existencia de varios elementos auxiliares, entre los que se encuentra un turbogenerador, encargado de producir energía eléctrica para todos aquellos equipos de la planta que la requieran, así como una turbobomba encargada de elevar la presión del vapor a la salida del desaireador hasta alcanzar la presión de la caldera.

Ambos elementos funcionan utilizando vapor recalentado obtenido a la salida de la caldera.

Por otra parte, observamos que a lo largo de la expansión en la turbina principal existen tres extracciones de vapor. La primera de ellas se destina al calentamiento del FUEL-OIL para su posterior combustión en la caldera.

La segunda extracción va destinada al calentador de baja presión, y la tercera de ellas va destinada a un intercambiador de mezcla que además cumple las funciones de desaireador.

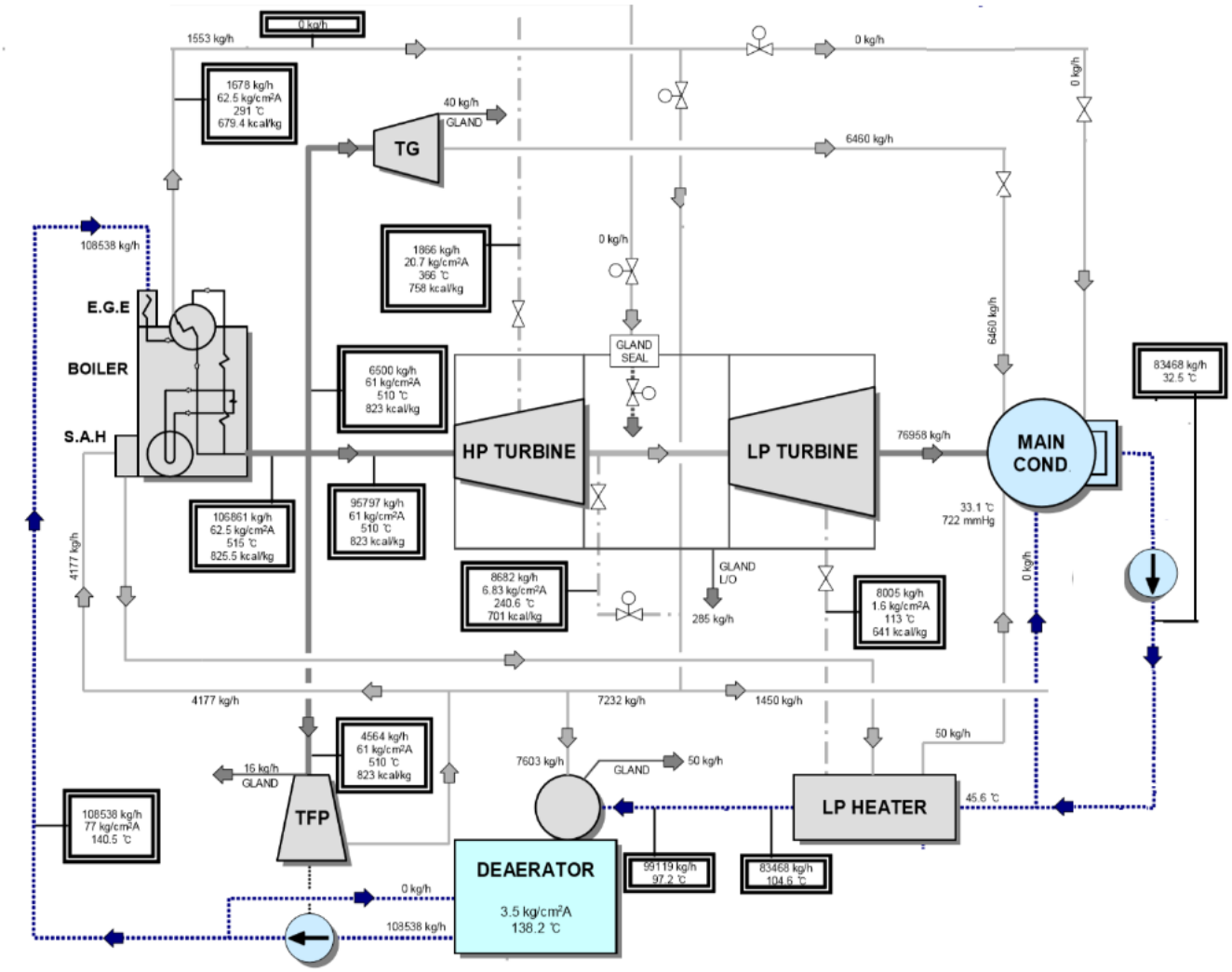


Figura 12: Esquema simplificado de la planta de propulsión. Fuente: [1], página 246

A continuación, en la tabla 1 podemos encontrar los diferentes parámetros característicos de la planta de propulsión en lo referente a pontencia y par suministrados por la turbina y entregados al eje de cola en función de las condiciones de carga de trabajo y régimen de giro, así como otros parámetros relativos a la presión y temperatura del vapor de la planta.

		MCR	NCR
Potencia de salida (Kw)		26985	24286
Revoluciones	Turbina HP	4962	4793
	Turbina LP	3275	3163
	Hélice de propulsión	88	85
Presion de vapor a la entrada de la valvula de maniobra		6MPa	
Temperatura del vapor a la entrada de la válvula de maniobra		510ºC	
Vacío en el condensador para la máximo caudal de salida y temperatura del agua de mar = 24ºC.		96,25 KPa	
Máximo par de la turbia de ciar para el 50% de las rpm MCR adelante.		80% del par MCR	
Máximo velocidad de rotación continua para la turbina de ciar.		70% de las rpm MCR adelante	
Dirección de rotación hacia delante.		Sentido horario visto desde popa	
Velocidad de rotación eje principal en vibración torsional		28,15 rpm y 58,48 rpm	
Velocidad crítica de rotación del rotor (convertido a rotación del eje cola)		Turbina HP	63,4 rpm
		Turbina LP	118 rpm

Tabla 1: Características de la planta propulsora de vapor. Fuente: [1], página 10

5.2. TURBINA PRINCIPAL

La unidad de propulsión principal es una turbina compuesta transversal del tipo Kawasaki UA con doble engranaje de reducción. Esta unidad consta de una turbina de alta presión de 10 etapas de acción, una turbina de baja presión

con 8 etapas de reacción, turbina de ciar con dos etapas de reacción, unidad de válvula de maniobra y reductora de doble engranaje.

El vapor es controlado únicamente por la unidad de válvula de maniobra. El vapor procedente de la válvula de maniobra es admitido en la turbina de alta presión a través del elemento de tobera de la primera etapa.

Los prensaestopas de la turbina están sellados con vapor; el sello es del tipo laberintico.

A continuación, en la figura 3 se muestra en una tabla los datos relativos a la potencia y otras características técnicas y constructivas de la turbina.

Turbina:	
Modelo	Kawasaki UA
Tipo:	Turbina de alta y baja presión con componente transversal.
Potencia:	-MCR. 26985 kW a 88 rpm. -NCR. 24286 kW a 85 rpm.
Vapor Recalentado:	60 Kg/cm2 a 510 ºC
Número de etapas	Turbina de alta presión (HP) - 10 etapas Turbina de baja presión (LP) - 8 etapas Turbina de ciar - 2 etapas

Tabla 2: Características de la turbina principal. *Fuente: [1], página 10*

5.3 CALDERAS

El vapor recalentado necesario para el funcionamiento de la turbina principal, así como del turbogenerador auxiliar o la turbobomba de alimentación es producido utilizando dos calderas acuotubulares Mitsubishi MB-4E.

Este modelo de caldera consta de 2 quemadores situados en el techo del hogar diseñados para uso combinado de Fuel-Oil y gas natural. Cada caldera tiene 2 colectores, uno inferior de agua y uno superior de vapor.

Entre los accesorios que incluyen las calderas encontramos recalentador, desrecalentador, economizador, calentador de aire y control de temperatura de recalentamiento.

A continuación, en la figura 3 se muestran en una tabla los datos relativos a la capacidad de vaporización y otras características técnicas y constructivas de las calderas.

Caldera	
Modelo	Mitsubishi MB-4E
Cantidad	2
Tipo	Acuotubular vertical con dos colectores
Capacidad máxima de evaporación	56000 kg/h
Capacidad de evaporación nominal:	49000 kg/h
Vapor Recalentado:	61,5 kg/cm^2 y 515 ºC

Tabla 3: Características de las calderas. Fuente: [1], página 18

5.4 SISTEMA DE CONDENSADO

El sistema de condensado principal, como parte del ciclo cerrado de alimentación, es la sección de la circulación del agua de alimentación desde el condensador principal hasta las bombas de alimentación principal a través del desgasificador.

El vapor de escape de las turbinas principales, los turbogeneradores, el vapor de descarga y otros auxiliares se condensa en vacío en el condensador principal refrigerado por agua de mar.

El agua condensada se extrae mediante una bomba de agua principal y circula a través de varios intercambiadores de calor antes de entrar en el desgasificador que está situado en un punto alto de la sala de máquinas.

El agua del desaireador proporciona a las bombas principales de alimentación una altura de aspiración positiva.

A continuación, en la tabla 3 se muestran los valores de los parámetros técnicos más relevantes del condensador principal.

vacío	mmHgV	722
Superficie de refrigeración	m^2	2400
Caudal de vapor condensado	Kg/h	83692
Temperatura de entrada del agua de enfriamiento	ºC	24
Caudal de agua de enfriamiento	m^3/h	13300

Tabla 4: Características del condensador. Fuente: [1], página 10

5.5 REDUCTORA

La caja reductora de la planta de propulsión es de tipo articulado en tándem con doble reducción y engranajes cilíndricos helicoidales. En la tabla 5 se muestra los datos relativos al diámetro y número de dientes de cada uno de los engranajes que componen la reductora.

		Diámetro (mm)	Nº dientes
Primer piñón reductor	HP	291,0	36
	LP	396,1	49
Primera corona reductora	HP	2626,9	325
	LP	2360,2	292
Segundo piñón reductor	HP	658,2	57
	LP	658,2	57
Corona principal		4110,7	356

Tabla 5: Características de la reductora. Fuente: [1], página 10

5.6 MAQUINARIA AUXILIAR

Entre la maquinaria auxiliar de la planta destaremos algunos de los equipos de mayor relevancia, entre los que se encuentran un turbo generador, un generador diésel, la bomba de condensado y la bomba de alimentación de la caldera, cuyas características técnicas pueden ser de relevancia para el desarrollo de este proyecto y por tanto se muestran las tablas 6, 7, 8 y 9 respectivamente.

DIESEL GENERADOR	
Potencia	4950 BHP
Velocidad máxima	720 rpm
Nº tiempos	4
Tipo de combustible	Diesel oil
Fabricante	FTX Corporation

Tabla 6: Características del diésel generador. Fuente: [1], página 262

TURBO GENERADOR	
Tipo	Horizontal multietapa
Potencia	3450 kW
Velocidad máxima	1800 rpm
Vapor recalentado	6 MPa 510ºC
Fabricante	Mitsubishi heavy ind Ltd

Tabla 7: Características del turbo generador. Fuente: [1], página 262

BOMBA DE ALIMENTACIÓN DE AGUA PRINCIPAL	
Tipo	Monoetapa, movida por turbina de vapor
Caudal máximo	140 m3/h
Presión máxima	85,5 bar
Fabricante	Coffin turbo pump Inc

Tabla 8: Características de la bomba de alimentación principal. Fuente: [1], página 263

BOMBA DE CONDENSADO PRINCIPAL	
Tipo	Vertical, centrífuga
Caudal máximo	110m3/h
Presión máxima	10 bar
Potencia motor accionamiento	60 kw
Velocidad motor de accionamiento	1800 rpm
Fabricante	Shinco ind co

Tabla 9: Características de la bomba de condensado principal. Fuente: [1], página 263

6. DESCRIPCIÓN DE LA PLANTA PROPULSORA DE CICLO COMBINADO

Como segunda opción de cálculo, el programa a desarrollar contará con la posibilidad de calcular los parámetros relativos a una planta propulsora de ciclo combinado, además de la ya expuesta primera opción consistente en una planta propulsora de vapor convencional, cuyos elementos constitutivos ya han sido descritos en el apartado anterior.

La planta propulsora de ciclo combinado, en lo referente a la parte de la planta operada por vapor estará constituida por los mismos elementos y con la misma estructura de la planta de vapor convencional ya descrita, con la única diferencia de que la caldera de vapor empleada será una caldera de recuperación destinada al aprovechamiento de los gases de escape de una turbina de gas General-Electric LM 2500 [13].

Además, el sistema de propulsión será de tipo eléctrico, careciendo en este caso de reductora y de cualquier tipo de accionamiento que transmita el movimiento entre el eje de la turbina y la hélice. En su lugar ambas turbinas

se encontrarán acopladas a sendos generadores eléctricos encargados de producir la energía eléctrica necesaria para la propulsión de buque.

6.1 TURBINA DE GAS

Tal como ya se ha introducido, en el caso de la planta propulsora de ciclo combinado, contaremos con una turbina de gas General-Electric LM 2500 [13]. Se trata de una turbina de tipo aeroderivada con una potencia neta de hasta 25000 kW. A continuación, se muestran en la tabla 10 los parámetros técnicos característicos más significativos de la misma.

Potencia neta	25060 kW
Relación de compresión	17
Velocidad máxima	3600 rpm
Temperatura de escape	566ºC
Caudal másico gases	70,5 kg/s

Tabla 10: Características de la turbina de gas. Fuente: [22]

6.2 CALDERA DE RECUPERACIÓN

La planta de ciclo combinado contará con una caldera de recuperación HRSG de tipo acuotubular fabricada a medida por RENTECH Boiler Systems con una capacidad de aprovechamiento del calor de hasta 40 MW y una temperatura máxima de trabajo de hasta 1000 º Celsius.

7. CÁLCULO

En lo referente a las herramientas software a emplear para el desarrollo de los cálculos que internamente debe realizar la aplicación informática a desarrollar, cabe destacar que el código fuente será íntegramente desarrollado en el lenguaje de programación Java en su versión 8 [12].

Todas las operaciones matemáticas y cálculos necesarios que internamente realice la aplicación harán uso de las funciones que a tal efecto proporciona la clase Math, contenida en la propia API de Java.

En lo referente a la obtención de los parámetros de estado del agua, los valores correspondientes a entropía específica, entalpía específica, volumen específico, temperatura y presión correspondientes a un punto concreto del diagrama podrán obtenerse haciendo uso de la librería IF97 [11], la cual implementa internamente el contenido de las tablas de vapor [10] necesarias para la resolución y cálculo del ciclo Rankine, así como el acceso e interpolación de los datos contenidos en ellas.

Por último, en lo referente a la representación de gráficas y diagramas, se hará uso de la clase Gráfica del paquete Fundamentos [9] desarrollado por Michael González y Mariano Benito Hoz, profesores de la universidad de Cantabria.

8. PLANIFICACIÓN

El desarrollo del software se llevará a cabo en una serie de etapas para las cuales se ha estimado y establecido unos tiempos de realización de acuerdo a lo expuesto en la tabla 11 con la estructura, cronología y plazos que se indican para cada una de las etapas.

		tiempo estimado (horas)
Etapa 1: Estructuración y diseño previo del código		
Tarea 1	Captura de requisitos y casos de uso del programa	8 h
Tarea 2	Elaboración de diagramas de clase del código a desarrollar.	8 h
Tarea 3	Elaboración de diagramas de secuencia y plantillas.	25 h
Tarea 4	Elaboración de mockups	5 h
Etapa 2: Desarrollo del software		
Tarea 1	Desarrollo del módulo de acceso a las tablas de Vapor.	22 h
Tarea 2	Desarrollo del módulo de cálculo del ciclo Brayton	25 h
Tarea 3	Desarrollo del módulo de cálculo del ciclo combinado.	25 h
Tarea 4	Desarrollo del módulo de cálculo del ciclo Rankine.	35 h

Tarea 5	Desarrollo de la pantalla de inicio del programa.	3 h
Tarea 6	Desarrollo de la interfaz de la ventana de la planta de vapor.	20 h
Tarea 7	Desarrollo de la interfaz de la ventana de la planta de ciclo combinado.	25 h
Tarea 8	Desarrollo de la interfaz de la ventana de resultados.	30 h
Tarea 9	Desarrollo de la interfaz de representación gráfica de diagramas	20 h
Etapa 4: Depuración de errores		
Tarea 1	Desarrollo de test y casos de prueba	30 h
Tarea 2	Corrección de errores	30 h
Etapa 5: Implantación		
Redacción de manuales de instrucciones y documentación del programa		10h
Instalación de equipos		2h
Instalación del programa		2h
Formación de los usuarios		5h
Total de horas del proyecto		330 h

Tabla 11: Planificación de tareas del proyecto. Fuente Propia.

9. REFERENCIAS Y BIBLIOGRAFÍA

[1] KNUTSEN, Machinery Operation Manual H2236 (2006)

[2] DNV, Rules for Classification, Part 4,
Chapter 9 Control and monitoring systems (2021)

[3] AENOR, UNE 157001 Criterios generales para la elaboración formal de los documentos que constituyen un proyecto técnico. (2014)

[4] Potter, M, Somerton C; Termodinámica para Ingenieros; Ed MCGrawHill, (2004)

[5] Calos Javier Renedo Estébanez, Termodinámica y mecánica de fluidos, Obtenido de https://personales.unican.es/renedoc/docencia.htm

[6] Agüera Soriano, J; Termodinámica Lógica y Motores Térmicos; Ed Ciencia 3, S.A (1999)

[7] Mataix C. Turbomáquinas Térmicas: Turbinas de Vapor, Turbinas de Gas y Turbocompresores. Editorial Dossat (2000)

[8] International Group of Liquefied Natural Gas Importers, The LNG Industry (2012)

[9] Michael González Harbour, Mariano Benito Hoz, Paquete fundamentos (1999), Obtenido de https://www.istr.unican.es/fundamentos/

[10] Wagner, Wolfgang & Kretzschmar, Hans-Joachim, International Steam Tables -Properties of Water and Steam Based on the Industrial Formulation IAPWS (2008)

[11] Ralph Hummeling, Librería IF97 (2009) Obtenido de https://www.if97.software/

[12] Oracle, Java JDK 8 (2019) Obtenido de https://www.oracle.com/es/java/technologies/javase/javase-jdk8-downloads.html

[13] General Electric, LM2500 aeroderivative gas turbine, Obtenido de https://www.ge.com/gas-power/products/gas-turbines/lm2500

[14] Wikipedia, Ciclo e Carnot. Obtenido de https://es.wikipedia.org/wiki/Ciclo_de_Carnot

[15] Universidad Politécnica de Valencia, El ciclo de Rankine de vapor de agua. Obtenido de https://www.youtube.com/watch?v=cVTAL-CV9pk

[16] Universidad Politécnica de Valencia, Introducción al ciclo de vapor. Obtenido de https://www.youtube.com/watch?v=r92QF2eb1-g

[17] Universidad Politécnica de Valencia. Ciclo de Rankine con recalentamiento. Obtenido de https://www.youtube.com/watch?v=FWqpeNlBRkg&t=90s

[18] Universidad Politécnica de Valencia, Ciclo de Rankine con regeneración. https://www.youtube.com/watch?v=aTwFQB4Zx8Y&t=765s

[19] Universidad Politécnica de Valencia, El ciclo de Brayton de turbina de gas. Obtenido de https://www.youtube.com/watch?v=XFNH32J5mpI

[20] Hugo Fernández, Calentadores abiertos. Obtenido de https://sites.google.com/site/201508hugofernandez/calentadores-abiertos

[21] Diagrama esquemático de generación de electricidad por ciclo combinado. Obtenido de https://jmirez.wordpress.com/2011/02/16/j192-diagrama-esquematico-de-generacion-de-electricidad-por-ciclo-combinado/

[22] Direct Industry, LM2500. Obtenido de https://www.directindustry.es/prod/ge-gas-turbines/product-34155-1731000.html

ANEXO: CÁLCULOS

En el presente apartado detallaremos el procedimiento de cálculo a seguir para el cálculo de los distintos valores que el programa deberá obtener en relación con el funcionamiento de las plantas propulsoras. Para ello deberán aplicarse las fórmulas que aquí se exponen.

1. CÁLCULO DE LA PLANTA DE VAPOR

En primer lugar, a continuación, se explicarán los procesos de cálculo a seguir para la planta propulsora de vapor, siendo estos de común aplicación para la planta de ciclo combinado en cuanto a la parte de la planta consistente en un ciclo de vapor.

Para la exposición del proceso de cálculo de la planta se tomará como referencia el diagrama de la figura 13, el cual ilustra el ciclo de rankine de la planta, así como los valores de presiones y temperturas de trabajo que se incican en el apartado “descripción de la planta propulsora de vapor” de la memoria de este proyecto.

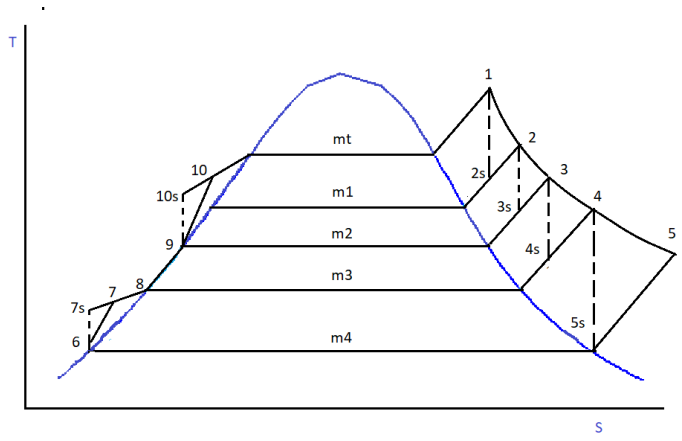


Figura 13: Diagrama de Tempertura-Entropía del ciclo de Rankine de la planta de vapor. Fuente Propia.

Como podemos ver se trata de un ciclo de rankine con regeneración. La primera extracción de vapor, correspondiente al caudal m1 se destina al calentmiento del fuel oil.

La segunda y la tercera extracción correspodientes a los caudales m2 y m3 son destinadas a la regeneración, realizandose el intercambio de calor correspondiente a la tercera extracción mediante un intercambiador de superficie alcanzando el agua de alimetación la entalpía y temperaturas del líquido saturado en el punto 8 a la salida del intercambiador.

La seguda regeneración se lleva a cabo mediante un intecambiador de mezcla encontrandose el agua de alimentación de la caldera a la temperatua y entalpía del punto 9 a la salida de dicho calentador.

Como podemos ver, el circuito cuenta con dos bombas que elevan la presión del agua de alimentación. La primera de ellas es una bomba de alimetación eléctrica que eleva la presión desde el condensador hasta alcanzar la presión de la tercera extracción en el punto 8, a la entrada del intercambiador de superficie.

La segunda de ellas es una turbobomba que eleva la presión del agua desde la presión de la segunda extracción (a a salida del inercambiador de mezcla), hasta alcanzar la presión de la caldera.

1.1 CAUDALES MÁSICOS

Tomando como referencia el diagrama antes expuesto, correspondiente al ciclo de Rankine de la planta, realizaremos el cálculo estimado del caudal de vapor de la extracción m1 en proporción a la cantidad de fuel oíl a calentar con el mismo, de acuerdo a la formula siguiente, en la cual encontramos un término fijo correspondiente al calentamiento del tanque de fueloil y otro término variable correspondiente al caudal de fueloil a calentar y proporcional al mismo en base al índice de carga y la proporción de fueloil en el combustible a quemar (ecuación 14).

$$m1 = m_{CalentamiTanque} + IndiceCarga \cdot PorcentajeFO \cdot m_{CalentamiCaudalFO} \quad (14)$$

En cuanto a los caudales m2, m3 y m4, los calcularemos en base al balance de caudales másico y entalpías en los intercambiadores de regeneración de acuerdo con lo expuesto en las ecuaciones 15, 16 y 17:

$$\mathrm{m2} = (\mathrm{h9} - \mathrm{h8}) \cdot \mathrm{mt}/(\mathrm{h3} - \mathrm{h8}) \quad (15)$$

$$m3 = ((h8 - h7) \cdot mt - (h8 - h7) \cdot m2)/(h4 - h7) \quad (16)$$

$$m4 = mt - m2 - m3 \quad (17)$$

En cuanto al caudal total que atraviesa la caldera, lo obtendremos a partir del caudal total que atraviesa la caldera en la situación de máxima carga aplicándole el porcentaje correspondiente en función del índice de carga actual de la misma, tal como se muestra en la ecuación 18.

$$mt = mtMax \cdot indiceCarga \quad (18)$$

1.2 POTENCIA Y CALOR DE LOS DIFERENTES ELEMENTOS

Para la obtención de la potencia desarrollada por la turbina o aportada por las bombas, así como del calor aportado en la caldera, se hará en base a los saltos entálpicos producidos entre la salida y la entrada del elemento en cuestión y al caudal másico que los atraviesa tal como se indica en las fórmulas 19, 20, 21, 22 y 23.

$$P_{turbina} = (h1 - h2) \cdot mt + (h2 - h3) \cdot (mt - m1) + (h3 - h4) \cdot (mt - m1 - m2) + (h4 - h5) \cdot m4 \quad (19)$$

$$P_{bombaCondensado} = (h7 - h6) \cdot m4 \quad (20)$$

$$P_{bombaAlimentación} = (h10 - h9) \cdot mt \quad (21)$$

$$Q_{caldera} = (h1 - h10) \cdot mt \quad (22)$$

$$Q_{condensador} = (h6 - h5) \cdot m4 \quad (23)$$

En el caso del turbogenerador consideraremos que requiere generar una potencia constante de 2000kw para abastecer todos los equipos y maquinaria auxiliar de la planta que requieran de energía eléctrica. Esto lo haremos así por simplicidad del problema.

En cuanto al caudal de vapor requerido por el turbogenerador lo obtendremos a partir del salto de entalpías entre la entrada y la salida del mismo y de la potencia que hemos establecido (ecuación 24). En este caso y de nuevo, por simplicidad se podrán despreciar rendimientos mecánicos, así como el rendimiento del generador eléctrico.

$$P_{turbogenerador} = m_{turbogenerador} \cdot (h1 - h5) = 2000\ kW \quad (24)$$

En el caso de la turbobomba, la potencia de su turbina de accionamiento se corresponderá con la potencia absorbida por la bomba a la que está acoplada. Por simplicidad en el cálculo no será necesario aplicar rendimientos mecánicos intermedios en el proceso. El caudal que pasará por la turbina de la turbobomba se podrá obtener en base al salto de entalpía en la expansión del vapor y la potencia demandada por la bomba de acuerdo con la siguiente fórmula. (ecuación 25)

$$P_{turbobomba} = mt_{urbobomba} \cdot (h1 - h5) \quad (25)$$

1.3 PARÁMETROS DE FUNCIONAMIENTO DEL CONDENSADOR

El calor cedido por el vapor del condensador, en un caso ideal y bajo la consideración de que no hay pérdidas será igual al calor absorbido por el agua de mar. Es decir, se cumple la expresión siguiente: (ecuaciones 26 y 27)

$$Qv = Qe \tag{26}$$

$$m \cdot (hv - hl) = m \cdot Cp \cdot (\Delta T) \tag{27}$$

Donde la cantidad de calor cedido por el vapor Qv puede expresarse también mediante la expresión siguiente (ecuación 28), en la que A es el área exterior de los tubos, U es el coeficiente total de transferencia de calor de los tubos y LMTD es la temperatura media logarítmica.

$$Qv = A \cdot U \cdot LMT\ D \tag{28}$$

Por otra parte, la temperatura media logarítmica quedará definida de acuerdo a la expresión siguiente (ecuaciones 29 y 30) a partir de la temperatura de entrada y la temperatura de salida del agua del condensador, así como la temperatura del vapor.

$$LMTD = \frac{\Delta T_A - \Delta T_B}{ln\frac{\Delta T_A}{\Delta T_B}} \tag{29}$$

$$LMTD = \frac{T_s - T_e}{ln\frac{T_c - T_e}{T_c - T_s}} \tag{30}$$

1.4 RENDIMIENTO ISOENTRÓPICO

A fin de poder tener en cuenta las irreversibilidades que presentan los procesos de expansión y compresión del fluido que idealmente serían isoentrópicos introduciremos un valor para el rendimiento isoentrópico que nos permita considerar esas irreversibilidades en los cálculos llevados a cabo por el programa.

El valor del rendimiento isoentrópico se entenderá como una relación entre los saltos de entalpía ideal y real y se aplicará de acuerdo a las ecuaciones 31 y 32.

En el caso de las bombas:

$$\eta_s = \frac{hs_{final} - h_{inicial}}{h_{final} - h_{inicial}} \quad (31)$$

En el caso de la turbina:

$$\eta_s = \frac{h_{inicial} - h_{final}}{hs_{final} - h_{inicial}} \quad (32)$$

1.5 RENDIMIENTO MECÁNICO

A la hora de aplicar los rendimientos mecánicos de las bombas y la turbina en el cálculo de la planta se hará en base a las ecuaciones 33 y 34, las cuales relacionan la potencia térmica con la potencia mecánica en el eje.

En el caso de la turbina:

$$\eta_{\mathrm{m}} = \frac{Potencia\ \ en\ el\ eje}{m(h_{inicial} - h_{final})} \tag{33}$$

En el caso de las bombas:

$$\eta_{\mathrm{m}} = \frac{m(h_{final} - h_{inicial})}{Potencia\ en\ el\ eje} \tag{34}$$

1.6 RENDIMIENTO DE LA CALDERA

El rendimiento de la caldera encargada de generar el vapor se aplicará en base a la fórmula que se indica a continuación (ecuación 35), la cual relaciona el calor aportado al fluido con el calor total producido en la combustión.

$$\eta_{\mathrm{caldera}} = \frac{m(h_{final} - h_{inicial})}{Qcombustión} \tag{35}$$

1.7 CONSUMO DE COMBUSTIBLE

La cantidad de combustible se determinará a partir del calor necesario que deba producirse en la combustión. Para ello tendremos en cuanta el PCI (poder calorífico inferior) de cada uno de los combustibles (ecuación 36), así como el porcentaje en el que se consume cada uno. (ecuaciones 37 y 38)

$$Qcombustión = m_{HFO} \cdot PCI_{\ HFO} + m_{LNG} \cdot PCI_{LNG} \quad (36)$$

$$\%_{HFO} = \frac{m_{HFO}}{m_{HFO}+m_{LNG}} \quad (37)$$

$$\%_{LNG} = \frac{m_{LNG}}{m_{HFO}+m_{LNG}} \quad (38)$$

1.5 RENDIMIENTO NETO DE LA PLANTA

El rendimiento neto de la planta lo entenderemos como la relación entre la potencia útil entregada por la turbina propulsora en el eje y la energía total liberada en forma de calor en la combustión del combustible. (ecuación 39)

Tal como hemos podido ver en los apartados anteriores, entre la combustión y la entrega final de potencia en el eje se producen una serie de pérdidas que engloban rendimiento de la combustión, rendimiento de la caldera, rendimientos mecánicos, rendimiento térmico de la turbina, irreversibilidades termodinámicas, así como el consumo de parte del vapor en elementos auxiliares necesarios para el funcionamiento de la planta como son el turbogenerador o la turbobomba de alimentación.

De esta forma, expresaremos el rendimiento neto de la planta de acuerdo a la siguiente expresión (ecuación 39):

$$\eta_{\text{planta}} = \frac{P_{propulsión}}{Q_{combustión}} \quad (39)$$

2. PLANTA DE CICLO COMBINADO

En lo referente a la segunda planta propulsora contemplada por el programa informático, la planta de ciclo combinado, el procedimiento de cálculo y ecuaciones indicadas para la planta propulsora de vapor serán también de aplicación en esta en lo que al ciclo de vapor se refiere, dado que este es de idénticas características en ambas plantas propulsoras tal como se ve en el diagrama del ciclo Rankine contenido en la figura 14.

Por otra parte, los aspectos referentes al ciclo de gas exclusivos únicamente de la planta propulsora de ciclo combinado, se expondrán a continuación junto con las ecuaciones y fórmulas necesarias para el cálculo de todos sus parámetros de funcionamiento. El ciclo de gas seguido por la planta será el indicado en la figura 14, y se tomará el mismo como referencia para el cálculo de potencias y rendimientos, así como lo indicado en el apartado destinado a la descripción de la planta en la memoria de este proyecto.

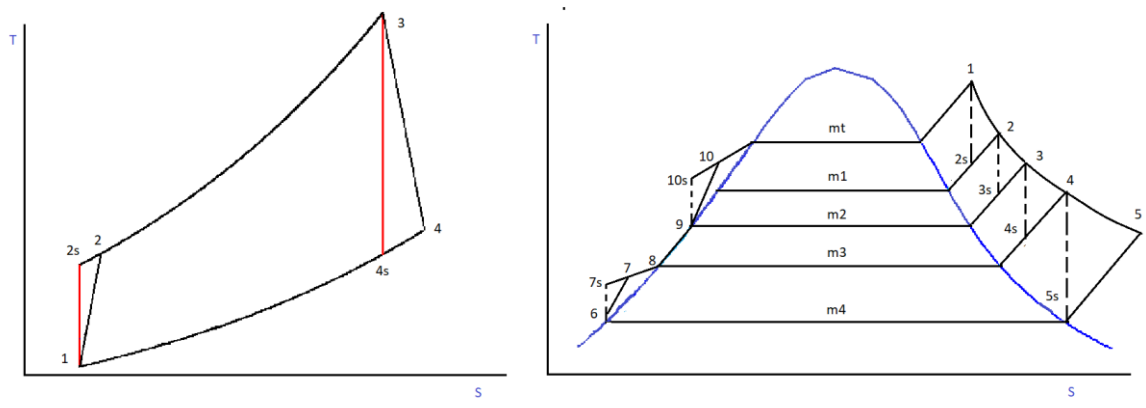


Figura 14: Diagramas de Temperatura-Entropía del ciclo Brayton y ciclo Rankine de la planta de ciclo combinado. Fuente Propia.

2.1 POTENCIA Y CALOR DE LOS DIFERENTES ELEMENTOS

En cuanto al cálculo de las potencias del compresor y de la turbina de gas, y en base a las trasformaciones termodinámicas indicadas en el diagrama de la figura 14, aplicaremos las siguientes fórmulas propias de las transformaciones adiabáticas para el cálculo del trabajo y la potencia desarrollados por el gas o aportados al mismo durante su expansión o compresión (ecuaciones 40 y 41). En dicha ecuación el término m hace referencia al caudal másico, Cp al coeficiente de transmisión de calor a presión constante y T a la temperatura.

$$P_{turbina} = m \cdot Cp \cdot (T3 - T4) \quad (40)$$

$$P_{compresor} = m \cdot Cp \cdot (T2 - T1) \quad (41)$$

Por otra parte, en lo referente al cálculo del calor aportado al ciclo y cedido por este al foco frío, procesos que se realizan en el ciclo Brayton a presión contante, las fórmulas de aplicación para el cálculo de dicho calor serán las siguientes. (ecuaciones 42 y 43)

$$Q_{combustión} = m \cdot Cp \cdot (T3 - T4) \quad (42)$$

$$Q_{gases\ de\ escape} = m \cdot Cp \cdot (T2 - T1) \quad (43)$$

2.2 RENDIMIENTO ISOENTRÓPICO

A fin de poder tener en cuenta las irreversibilidades que presentan los procesos de expansión y compresión del gas que idealmente serían isoentrópicos introduciremos un valor para el rendimiento isoentrópico que

nos permita considerar esas irreversibilidades en los cálculos llevados a cabo por el programa.

El valor del rendimiento isoentrópico se entenderá como una relación entre los saltos de temperatura ideal y real y se aplicará de acuerdo a las fórmulas 44 y 45.

En el caso del compresor:

$$\eta_s = \frac{Ts_{final} - T_{inicial}}{T_{final} - T_{inicial}} \quad (44)$$

En el caso de la turbina:

$$\eta_s = \frac{T - T_{final}}{Ts_{final} - T_{inicial}} \quad (45)$$

2.3 RENDIMIENTO MECÁNICO

A la hora de aplicar los rendimientos mecánicos del compresor y la turbina en el cálculo de la planta se hará en base a las fórmulas siguientes, las cuales relacionan la potencia térmica con la potencia mecánica en el eje (ecuaciones 46 y 47).

En el caso de la turbina:

$$\eta_m = \frac{Potencia\ en\ el\ eje}{m \cdot Cp \cdot (T_{inicial} - T_{final})} \quad (46)$$

En el caso del compresor:

$$\eta_{\mathrm{m}} = \frac{m \cdot Cp \cdot (T_{final} - T_{inicial})}{Potencia\ en\ el\ eje} \tag{47}$$

2.4 RENDIMIENTO DE LA CALDERA DE RECUPERACIÓN

El rendimiento de la caldera de recuperación encargada de generar el vapor empleando el calor residual de los gases de escape de la turbina de gas se aplicará en base a la ecuación 48, la cual relaciona el calor aportado al fluido con el calor total contenido en los gases de escape que se introducen en la caldera recuperación.

$$\eta_{\mathrm{caldera}} = \frac{m(h_{final} - h_{inicial})}{Q_{gases\ de\ escape}} \tag{48}$$

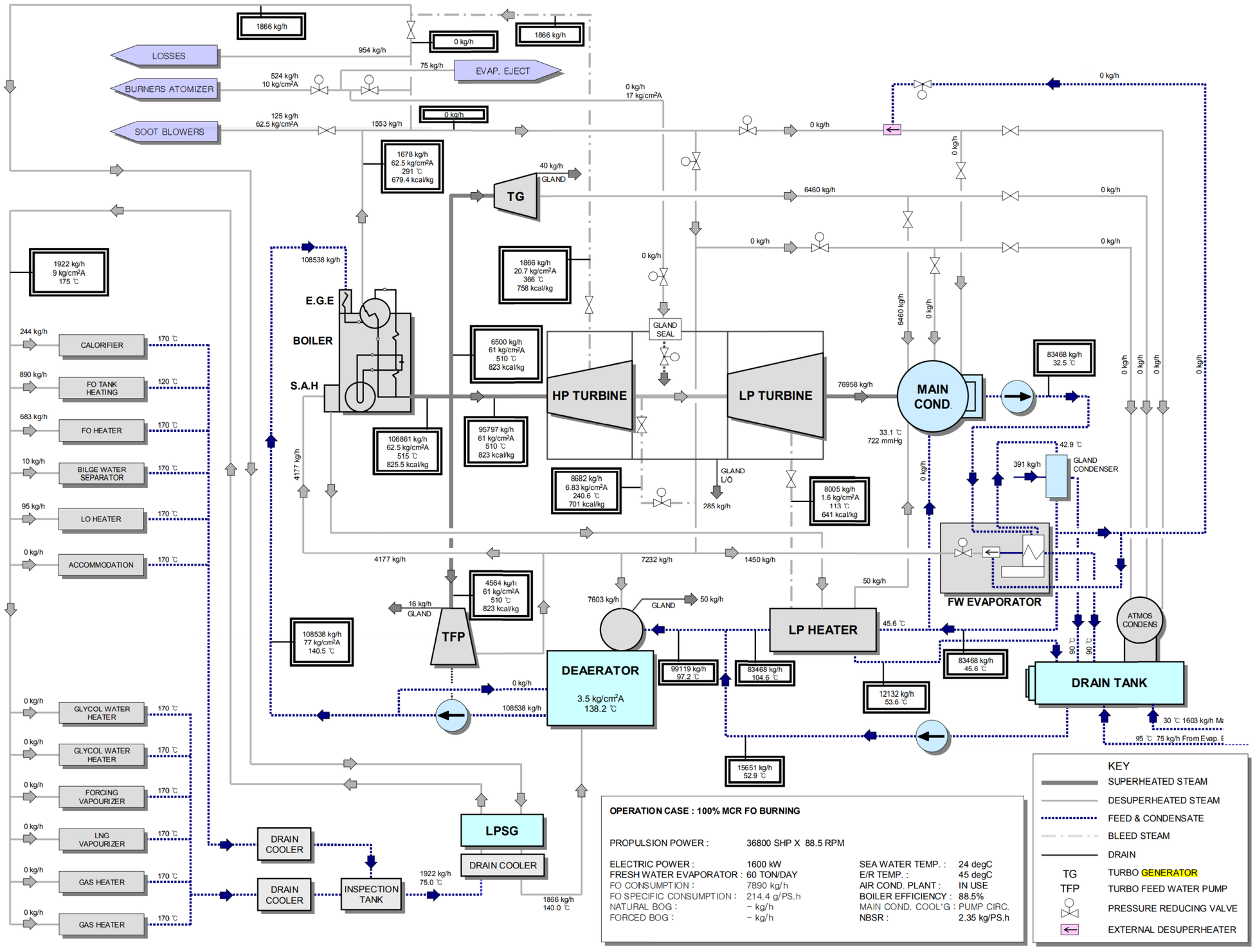
2.5 RENDIMIENTO NETO DE LA PLANTA

El rendimiento neto de la planta lo entenderemos como la relación entre la potencia útil total entregada por ambas turbinas y la energía total liberada en forma de calor en la combustión del gas Natural, de acuerdo con la expresión siguiente. (ecuación 49)

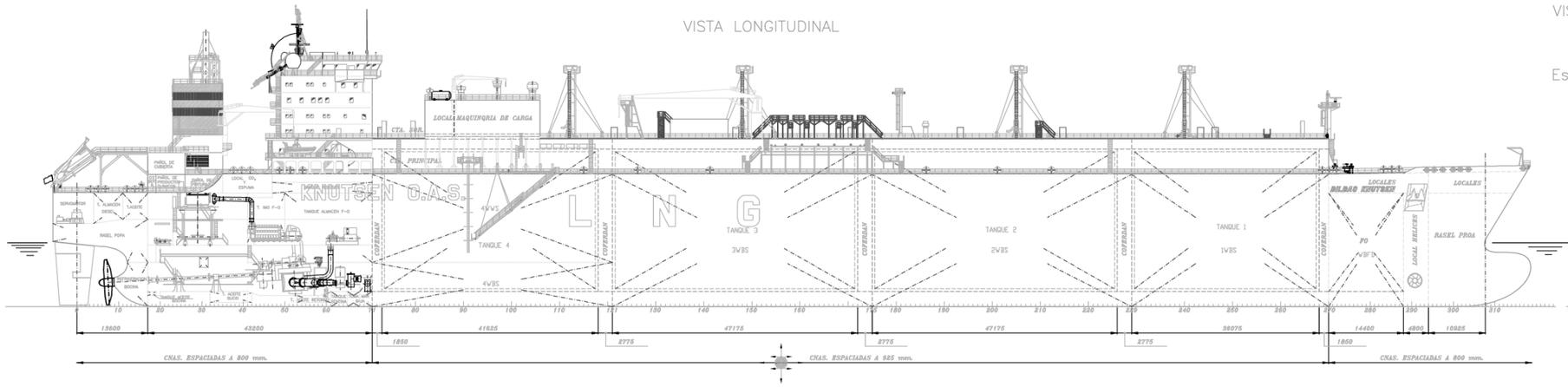
$$\eta_{\mathrm{planta}} = \frac{P_{propulsora\ total}}{Q_{combustión}} \tag{49}$$

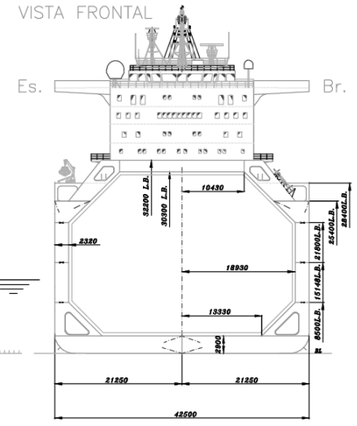
ANEXO: PLANOS

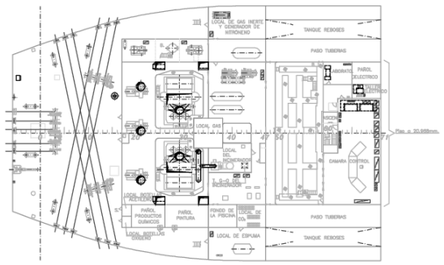
7.1 100% MCR FO Burning Condition

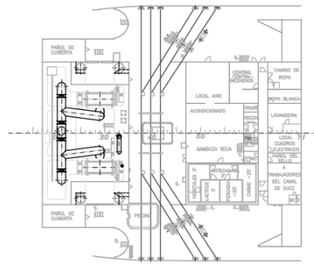


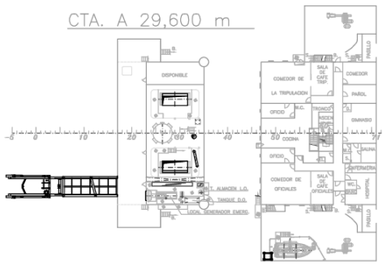
Plano 1. Diagrama del balance térmico de la planta de vapor.

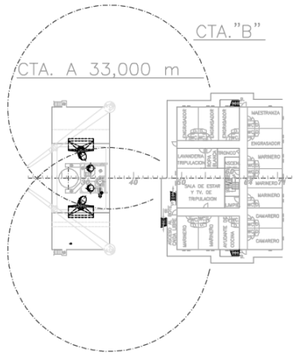


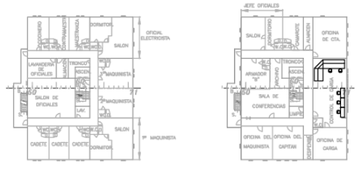


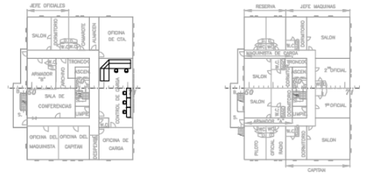


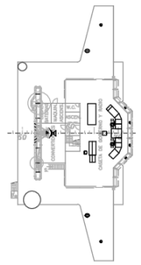


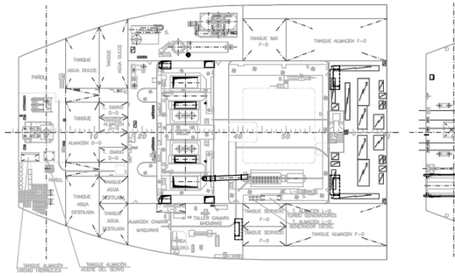


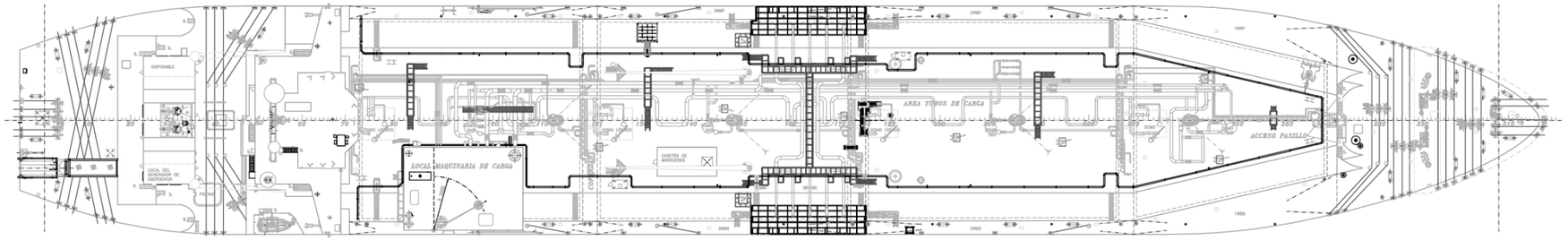


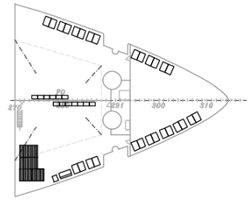


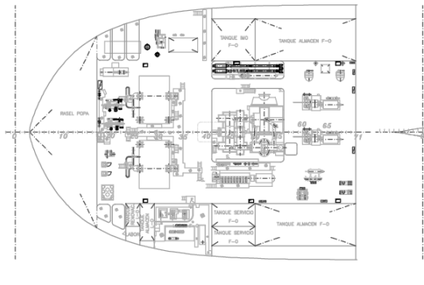


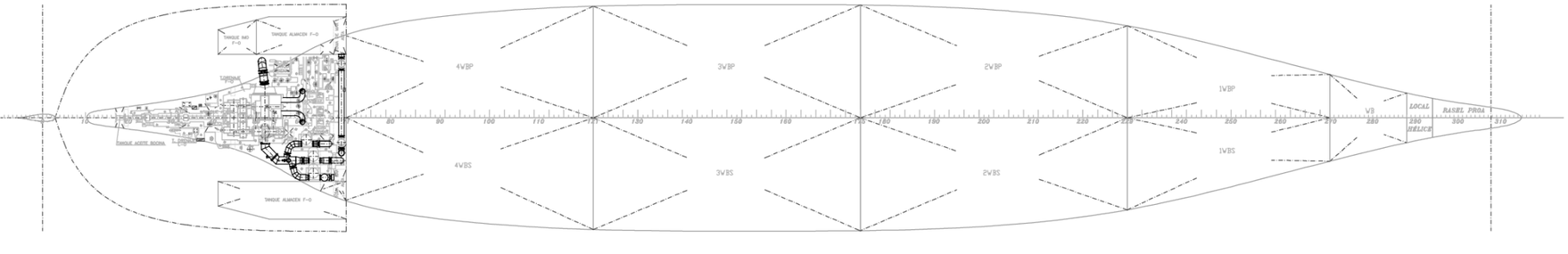












DIMENSIONES PRINCIPALES

ESLORA TOTAL	284,379M
ESLORA ENTRE PERPENDICULARES	271,000M
MANGA DE TRAZADO	42,500M
PUNTAL CTA. PRINCIPAL	25,400M
CALADO DE DISEÑO	11,400M

MODIFICACIONES / MODIFICATIONS

SOCIEDAD DE CLASIFICACION Y ARMADOR / CLASSIFICATION SOCIETY AND OWNER

IZAR

Astillero Sestao
Departamento Tecnico

DISPOSICION GENERAL

Escala/Scale	Fecha/Date	N° de Plano/Drawing Number	Revision
1/300	J.C. Pinedo 16-12-03	321.2.14.V0000.013	
	Comp./Check J.Urbieta 16-12-03	Hoja/Sheet	N° Hojas/Sheets
	Aprob./Approved A. Barbastro 16-12-03	21-13	1/1

Plano 2. Plano de la disposición general del buque.

ANEXO: EL SOFTWARE DESAROLLADO

En el presente apartado, se presenta una demostración del funcionamiento del programa de cálculo una vez desarrollado y construido de acuerdo a lo especificado en el proyecto. A modo de demostración del trabajo realizado se muestran algunos casos prácticos de configuración y ejecución del mismo ilustrado mediante capturas de pantalla de la interfaz del programa en las que se puede ver el proceso seguido para su manipulación y utilización por parte del usuario.

1. SELECCIÓN DEL MODO DE FUNCIONAMIENO

Según se inicia el programa se muestra la ventana de la figura 15, en la cual se da la opción al usuario de seleccionar el tipo de planta de vapor sobre la que se desean realizar los cálculos de sus parámetros operativos y de funcionamiento. Las opciones son dos, por un lado, la planta de vapor y por otro la planta de ciclo combinado. La selección del tipo de planta se realiza pulsando en el botón correspondiente.

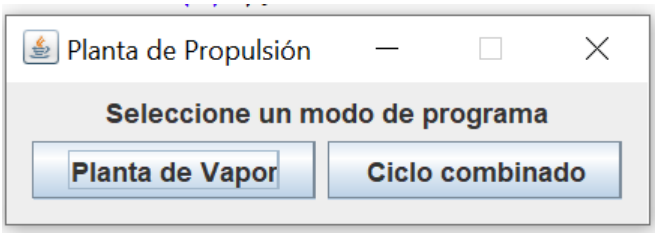


Figura 15: Ventana de inicio del programa. Fuente Propia.

2. MODO DE FUNCIONAMIETO BASADO EN PLANTA DE VAPOR

Cuando el usuario selecciona la opción “planta de vapor” la ventana de inicio se cierra y automáticamente se abre la interfaz correspondiente al modo de cálculo basado en la planta de vapor tal como se muestra en la figura 16.

2.1 CONFIGURACIÓN DE PARÁMETROS DE LA PLANTA

Según podemos ver en la figura 16, la región izquierda de la pantalla consta de una serie de campos donde el usuario puede parametrizar todas las propiedades y características de la planta necesarias para que el programa realice el cálculo de la misma.

Junto a estos campos de configuración se encuentran en la parte inferior una serie de botones para dar al usuario la opción de iniciar el cálculo o de obtener y representar los diagramas entalpía entropía y temperatura entropía correspondientes al ciclo de Rankine seguido por la planta de vapor configurada por el usuario.

Por otra parte, en el lado derecho de la interfaz se muestra un esquema simplificado de la estructura de la planta propulsora.

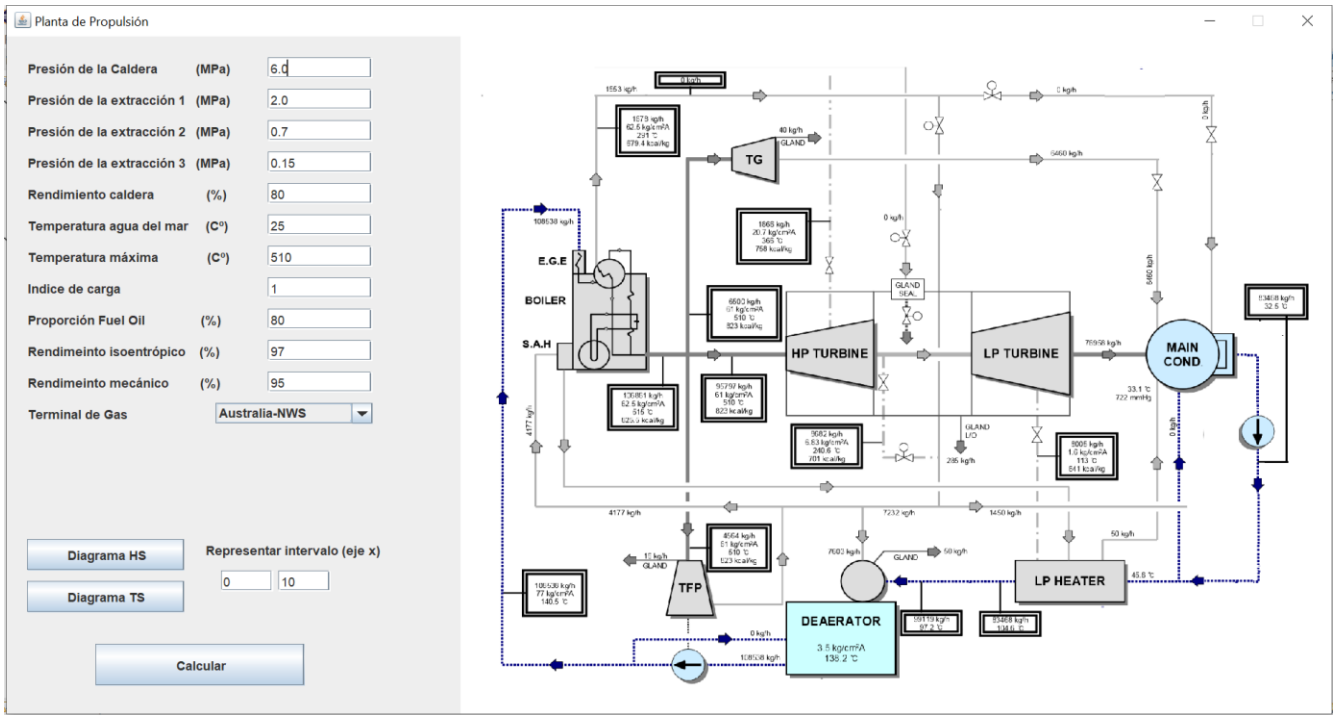


Figura 16: Pantalla de configuración del programa de cálculo en el modo de planta de vapor. Fuente Propia.

Cabe destacar que todos los parámetros de configuración son introducidos por el usuario a través del teclado, a excepción del referente a la terminal de procedencia del gas [8], la cual de be seleccionarse mediante un comboBox de varias opciones (figura 17). Este parámetro será utilizado por el programa para determinar el poder calorífico del gas natural empleado.

Figura 17: Combo box para la selección de la terminal de origen del gas. Fuente Propia.

2.2 REPRESENTACIÓN DE DIAGRAMAS

En lo referente a la representación de diagramas, el usuario dispone de dos opciones en función del botón que pulse, pudiendo optar por representar el diagrama entalpía-entropía o el diagrama temperatura-entropía.

Dichos diagramas se corresponden fielmente al ciclo de Rankine seguido por la planta de vapor en las condiciones que han sido parametrizadas por el usuario, pudiendo incluso seleccionare un rango concreto y acotado del eje x en el cual centrar la representación del diagrama. En las figuras 18 y 19 se muestran los diagramas generados por el programa para el caso de configuración de la planta mostrado anteriormente.

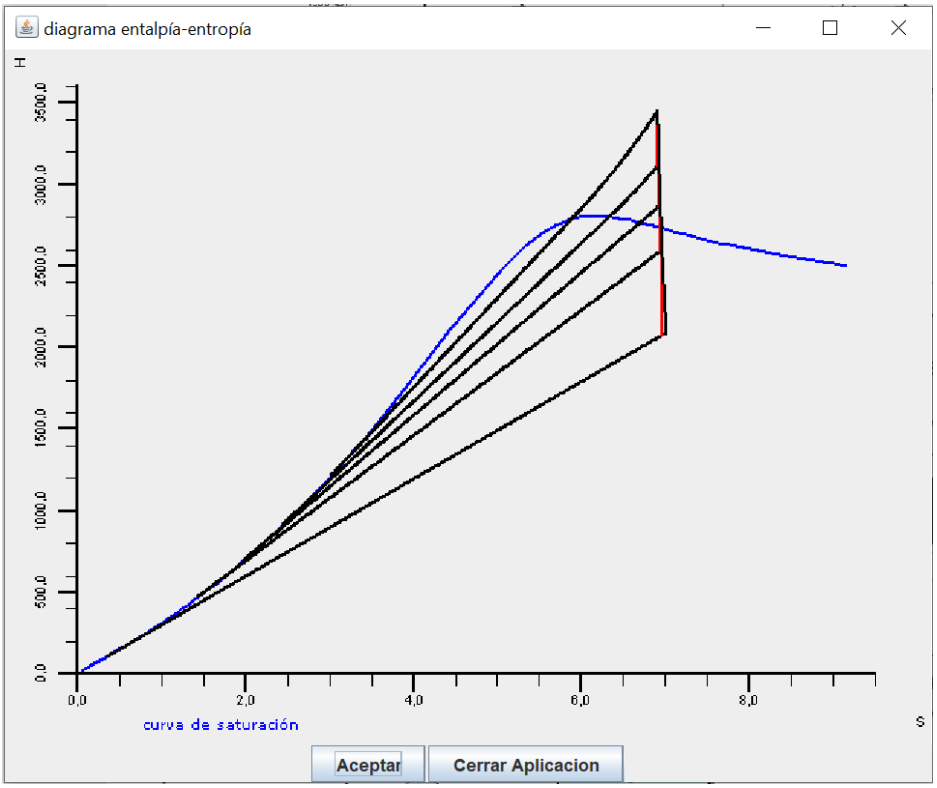


Figura 18: Diagrama de Entalpía-Entropía. Fuente Propia.

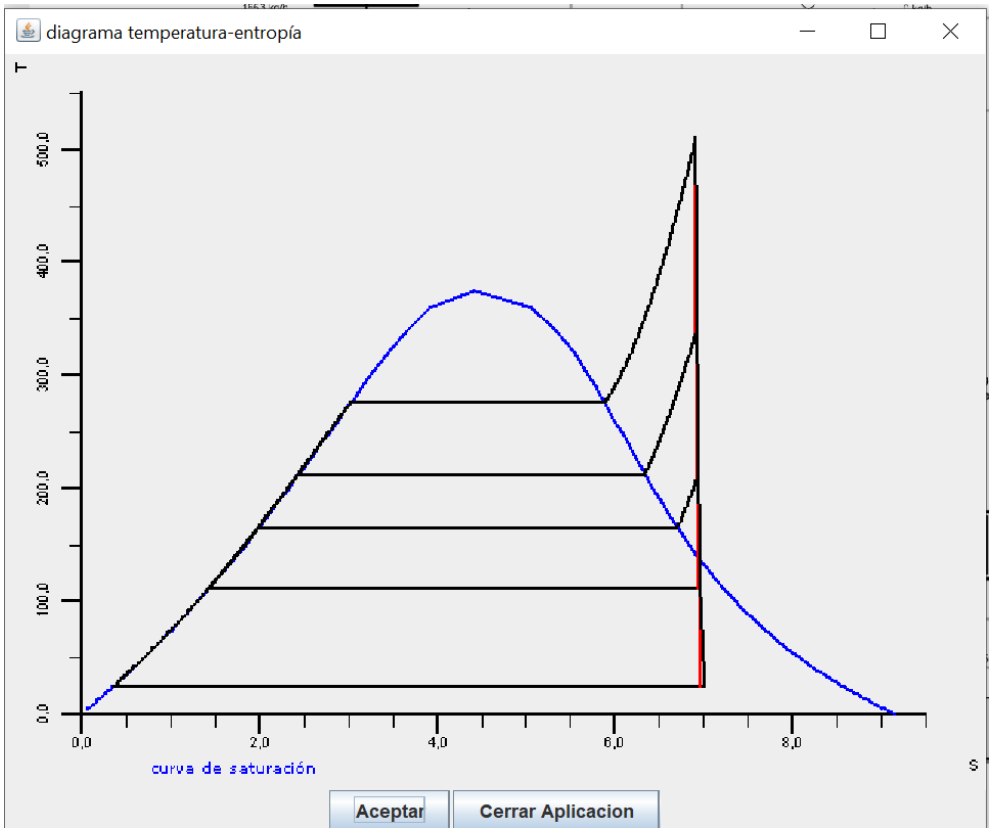


Figura 19: Diagrama de Temperatura-Entropía. Fuente Propia.

2.3 PANTALLA DE RESULTADOS

Una vez que el usuario de la aplicación pulsa sobre e botón de calcular se abre a continuación la ventana mostrada en la figura 20, en la cual se pueden ver todos los resultados de los cálculos realizados por el programa.

Como podemos ver, entre los datos arrojados por el programa se encuentran los valores característicos de presión, temperatura entropía, entalpía y volumen específico de cada punto del ciclo Rankine descrito por la planta.

De la misma forma, se incluyen a continuación los valores a todos los caudales másicos del ciclo, rendimientos y potencias de los diferentes elementos de la planta, así como de la planta propulsora en su conjunto.

Por último, en la fracción inferior de la ventana se muestra de forma esquemática el ciclo de Rankine descrito por la planta a modo de leyenda identificativa de los diferentes puntos del ciclo y de forma que ayude a la interpretación de los datos y resultados arrojados por el programa.

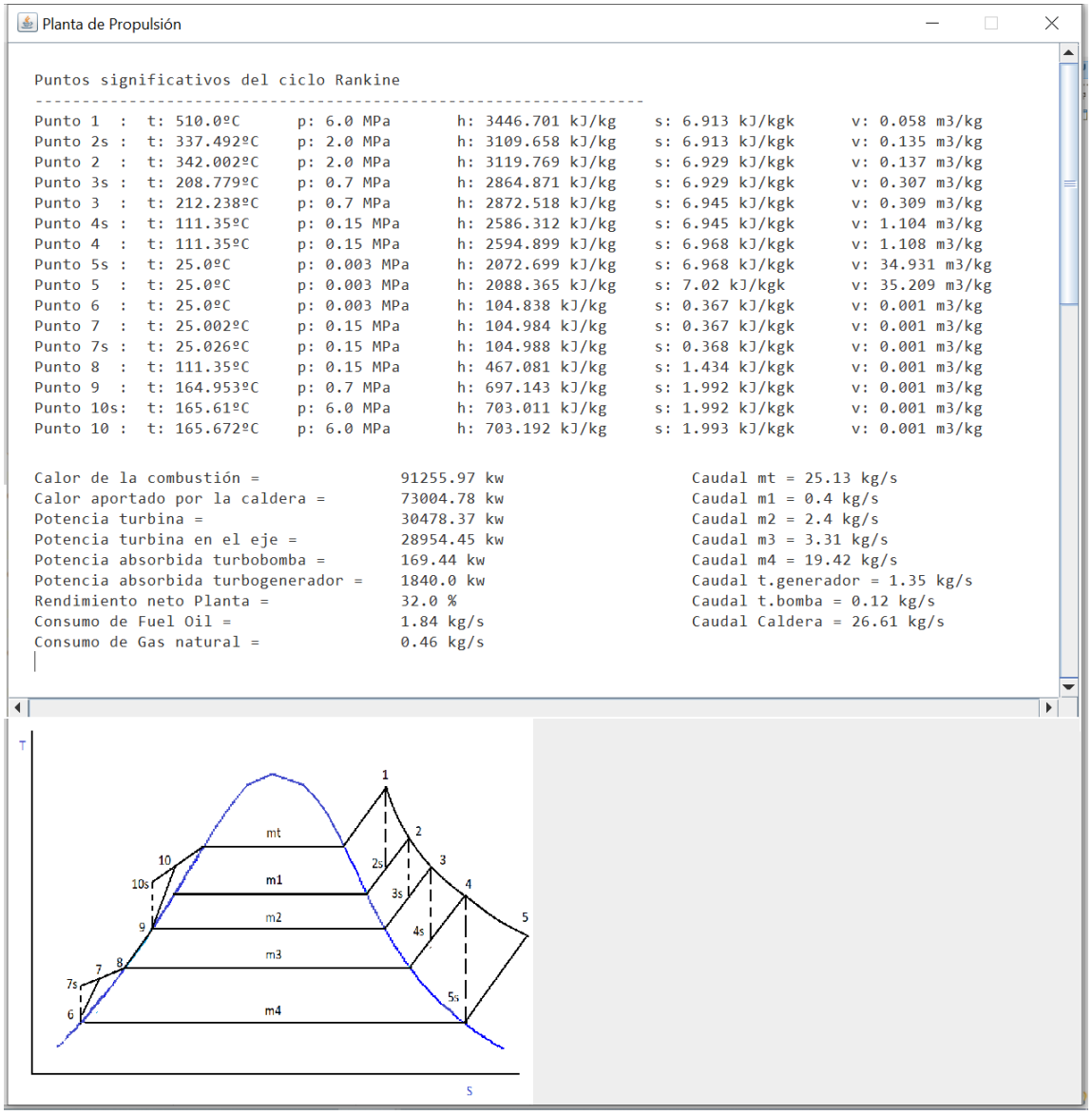


Figura 20: Pantalla de resultados del programa cuando se ejecuta en el modo de planta propulsora de vapor. Fuente Propia.

3. MODO DE FUNCIONAMIENTO BASADO EN PLANTA DE CICLO COMBINADO

Como segunda opción de funcionamiento ofertada por el programa nos encontramos con la posibilidad de hacer los cálculos correspondientes basándonos en el modelo de planta de ciclo combinado. A continuación, analizaremos el funcionamiento de dicha opción haciendo especial énfasis en aquellas funciones y opciones en las cuales difiere de la planta del modo de funcionamiento basado en la planta de vapor ya descrito en el apartado anterior.

3.1 CONFIGURACIÓN DE PARÁMETROS DE LA PLANTA

En la figura 21 podemos ver la pantalla de configuración correspondiente a la planta de ciclo combinado. Tal como podemos observar, la estructura de la ventana es muy similar a la del modo de cálculo basado en la planta de vapor. Como diferencias podemos destacar que ahora el esquema de la planta muestra una turbina de gas acoplada a una caldera de recuperación. Además, algunos parámetros de configuración cambian haciendo ahora por ejemplo referencia al rendimiento de la recuperación, parámetro que antes no existía. Por otra parte, el parámetro correspondiente al porcentaje de fueloil pasa a ser fijo, siendo siempre de un 0% de fuel y un 100% de gas natural.

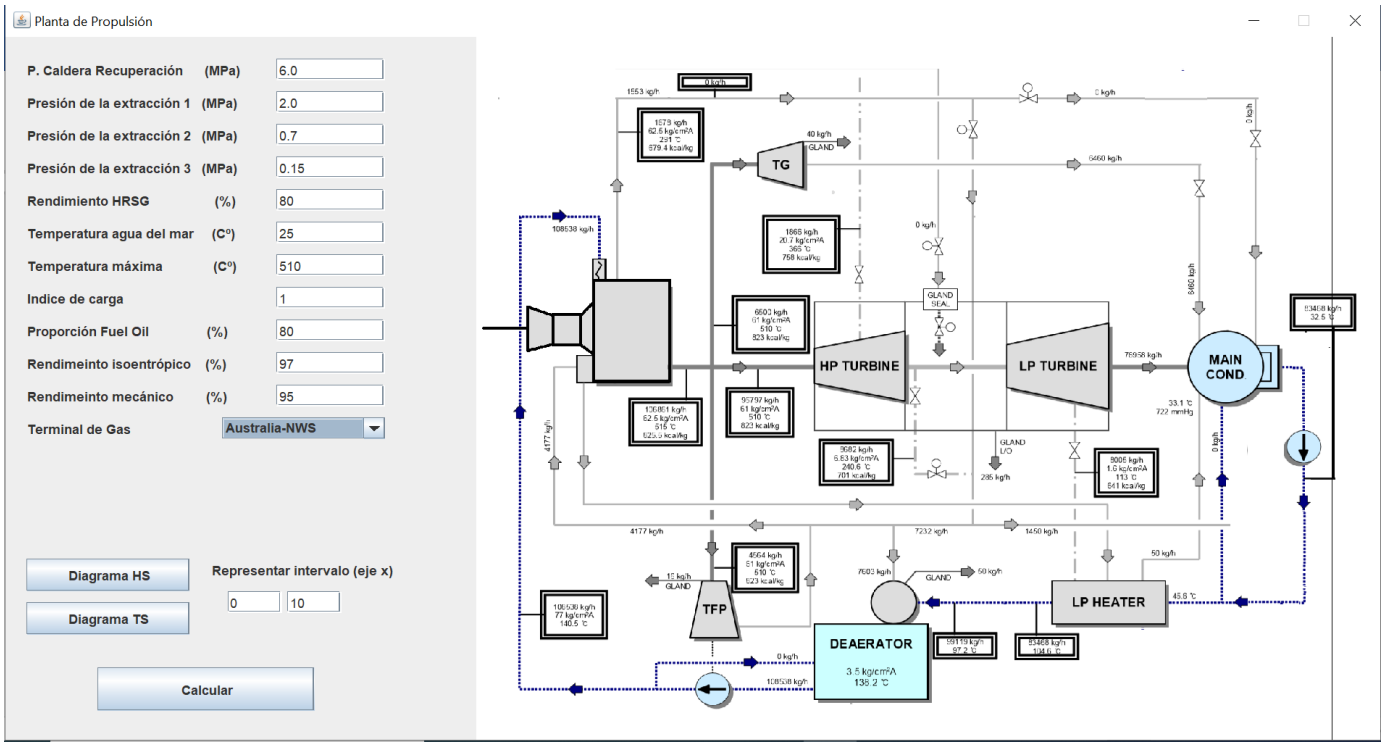


Figura 21: Pantalla de configuración del progama de cálculo en el modo de ciclo combinado. Fuente Propia.

3.2 REPRESENTACIÓN DE DIAGRAMAS

En lo referente a la representación de diagrama cabe destacar que el diagrama de temperatura-entropía se incluye ahora una representación del ciclo Brayton junto al ciclo de Rankine tal como se muestra en la figura 22, siendo ahora relevante la representación de ambos por tratarse de una planta de ciclo combinado.

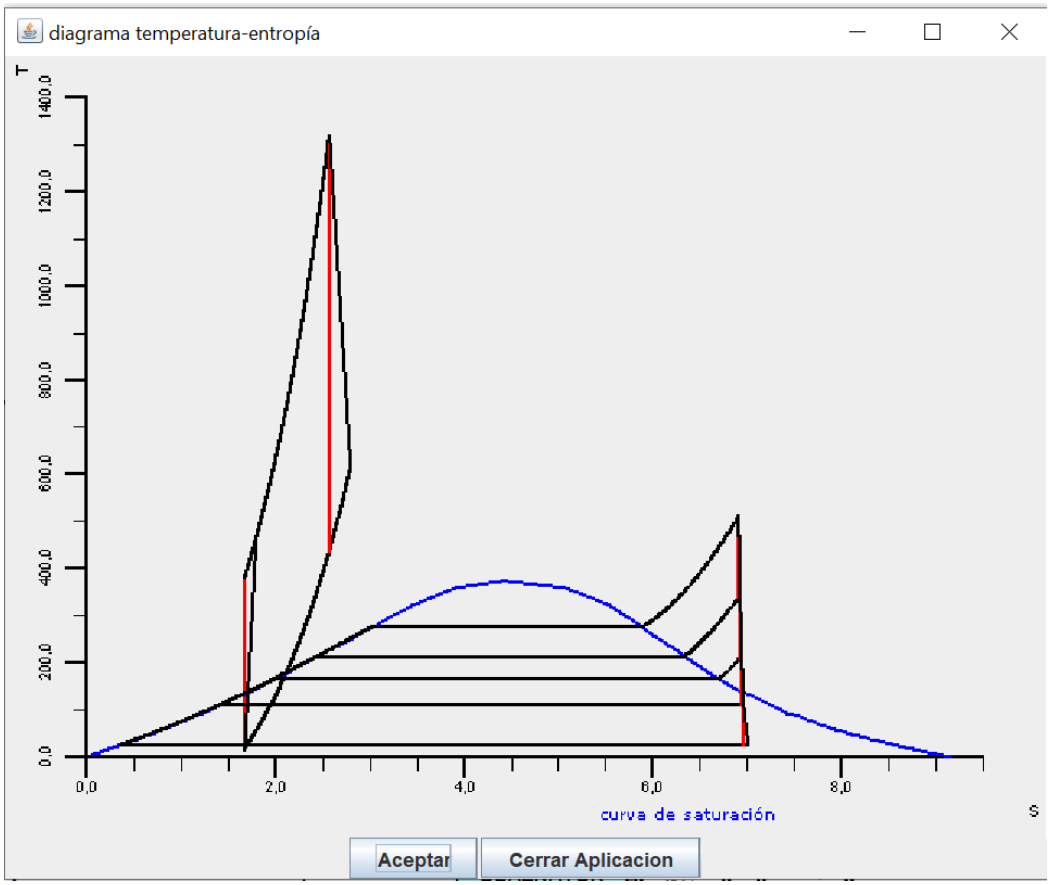


Figura 22: Diagrama de Temperatura-Entropía del ciclo combinado. Fuente Propia.

3.3 PANTALLA DE RESULTADOS

Por último, en lo referente a la pantalla de resultados vemos en la figura 23 que ahora se muestran en primer lugar los valores, puntos característicos, potencias y rendimientos correspondientes al ciclo Brayton y precediendo a los del ciclo de Rankine los cuales se siguen mostrando en este caso de igual forma que se hacía para la planta de vapor convencional.

Los valores de rendimiento y potencia neta mostrados al final del cuadro de texto de la ventana hacen referencia ahora al ciclo combinado en su conjunto.

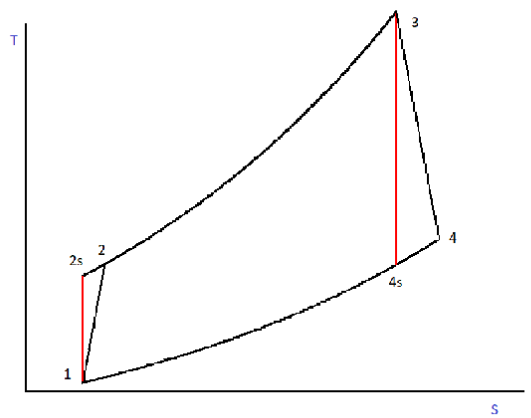
Planta de Propulsión

```
Puntos significativos del ciclo Brayton
-----------------------------------------------------------------
punto1:     t: 14.85 ºC         p: 0.1 MPa
punto2s:    t: 373.917 ºC       p: 1.7 MPa
punto2:     t: 463.684 ºC       p: 1.7 MPa
punto3:     t: 1316.85 ºC       p: 1.7 MPa
punto4s:    t: 434.535 ºC       p: 0.1 MPa
punto4:     t: 610.998 ºC       p: 0.1 MPa

Potencia térmica turbina:     42393.942 kW
Potencia absorbida compresor: 26957.277 kW
Potencia en el eje:           15436.665 kW
Rendimiento:                  30.125 %
Calor cedido al foco frío:    35805.09 kW
Cudal másico:                 59.75 kg/s

Puntos significativos del ciclo Rankine
-----------------------------------------------------------------
Punto 1  :  t: 510.0ºC      p: 6.0 MPa      h: 3446.701 kJ/kg    s: 6.913 kJ/kgk      v: 0.058 m3/kg
Punto 2s :  t: 337.492ºC    p: 2.0 MPa      h: 3109.658 kJ/kg    s: 6.913 kJ/kgk      v: 0.135 m3/kg
Punto 2  :  t: 342.002ºC    p: 2.0 MPa      h: 3119.769 kJ/kg    s: 6.929 kJ/kgk      v: 0.137 m3/kg
Punto 3s :  t: 208.779ºC    p: 0.7 MPa      h: 2864.871 kJ/kg    s: 6.929 kJ/kgk      v: 0.307 m3/kg
Punto 3  :  t: 212.238ºC    p: 0.7 MPa      h: 2872.518 kJ/kg    s: 6.945 kJ/kgk      v: 0.309 m3/kg
Punto 4s :  t: 111.35ºC     p: 0.15 MPa     h: 2586.312 kJ/kg    s: 6.945 kJ/kgk      v: 1.104 m3/kg
Punto 4  :  t: 111.35ºC     p: 0.15 MPa     h: 2594.899 kJ/kg    s: 6.968 kJ/kgk      v: 1.108 m3/kg
Punto 5s :  t: 25.0ºC       p: 0.003 MPa    h: 2072.699 kJ/kg    s: 6.968 kJ/kgk      v: 34.931 m3/kg
Punto 5  :  t: 25.0ºC       p: 0.003 MPa    h: 2088.365 kJ/kg    s: 7.02 kJ/kgk       v: 35.209 m3/kg
Punto 6  :  t: 25.0ºC       p: 0.003 MPa    h: 104.838 kJ/kg     s: 0.367 kJ/kgk      v: 0.001 m3/kg
Punto 7  :  t: 25.002ºC     p: 0.15 MPa     h: 104.984 kJ/kg     s: 0.367 kJ/kgk      v: 0.001 m3/kg
Punto 7s :  t: 25.026ºC     p: 0.15 MPa     h: 104.988 kJ/kg     s: 0.368 kJ/kgk      v: 0.001 m3/kg
Punto 8  :  t: 111.35ºC     p: 0.15 MPa     h: 467.081 kJ/kg     s: 1.434 kJ/kgk      v: 0.001 m3/kg
Punto 9  :  t: 164.953ºC    p: 0.7 MPa      h: 697.143 kJ/kg     s: 1.992 kJ/kgk      v: 0.001 m3/kg
Punto 10s:  t: 165.61ºC     p: 6.0 MPa      h: 703.011 kJ/kg     s: 1.992 kJ/kgk      v: 0.001 m3/kg
Punto 10 :  t: 165.672ºC    p: 6.0 MPa      h: 703.192 kJ/kg     s: 1.993 kJ/kgk      v: 0.001 m3/kg

Calor de la recuperación =              26746.61 kw                 Caudal mt = 6.41 kg/s
Calor aprovechado recuperación =        21397.28 kw                 Caudal m1 = 0.4 kg/s
Potencia en el eje turbina vapor =      7698.21 kw                  Caudal m2 = 0.61 kg/s
Potencia en el eje turbina de gas =     15436.66 kw                 Caudal m3 = 0.84 kg/s
Potencia neta de la planta COGES =      22749.96 kw                 Caudal m4 = 4.95 kg/s
Potencia absorbida turbobomba =         49.66 kw                    Caudal t.generador = 1.35 kg/s
Potencia absorbida turbogenerador =     1840.0 kw                   Caudal t.bomba = 0.04 kg/s
Rendimiento neto Planta COGES=          44.0 %                      Caudal Caldera = 7.8 kg/s
Consumo de Gas natural =                1.28 kg/s
```



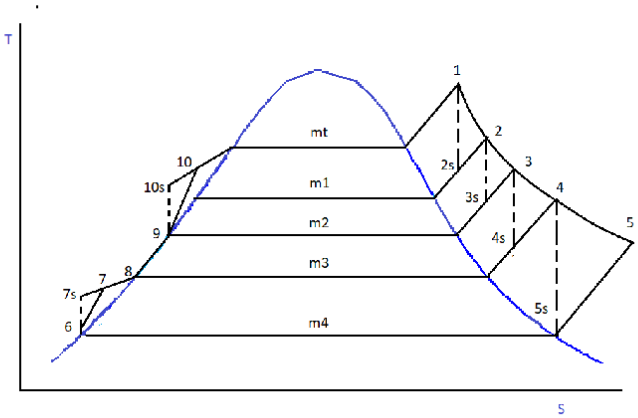


Figura 23: Pantalla de resultados del programa cuando se ejecuta en el modo de ciclo combinado. Fuente Propia.

ANEXO: CÓDIGO FUENTE DEL PROGRAMA DESARROLLADO

```
/**
 * Clase encargada de almacenar los datos y valores del ciclo Brayton y
 * realizar los cálculos del ciclo.
 * @author El autor de este TFG
 *
 */
public class Brayton {

public static final double R=0.2870;
public static final double n=1.4;
public static double Cv=0.7175;
public static double Cp=1.0052;



public static double presionAtmosferica=100000;
public static double temperaturaAmbiente=288;
public static double temperaturaMax=1590;
public static double relacionCopresion=17;
public static double rendimientoIsoentropico=0.8;
public static double rendimientoMecanico=1;

public static PuntoDiagramaBrayton punto1=new PuntoDiagramaBrayton();
public static PuntoDiagramaBrayton punto2=new PuntoDiagramaBrayton();
public static PuntoDiagramaBrayton punto2s=new PuntoDiagramaBrayton();
public static PuntoDiagramaBrayton punto3=new PuntoDiagramaBrayton();
public static PuntoDiagramaBrayton punto4=new PuntoDiagramaBrayton();
public static PuntoDiagramaBrayton punto4s=new PuntoDiagramaBrayton();


static double calorAportado;
static double calorExtraido;
static double trabajoAportado;
```

```
    static double trabajoExtraido;
    static double caudalAire;
    static double calorRecuperar;


    final static double caudalMax=59.75;
    final static double indiceCarga=1;

    static double potenciaTermica;
    static double potenciaEje;
    static double rendimiento;


    /**
     * Método para calcular los puntos significativos del ciclo Brayton
     */
    public static void calcular() {



    punto1.presion=presionAtmosferica;
    punto1.temperatura=temperaturaAmbiente;
    Brayton.punto1.entropia=1.668;
    punto1.imprimePunto();


    punto2s.presion=punto1.presion*relacionCopresion;
    double aux1=Math.pow(punto1.presion, -0.4);
    double aux2=Math.pow(punto1.temperatura, 1.4);
    double aux3=Math.pow(punto2s.presion, -0.4);
    double aux4=(aux1*aux2)/aux3;

    punto2s.temperatura=Math.pow(aux4,(1.0/1.4));
    punto2s.entropia=punto1.entropia;
    punto2s.imprimePunto();


    punto2.presion=punto2s.presion;
    punto2.temperatura=punto1.temperatura+(punto2s.temperatura-
punto1.temperatura)/rendimientoIsoentropico;
```

```
    punto2.entropia=punto2s.entropia+Cp*Math.log(punto2.temperatura/punto2s.temperatura);
    punto2.imprimePunto();

    punto3.temperatura =temperaturaMax;
    punto3.presion=punto2.presion;
    punto3.entropia=punto2s.entropia+Cp*Math.log(punto3.temperatura/punto2s.temperatura);
    punto3.imprimePunto();

    punto4s.presion=punto1.presion;
    aux1=Math.pow(punto3.presion, -0.4);
    aux2=Math.pow(punto3.temperatura, 1.4);
    aux3=Math.pow(punto4s.presion, -0.4);
    aux4=(aux1*aux2)/aux3;
    punto4s.temperatura=Math.pow(aux4,1.0/1.4);
    punto4s.entropia=punto3.entropia;
    punto4s.imprimePunto();

    punto4.presion=punto4s.presion;
    punto4.temperatura=punto3.temperatura-(punto3.temperatura-
punto4s.temperatura)*rendimientoIsoentropico;
    punto4.entropia=punto4s.entropia+Cp*Math.log(punto4.temperatura/punto4s.temperatura);
    punto4.imprimePunto();

    }


    /**
     * Método para calcular las potencias y rendimietos del ciclo Brayton
     */
    public static void calculaPoenciasRendimientos() {
    double t1=Brayton.punto1.temperatura;
    double t2=Brayton.punto2.temperatura;
    double t3=Brayton.punto3.temperatura;
    double t4=Brayton.punto4.temperatura;

    Brayton.trabajoAportado=Cp*(t2-t1);
    Brayton.calorAportado=Cp*(t3-t2);
    Brayton.calorExtraido=Cp*(t4-t1);
    Brayton.trabajoExtraido=Cp*(t3-t4);
```

```
    Brayton.caudalAire=Brayton.caudalMax*indiceCarga;
    Brayton.potenciaTermica=caudalAire*trabajoExtraido;
    potenciaEje=potenciaTermica*Brayton.rendimientoMecanico-
(trabajoAportado*caudalAire)/Brayton.rendimientoMecanico;
    rendimiento=(trabajoExtraido*rendimientoMecanico-
trabajoAportado/rendimientoMecanico)/(calorAportado);
    calorRecuperar=calorExtraido*caudalAire;


    }


    /**
     * Método para calcular el calor aportado al recuperador
     * @param temperaturaSalidaRecuperador
     * @return calor recuperación
     */
    public static double calculaCalorRecuperacion(double temperaturaSalidaRecuperador) {

    return caudalAire*Cp*(Brayton.punto4.temperatura-(temperaturaSalidaRecuperador+273.15));

    }


    /**
     * Método para generar el mensaje descriptivo con la información del ciclo Brayton
     * para poder mostrar en la interfaz del programa.
     * @return texto descriptivo el ciclo Brayton
     */
    public static String generaTexto() {


    String texto="Puntos significativos del ciclo Brayton\n-----------------------------------
----------------------------";
    texto=texto+"\npunto1:     "+punto1.generaTexto();
    texto=texto+"\npunto2s:    "+punto2s.generaTexto();
    texto=texto+"\npunto2:     "+punto2.generaTexto();
    texto=texto+"\npunto3:     "+punto3.generaTexto();
    texto=texto+"\npunto4s:    "+punto4s.generaTexto();
    texto=texto+"\npunto4:     "+punto4.generaTexto()+"\n";
```

```
    texto=texto+"\nPotencia térmica turbina:     "+redondear(potenciaTermica)+" kW";
    texto=texto+"\nPotencia absorbida compresor:
"+redondear(trabajoAportado*caudalAire/rendimientoMecanico)+" kW";
    texto=texto+"\nPotencia en el eje:           "+redondear(potenciaEje)+" kW";
    texto=texto+"\nRendimiento:                  "+redondear(rendimiento*100.0)+ " %";
    texto=texto+"\nCalor cedido al foco frío:    "+redondear(calorRecuperar)+ " kW";
    texto=texto+"\nCudal másico:                 "+redondear(caudalAire)+" kg/s\n\n";

    return texto;
    }

    /**
     * Método auxiliar para rellenar con espacios una cadena de texto
     * @param completar
     * @param longitud
     * @return cadena completa
     */
       public static String completarConEspacios(String completar, int longitud) {

       int inicial=completar.length();
       int anadir=longitud-inicial;
       String retornar=""+completar;

       for(int i=0; i<anadir; i++) {
       retornar=retornar+" ";
       }

    return retornar;

       }

       /**
        * Método auxiliar para redondear un número
        * @param num
        * @return
        */
       public static double redondear(double num) {
       return Math.round(num*1000.0)/1000.0;
       }
```

```
}



/**
 * Clase encargada de la generación y representación de diagramas termodinámicos
 * @author El autor de este TFG
 *
 */
public class GeneraGraficas {

public static Grafica HS;
public static Grafica TS;

public static double min;
public static double max;

/**
 * Método que muestra el diagrama T-S del ciclo Combinado
 */
public static void muestraDiagramaTSCombinado() {
Grafica g = new Grafica ("diagrama temperatura-entropía","S","T");
      double x;
      double y;

      float r=6.0f;

      ////////////////////////////////////////////////////////////////////////7
      ////////////////////////////////////////////////////////////////////////
      ///AQUI COMIENZO A DIBUJAR EL CICLO BRAYTON
      boolean anadido=false;

      g.ponColor(Grafica.negro);
      for(double i=Brayton.punto2s.temperatura; i<(Brayton.punto3.temperatura); i+=1){
        double incrementoEntropia=Brayton.Cp*Math.log(i/Brayton.punto2s.temperatura);
        double entropia=Brayton.punto2s.entropia+incrementoEntropia;
        if(entropia>min&&entropia<max) {
              g.inserta(entropia, i-273.15);
              anadido=true;
```

```
        }
      }

      if(anadido) {
        g.otraGrafica();
      }
      anadido=false;
      for(double i=Brayton.punto1.temperatura; i<(Brayton.punto4.temperatura); i+=1){
        double incrementoEntropia=Brayton.Cp*Math.log(i/Brayton.punto1.temperatura);
        double entropia=Brayton.punto1.entropia+incrementoEntropia;
        if(entropia>min&&entropia<max) {
              g.inserta(entropia, i-273.15);
            anadido=true;
        }
      }

    if(anadido) {
       g.otraGrafica();
       anadido=false;
    }
if(Brayton.punto4s.entropia<max&&Brayton.punto4s.entropia>min) {
      g.ponColor(Grafica.rojo);
      g.inserta(Brayton.punto3.entropia, Brayton.punto3.temperaturaCelsius());
      g.inserta(Brayton.punto4s.entropia, Brayton.punto4s.temperaturaCelsius());
      anadido=true;
}

if(Brayton.punto2s.entropia<max&&Brayton.punto2s.entropia>min) {
      g.otraGrafica();
      g.ponColor(Grafica.rojo);
      g.inserta(Brayton.punto1.entropia, Brayton.punto1.temperaturaCelsius());
      g.inserta(Brayton.punto2s.entropia, Brayton.punto2s.temperaturaCelsius());
      anadido=true;
}

if(Brayton.punto1.entropia>min&&Brayton.punto2.entropia<max) {
      g.otraGrafica();
      g.ponColor(Grafica.negro);
      g.inserta(Brayton.punto1.entropia, Brayton.punto1.temperaturaCelsius());
      g.inserta(Brayton.punto2.entropia, Brayton.punto2.temperaturaCelsius());
```

```
		anadido=true;
	}

	if(Brayton.punto3.entropia>min&&Brayton.punto4.entropia<max) {
		g.otraGrafica();
		g.ponColor(Grafica.negro);
		g.inserta(Brayton.punto3.entropia, Brayton.punto3.temperaturaCelsius());
		g.inserta(Brayton.punto4.entropia, Brayton.punto4.temperaturaCelsius());
		anadido=true;
	}

		////////////////////////////////////////////////////////////////////////7
		////////////////////////////////////////////////////////////////////////
		///AQUI COMIENZO A DIBUJAR EL CICLO RANKINE



		//dibujo la campana de saturación
	if(anadido) {
		g.otraGrafica();
	}
		g.ponColor(Grafica.azul);
		g.ponTitulo("curva de saturación");



		for(int i=0; i<TablaSaturadoTemperatura.listaPuntos.size(); i++) {
		  x=TablaSaturadoTemperatura.listaPuntos.get(i).entropiaLiquido;
		  y=TablaSaturadoTemperatura.listaPuntos.get(i).temperatura;
		  if(x>min&&x<max) {
			g.inserta(x,y);
		  }
		}

		for(int i=TablaSaturadoTemperatura.listaPuntos.size()-1; i>=0; i--) {
		  x=TablaSaturadoTemperatura.listaPuntos.get(i).entropiaVapor;
		  y=TablaSaturadoTemperatura.listaPuntos.get(i).temperatura;
		  if(x>min&&x<max) {
			g.inserta(x,y);
```

```
            }
        }


        //dibujo tramo 8-9


        double temperaturaInicial=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion3).temperatura;
        double temperaturaFinal=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion2).temperatura;
        boolean algunoAnadido=false;



        for(double i=temperaturaInicial; i<temperaturaFinal; i+=0.5) {
          double e=TablaSaturadoTemperatura.getPuntoTemperatura(i).entropiaLiquido;
          if(e<max&&e>min) {
                if(!algunoAnadido) {
                      g.otraGrafica();
                   g.ponColor(Grafica.negro);
                }
                g.inserta(e, i);
                algunoAnadido=true;
          }
        }


        //dibujo las Isóbaras
        if(!(Rankine.punto1.entropia<min)&&!(Rankine.punto10s.entropia>max)) {
        g.otraGrafica();
        g.ponColor(Grafica.negro);
        Transformacion t1=new Transformacion();
        t1.primero=Rankine.punto10s.clone();
        t1.ultimo=Rankine.punto1.clone();
        t1.generaIsobara(500);
        t1.pintaTemperaturaEntropia(g);
        }
```

```
if(!(Rankine.punto2.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionEx
traccion1).entropiaLiquido>max)) {
          g.otraGrafica();
          g.ponColor(Grafica.negro);
          Transformacion t2=new Transformacion();
          PuntoSaturado
ps2=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion1);
          ps2.tituloVapor=0;
          t2.primero=ps2.getPuntoDiagrama();
          t2.ultimo=Rankine.punto2.clone();
          t2.generaIsobara(500);
          t2.pintaTemperaturaEntropia(g);

          }



if(!(Rankine.punto3.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionEx
traccion2).entropiaLiquido>max)) {
          g.otraGrafica();
          g.ponColor(Grafica.negro);
          Transformacion t3=new Transformacion();
          PuntoSaturado
ps3=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion2);
          ps3.tituloVapor=0;
          t3.primero=ps3.getPuntoDiagrama();
          t3.ultimo=Rankine.punto3.clone();
          t3.generaIsobara(500);
          t3.pintaTemperaturaEntropia(g);
          }


          if(!(Rankine.punto4.entropia<min)&&!(Rankine.punto7s.entropia>max)) {
          g.otraGrafica();
          g.ponColor(Grafica.negro);
          Transformacion t4=new Transformacion();
          t4.primero=Rankine.punto7s.clone();
          t4.ultimo=Rankine.punto4.clone();
```

```
            t4.generaIsobara(500);
            t4.pintaTemperaturaEntropia(g);
            }



if(!(Rankine.punto5.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.PresionCo
ndensador).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t5=new Transformacion();
            PuntoSaturado
ps5=TablaSaturadoTemperatura.getPuntoPresion(Rankine.PresionCondensador);
            ps5.tituloVapor=0;
            t5.primero=ps5.getPuntoDiagrama();
            t5.ultimo=Rankine.punto5.clone();
            t5.generaIsobara(500);
            t5.pintaTemperaturaEntropia(g);
            }



            //dibujo las isoentrópicas ideales
            if(Rankine.punto1.entropia>min&&Rankine.punto2s.entropia<max) {
              g.otraGrafica();
              g.ponColor(Grafica.rojo);
            g.inserta(Rankine.punto1.entropia, Rankine.punto1.temperatura);
            g.inserta(Rankine.punto2s.entropia, Rankine.punto2s.temperatura);
            }
            if(Rankine.punto2.entropia>min&&Rankine.punto3s.entropia<max) {
                g.otraGrafica();
              g.ponColor(Grafica.rojo);
            g.inserta(Rankine.punto2.entropia, Rankine.punto2.temperatura);
            g.inserta(Rankine.punto3s.entropia, Rankine.punto3s.temperatura);
            }



            if(Rankine.punto3.entropia>min&&Rankine.punto4s.entropia<max) {
            g.otraGrafica();
            g.ponColor(Grafica.rojo);
```

```
g.inserta(Rankine.punto3.entropia, Rankine.punto3.temperatura);
g.inserta(Rankine.punto4s.entropia, Rankine.punto4s.temperatura);
}


if(Rankine.punto4.entropia>min&&Rankine.punto5s.entropia<max) {
g.otraGrafica();
g.ponColor(Grafica.rojo);
g.inserta(Rankine.punto4.entropia, Rankine.punto4.temperatura);
g.inserta(Rankine.punto5s.entropia, Rankine.punto5s.temperatura);
}




//dibujo las isoentrópicas reales
if(!(Rankine.punto1.entropia>max)&&!(Rankine.punto2.entropia<min)) {
  g.otraGrafica();
  g.ponColor(Grafica.negro);
g.inserta(Rankine.punto1.entropia, Rankine.punto1.temperatura);
g.inserta(Rankine.punto2.entropia, Rankine.punto2.temperatura);
}
if(!(Rankine.punto2.entropia>max)&&!(Rankine.punto3.entropia<min)) {
    g.otraGrafica();
  g.ponColor(Grafica.negro);
g.inserta(Rankine.punto2.entropia, Rankine.punto2.temperatura);
g.inserta(Rankine.punto3.entropia, Rankine.punto3.temperatura);
}


if(!(Rankine.punto3.entropia>max)&&!(Rankine.punto4.entropia<min)) {
g.otraGrafica();
g.ponColor(Grafica.negro);
g.inserta(Rankine.punto3.entropia, Rankine.punto3.temperatura);
g.inserta(Rankine.punto4.entropia, Rankine.punto4.temperatura);
}


if(!(Rankine.punto4.entropia>max)&&!(Rankine.punto5.entropia<min)) {
g.otraGrafica();
```

```
        g.ponColor(Grafica.negro);
        g.inserta(Rankine.punto4.entropia, Rankine.punto4.temperatura);
        g.inserta(Rankine.punto5.entropia, Rankine.punto5.temperatura);
        }


        g.pinta();


}

/**
 * Método que muestra el diagrama T-S del ciclo Rankine
 */
public static void muestraDiagramaTS() {
Grafica g = new Grafica ("diagrama temperatura-entropía","S","T");
        double x;
        double y;

        float r=6.0f;

        //dibujo la campana de saturación
        g.ponColor(Grafica.azul);
        g.ponTitulo("curva de saturación");




        for(int i=0; i<TablaSaturadoTemperatura.listaPuntos.size(); i++) {
          x=TablaSaturadoTemperatura.listaPuntos.get(i).entropiaLiquido;
          y=TablaSaturadoTemperatura.listaPuntos.get(i).temperatura;
          if(x>min&&x<max) {
                g.inserta(x,y);
          }
        }

        for(int i=TablaSaturadoTemperatura.listaPuntos.size()-1; i>=0; i--) {
          x=TablaSaturadoTemperatura.listaPuntos.get(i).entropiaVapor;
          y=TablaSaturadoTemperatura.listaPuntos.get(i).temperatura;
          if(x>min&&x<max) {
```

```
                g.inserta(x,y);
            }
        }


        //dibujo tramo 8-9


        double temperaturaInicial=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion3).temperatura;
        double temperaturaFinal=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion2).temperatura;
        boolean algunoAnadido=false;



        for(double i=temperaturaInicial; i<temperaturaFinal; i+=0.5) {
            double e=TablaSaturadoTemperatura.getPuntoTemperatura(i).entropiaLiquido;
            if(e<max&&e>min) {
                if(!algunoAnadido) {
                    g.otraGrafica();
                g.ponColor(Grafica.negro);
                }
                g.inserta(e, i);
                algunoAnadido=true;
            }
        }



        //dibujo las Isóbaras
        if(!(Rankine.punto1.entropia<min)&&!(Rankine.punto10s.entropia>max)) {
        g.otraGrafica();
        g.ponColor(Grafica.negro);
        Transformacion t1=new Transformacion();
        t1.primero=Rankine.punto10s.clone();
        t1.ultimo=Rankine.punto1.clone();
        t1.generaIsobara(500);
```

```
            t1.pintaTemperaturaEntropia(g);
            }


if(!(Rankine.punto2.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionEx
traccion1).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t2=new Transformacion();
            PuntoSaturado
ps2=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion1);
            ps2.tituloVapor=0;
            t2.primero=ps2.getPuntoDiagrama();
            t2.ultimo=Rankine.punto2.clone();
            t2.generaIsobara(500);
            t2.pintaTemperaturaEntropia(g);

            }



if(!(Rankine.punto3.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionEx
traccion2).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t3=new Transformacion();
            PuntoSaturado
ps3=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion2);
            ps3.tituloVapor=0;
            t3.primero=ps3.getPuntoDiagrama();
            t3.ultimo=Rankine.punto3.clone();
            t3.generaIsobara(500);
            t3.pintaTemperaturaEntropia(g);
            }


            if(!(Rankine.punto4.entropia<min)&&!(Rankine.punto7s.entropia>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t4=new Transformacion();
```

```
            t4.primero=Rankine.punto7s.clone();
            t4.ultimo=Rankine.punto4.clone();
            t4.generaIsobara(500);
            t4.pintaTemperaturaEntropia(g);
            }



if(!(Rankine.punto5.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.PresionCo
ndensador).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t5=new Transformacion();
            PuntoSaturado
ps5=TablaSaturadoTemperatura.getPuntoPresion(Rankine.PresionCondensador);
            ps5.tituloVapor=0;
            t5.primero=ps5.getPuntoDiagrama();
            t5.ultimo=Rankine.punto5.clone();
            t5.generaIsobara(500);
            t5.pintaTemperaturaEntropia(g);
            }


            //dibujo las isoentrópicas ideales
            if(Rankine.punto1.entropia>min&&Rankine.punto2s.entropia<max) {
              g.otraGrafica();
              g.ponColor(Grafica.rojo);
            g.inserta(Rankine.punto1.entropia, Rankine.punto1.temperatura);
            g.inserta(Rankine.punto2s.entropia, Rankine.punto2s.temperatura);
            }
            if(Rankine.punto2.entropia>min&&Rankine.punto3s.entropia<max) {
                g.otraGrafica();
              g.ponColor(Grafica.rojo);
            g.inserta(Rankine.punto2.entropia, Rankine.punto2.temperatura);
            g.inserta(Rankine.punto3s.entropia, Rankine.punto3s.temperatura);
            }


            if(Rankine.punto3.entropia>min&&Rankine.punto4s.entropia<max) {
            g.otraGrafica();
```

```
g.ponColor(Grafica.rojo);
g.inserta(Rankine.punto3.entropia, Rankine.punto3.temperatura);
g.inserta(Rankine.punto4s.entropia, Rankine.punto4s.temperatura);
}


if(Rankine.punto4.entropia>min&&Rankine.punto5s.entropia<max) {
g.otraGrafica();
g.ponColor(Grafica.rojo);
g.inserta(Rankine.punto4.entropia, Rankine.punto4.temperatura);
g.inserta(Rankine.punto5s.entropia, Rankine.punto5s.temperatura);
}



//dibujo las isoentrópicas reales
if(!(Rankine.punto1.entropia>max)&&!(Rankine.punto2.entropia<min)) {
  g.otraGrafica();
  g.ponColor(Grafica.negro);
g.inserta(Rankine.punto1.entropia, Rankine.punto1.temperatura);
g.inserta(Rankine.punto2.entropia, Rankine.punto2.temperatura);
}
if(!(Rankine.punto2.entropia>max)&&!(Rankine.punto3.entropia<min)) {
    g.otraGrafica();
  g.ponColor(Grafica.negro);
g.inserta(Rankine.punto2.entropia, Rankine.punto2.temperatura);
g.inserta(Rankine.punto3.entropia, Rankine.punto3.temperatura);
}


if(!(Rankine.punto3.entropia>max)&&!(Rankine.punto4.entropia<min)) {
g.otraGrafica();
g.ponColor(Grafica.negro);
g.inserta(Rankine.punto3.entropia, Rankine.punto3.temperatura);
g.inserta(Rankine.punto4.entropia, Rankine.punto4.temperatura);
}


if(!(Rankine.punto4.entropia>max)&&!(Rankine.punto5.entropia<min)) {
g.otraGrafica();
```

```
        g.ponColor(Grafica.negro);
        g.inserta(Rankine.punto4.entropia, Rankine.punto4.temperatura);
        g.inserta(Rankine.punto5.entropia, Rankine.punto5.temperatura);
        }


        g.pinta();


}

/**
 * Método que muestra el diagrama H-S del ciclo Combinado
 */
public static void muestraDiagramaHS() {
Grafica g = new Grafica ("diagrama entalpía-entropía","S","H");
        double x;
        double y;

        float r=6.0f;

        //Pinto la campana de saturación
        g.ponColor(Grafica.azul);
        g.ponTitulo("curva de saturación");

        for(int i=0; i<TablaSaturadoTemperatura.listaPuntos.size(); i++) {
          x=TablaSaturadoTemperatura.listaPuntos.get(i).entropiaLiquido;
          y=TablaSaturadoTemperatura.listaPuntos.get(i).entalpiaLiquido;
          if(x>min&&x<max) {
                g.inserta(x,y);
          }
        }

        for(int i=TablaSaturadoTemperatura.listaPuntos.size()-1; i>=0; i--) {
          x=TablaSaturadoTemperatura.listaPuntos.get(i).entropiaVapor;
          y=TablaSaturadoTemperatura.listaPuntos.get(i).entalpiaVapor;
          if(x>min&&x<max) {
                g.inserta(x,y);
          }
        }
```

```
            //Pinto las Isóbaras

if(!(Rankine.punto1.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionCa
ldera).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t1=new Transformacion();
            PuntoSaturado ps1=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionCaldera);
            ps1.tituloVapor=0;
            t1.primero=ps1.getPuntoDiagrama();
            t1.ultimo=Rankine.punto1.clone();
            t1.generaIsobara(500);
            t1.pintaEntalpiaEntropia(g);
            }


if(!(Rankine.punto2.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionEx
traccion1).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t2=new Transformacion();
            PuntoSaturado
ps2=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion1);
            ps2.tituloVapor=0;
            t2.primero=ps2.getPuntoDiagrama();
            t2.ultimo=Rankine.punto2.clone();
            t2.generaIsobara(500);
            t2.pintaEntalpiaEntropia(g);

            }


if(!(Rankine.punto3.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionEx
traccion2).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t3=new Transformacion();
```

```
            PuntoSaturado
ps3=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion2);
            ps3.tituloVapor=0;
            t3.primero=ps3.getPuntoDiagrama();
            t3.ultimo=Rankine.punto3.clone();
            t3.generaIsobara(500);
            t3.pintaEntalpiaEntropia(g);
            }



if(!(Rankine.punto4.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionEx
traccion3).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t4=new Transformacion();
            PuntoSaturado
ps4=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion3);
            ps4.tituloVapor=0;
            t4.primero=ps4.getPuntoDiagrama();
            t4.ultimo=Rankine.punto4.clone();
            t4.generaIsobara(500);
            t4.pintaEntalpiaEntropia(g);
            }



if(!(Rankine.punto5.entropia<min)&&!(TablaSaturadoTemperatura.getPuntoPresion(Rankine.PresionCo
ndensador).entropiaLiquido>max)) {
            g.otraGrafica();
            g.ponColor(Grafica.negro);
            Transformacion t5=new Transformacion();
            PuntoSaturado
ps5=TablaSaturadoTemperatura.getPuntoPresion(Rankine.PresionCondensador);
            ps5.tituloVapor=0;
            t5.primero=ps5.getPuntoDiagrama();
            t5.ultimo=Rankine.punto5.clone();
            t5.generaIsobara(500);
            t5.pintaEntalpiaEntropia(g);
            }
```

```
//pinto las isoentrópicas ideales
if(Rankine.punto1.entropia>min&&Rankine.punto2s.entropia<max) {
   g.otraGrafica();
   g.ponColor(Grafica.rojo);
g.inserta(Rankine.punto1.entropia, Rankine.punto1.entalpia);
g.inserta(Rankine.punto2s.entropia, Rankine.punto2s.entalpia);
}
if(Rankine.punto2.entropia>min&&Rankine.punto3s.entropia<max) {
    g.otraGrafica();
   g.ponColor(Grafica.rojo);
g.inserta(Rankine.punto2.entropia, Rankine.punto2.entalpia);
g.inserta(Rankine.punto3s.entropia, Rankine.punto3s.entalpia);
}


if(Rankine.punto3.entropia>min&&Rankine.punto4s.entropia<max) {
g.otraGrafica();
g.ponColor(Grafica.rojo);
g.inserta(Rankine.punto3.entropia, Rankine.punto3.entalpia);
g.inserta(Rankine.punto4s.entropia, Rankine.punto4s.entalpia);
}


if(Rankine.punto4.entropia>min&&Rankine.punto5s.entropia<max) {
g.otraGrafica();
g.ponColor(Grafica.rojo);
g.inserta(Rankine.punto4.entropia, Rankine.punto4.entalpia);
g.inserta(Rankine.punto5s.entropia, Rankine.punto5s.entalpia);
}



//pinto las isoentrópicas reales
if(!(Rankine.punto1.entropia>max)&&!(Rankine.punto2.entropia<min)) {
   g.otraGrafica();
   g.ponColor(Grafica.negro);
```

```
        g.inserta(Rankine.punto1.entropia, Rankine.punto1.entalpia);
        g.inserta(Rankine.punto2.entropia, Rankine.punto2.entalpia);
        }
        if(!(Rankine.punto2.entropia>max)&&!(Rankine.punto3.entropia<min)) {
            g.otraGrafica();
          g.ponColor(Grafica.negro);
        g.inserta(Rankine.punto2.entropia, Rankine.punto2.entalpia);
        g.inserta(Rankine.punto3.entropia, Rankine.punto3.entalpia);
        }


        if(!(Rankine.punto3.entropia>max)&&!(Rankine.punto4.entropia<min)) {
        g.otraGrafica();
        g.ponColor(Grafica.negro);
        g.inserta(Rankine.punto3.entropia, Rankine.punto3.entalpia);
        g.inserta(Rankine.punto4.entropia, Rankine.punto4.entalpia);
        }


        if(!(Rankine.punto4.entropia>max)&&!(Rankine.punto5.entropia<min)) {
        g.otraGrafica();
        g.ponColor(Grafica.negro);
        g.inserta(Rankine.punto4.entropia, Rankine.punto4.entalpia);
        g.inserta(Rankine.punto5.entropia, Rankine.punto5.entalpia);
        }


        g.pinta();

}

}


public class HiloHS extends Thread{

     public void run(){
            GeneraGraficas.muestraDiagramaHS();
     }
}
```

```
public class HiloTS extends Thread{

    public void run(){

      GeneraGraficas.muestraDiagramaTS();
    }
}

public class HiloTSBrayton extends Thread{

public void run(){

  GeneraGraficas.muestraDiagramaTSCombinado();
}


}
```

```
/**
 * Clase principal del programa. Aquí se ejecuta el método main
 * @author El autor de este TFG
 *
 */
public class Principal {

 public static final String SEPARATOR=";";
 public static final String QUOTE="\"";

 /**
  * Método main del programa
  * @param args
  */
   public static void main(String[] args) {
       TablaSaturadoPresion.leerTabla();
       TablaSaturadoTemperatura.leerTabla();
       TablasRecalentado.leeTablas();
       VentanaInicio vi=new VentanaInicio();
       vi.setVisible(true);
```

```
    }


    /**
     * Método auxiliar para redondear un número
     * @param num
     * @return
     */
    public static double redondear(double num) {
    return Math.round(num*100.0)/100.0;
    }


}


/**
 * Clase que identifica a un punto del ciclo Rankine
 * @author El autor de este TFG
 *
 */
public class PuntoDiagrama {

double entalpia=0;
double entropia=0;
double temperatura=0;
double volumen=0;
double presion=0;
boolean saturado;
boolean recalentado;
boolean subenfriado;
double tituloVapor=-1;


/**
 * Método para imprimir un punto por consola
 */
public void imprimePunto() {
System.out.print("temperatura: "+redondear(this.temperatura)+ " ºC");
System.out.print("  presion: "+redondear(this.presion)+ " bar");
```

```
System.out.print("  entalpia: "+redondear(this.entalpia)+ " kJ/kg");
System.out.print("  entropia: "+redondear(this.entropia)+ " kJ/kgk");
System.out.println("  volumen: "+redondear(this.volumen)+ " m3");
}

/**
 * Método para regenrar un texto descriptivo del punto
 * @param titulo
 * @return texto descriptivo del punto
 */
public String generaTexto(String titulo) {

String retornar=titulo+":  ";
String nuevaPalabra="t: " +redondear(this.temperatura)+"ºC";
nuevaPalabra=this.completarConEspacios(nuevaPalabra, 16);
retornar=retornar+nuevaPalabra;

nuevaPalabra="p: "+redondear(this.presion/10.0)+ " MPa";
nuevaPalabra=this.completarConEspacios(nuevaPalabra, 17);
retornar=retornar+nuevaPalabra;

nuevaPalabra="h: "+redondear(this.entalpia)+ " kJ/kg";
nuevaPalabra=this.completarConEspacios(nuevaPalabra, 21);
retornar=retornar+nuevaPalabra;

nuevaPalabra="s: "+redondear(this.entropia)+ " kJ/kgk";
nuevaPalabra=this.completarConEspacios(nuevaPalabra, 21);
retornar=retornar+nuevaPalabra;

nuevaPalabra="v: "+redondear(this.volumen)+ " m3/kg";
nuevaPalabra=this.completarConEspacios(nuevaPalabra, 15);
retornar=retornar+nuevaPalabra;

return retornar;
}

/**
 * Método para determinar si el punto leído del as tablas de vapor el correcto
 * @return
 */
```

```
    public boolean esValido() {


    if(Double.isNaN(this.entalpia)||Double.isNaN(this.entropia)||Double.isNaN(this.temperatura)
||Double.isNaN(this.presion)) {
    return false;
    }

    return true;

    }

    /**
     * Método para clonar un punto
     */
    public PuntoDiagrama clone() {

    PuntoDiagrama d=new PuntoDiagrama();
    d.entalpia=this.entalpia;
    d.entropia=this.entropia;
    d.volumen=this.volumen;
    d.presion=this.presion;
    d.saturado=this.saturado;
    d.subenfriado=this.subenfriado;
    d.recalentado=this.recalentado;
    d.temperatura=this.temperatura;
    d.tituloVapor=this.tituloVapor;
    return d;

    }

    /**
     * Método auxiliar para redondear un número
     * @param número a redondear
     * @return número redondeado
     */
       public static double redondear(double num) {
       return Math.round(num*1000.0)/1000.0;
       }
```

```
    /**
     * Método auxiliar para compleatar una cadena de texto con espacios
     * @param completar
     * @param longitud
     * @return cadena completa
     */
    public String completarConEspacios(String completar, int longitud) {

    int inicial=completar.length();
    int anadir=longitud-inicial;
    String retornar=""+completar;

    for(int i=0; i<anadir; i++) {
    retornar=retornar+" ";
    }

return retornar;

    }
}



/**
 * Clase que representa un punto del ciclo Brayton
 * @author El autor de este TFG
 *
 */
public class PuntoDiagramaBrayton {
double temperatura;
double presion;
double volumen;
double entropia;

/**
 * Método para mostrar el punto por consola
 */
public void imprimePunto() {
System.out.println("temperatura: "+(this.temperatura-273.15)+" ºC");
System.out.println("presion: "+this.presion/100000 + " bar");
System.out.println("volumen: "+this.volumen);
```

```
System.out.println("entropia: "+this.entropia+" kj/kgk");
}

/**
 * Método para retornar la temperatura en celsius
 * @return temperatura en º C
 */
public double temperaturaCelsius(){
return temperatura-273.15;

}

/**
 * Método para genera texto descriptivo del punto
 * @return texto descriptivo
 */
public String generaTexto() {
String mensaje="";
mensaje=mensaje+completarConEspacios("t: "+redondear(temperatura-273.15)+" ºC", 20);
mensaje=mensaje+completarConEspacios("p: "+redondear(presion/1000000)+" MPa", 20);
return mensaje;
}

/**
 * Método auxiliar para completar con espacios
 * @param completar
 * @param longitud
 * @return cadena completa
 */
public static String completarConEspacios(String completar, int longitud) {

   int inicial=completar.length();
   int anadir=longitud-inicial;
   String retornar=""+completar;

   for(int i=0; i<anadir; i++) {
   retornar=retornar+" ";
   }

           return retornar;
```

```
}

/**
 * Método auxiliar para redondear un número
 * @param num
 * @return
 */
   public static double redondear(double num) {
   return Math.round(num*1000.0)/1000.0;
   }
}


/**
 * Clase que representa un punto en la zona de líquido reclentado
 * @author El autor de este TFG
 *
 */
public class PuntoRecalentado {
double temperatura;
double volumen;
double entalpia;
double entropia;
public double presion;


/**
 * Método para mostrar punto recalentado por consola
 */
public void imprimePunto() {
System.out.print("temperatura: "+this.temperatura);
System.out.print("  presion: "+this.presion);
System.out.print("  entalpia: "+this.entalpia);
System.out.print("  entropia: "+this.entropia);
System.out.println("  volumen: "+this.volumen);
}

/**
 * Método para obtener un punto diagrama a partir de un punto recalentado
```

```
 * @return
 */
public PuntoDiagrama getPuntoDiagrama() {

PuntoDiagrama p=new PuntoDiagrama();
p.entalpia=this.entalpia;
p.entropia=this.entropia;
p.volumen=this.volumen;
p.presion=this.presion;
p.temperatura=this.temperatura;
p.subenfriado=false;
p.recalentado=true;
p.saturado=false;

return p;

}

/**
 * Método para clonar un punto recalentado
 */
public PuntoRecalentado clone() {

PuntoRecalentado d=new PuntoRecalentado();
d.entalpia=this.entalpia;
d.entropia=this.entropia;
d.volumen=this.volumen;
d.presion=this.presion;
d.temperatura=this.temperatura;
return d;

}


/**
 * Método para verificar si el punto leído de las tablas es válido
 * @return
 */
public boolean esValido() {
```

```
    if(Double.isNaN(this.entalpia)||Double.isNaN(this.entropia)||Double.isNaN(this.temperatura)
||Double.isNaN(this.presion)) {
    return false;
    }

    return true;

    }
    }


    /**
     * Clase que representa un punto saturado en el diagrama
     * @author El autor de este TFG
     *
     */
    public class PuntoSaturado {
    double temperatura;
    double presion;
    double entalpiaLiquido;
    double entalpiaVapor;
    double entropiaLiquido;
    double entropiaVapor;
    double volumenLiquido;
    double volumenVapor;
    double tituloVapor=-1;


    /**
     * Método para imprimir un punto por consola
     */
    public void imprime() {
    System.out.print("presion: "+presion);
    System.out.print("  temperatura: "+temperatura);
    System.out.print("  entalpia Liquido: "+entalpiaLiquido);
    System.out.print("  entalpia Vapor: " +entalpiaVapor);
    System.out.print("  entropia Liquido: "+entropiaLiquido);
    System.out.print("  entropia Vapor: "+entropiaVapor);
    System.out.print("  volumen Liquido: "+volumenLiquido);
```

```
System.out.println("  volumen Vapor: "+volumenVapor);
System.out.println("  título Vapor: "+tituloVapor);
}

/**
 * Método para obtener un punto de diagrama a partir de un punto saturado
 * @return
 */
public PuntoDiagrama getPuntoDiagrama() {

if(this.tituloVapor==-1) {
return null;
}
PuntoDiagrama p=new PuntoDiagrama();
p.entalpia=entalpiaLiquido+this.tituloVapor*(entalpiaVapor-entalpiaLiquido);
p.entropia=entropiaLiquido+this.tituloVapor*(entropiaVapor-entropiaLiquido);
p.volumen=volumenLiquido+this.tituloVapor*(volumenVapor-volumenLiquido);
p.presion=presion;
p.temperatura=temperatura;
p.subenfriado=false;
p.recalentado=false;
p.saturado=true;
p.tituloVapor=this.tituloVapor;

return p;

}

/**
 * Método para clonar un punto saturado
 */
public PuntoSaturado clone() {

PuntoSaturado d=new PuntoSaturado();
d.entalpiaVapor=this.entalpiaVapor;
d.entropiaVapor=this.entropiaVapor;
d.entalpiaLiquido=this.entalpiaLiquido;
d.entropiaLiquido=this.entropiaLiquido;
d.volumenLiquido=this.volumenLiquido;
d.volumenVapor=this.volumenVapor;
```

```
d.temperatura=this.temperatura;
d.tituloVapor=this.tituloVapor;
return d;

}

}



/**
 * Clase que representa un punto subenfriado en el diagrama
 * @author El autor de este TFG
 *
 */
public class PuntoSubenfriado {
double temperatura;
double volumen;
double entalpia;
double entropia;
public double presion;

/**
 * Método para imprimir un punto por consola
 */
public void imprimePunto() {
System.out.print("temperatura: "+this.temperatura);
System.out.print("  presion: "+this.presion);
System.out.print("  entalpia: "+this.entalpia);
System.out.print("  entropia: "+this.entropia);
System.out.println("  volumen: "+this.volumen);
}

/**
 * Método para obtener un punti de diagrama a partir de un punto subenfriado
 * @return
 */
public PuntoDiagrama getPuntoDiagrama() {

PuntoDiagrama p=new PuntoDiagrama();
p.entalpia=this.entalpia;
```

```
    p.entropia=this.entropia;
    p.volumen=this.volumen;
    p.presion=this.presion;
    p.temperatura=this.temperatura;
    p.subenfriado=true;
    p.recalentado=false;
    p.saturado=false;

    return p;

    }

    /**
     * Método para clonar un punto subenfriado
     */
    public PuntoSubenfriado clone() {

    PuntoSubenfriado d=new PuntoSubenfriado();
    d.entalpia=this.entalpia;
    d.entropia=this.entropia;
    d.volumen=this.volumen;
    d.presion=this.presion;
    d.temperatura=this.temperatura;
    return d;

    }


    /**
     * Método para verificar si el punto subenfriado leído de las tablas es correcto
     * @return
     */
    public boolean esValido() {


    if(Double.isNaN(this.entalpia)||Double.isNaN(this.entropia)||Double.isNaN(this.temperatura)
||Double.isNaN(this.presion)) {
    return false;
    }
```

```
return true;

}
}


/**
 * Clase que almacena los datos correspondientes al ciclo Rankie
 * y los métodos para el cálculo del mismo.
 * @author El autor de este TFG
 *
 */
public class Rankine {

static double rendimientoIsoentropico=0.95;
static double rendimientoIsoentropicoBombas=0.95;
static double rendimientoMecanicoBombas=0.97;
static double rendimientoMecanicoTurbinas=0.97;
static double rendimientoCaldera=1;

static double presionCaldera=60;
static double presionExtraccion1=20;
static double presionExtraccion2=7;
static double presionExtraccion3=1.5;
static double PresionCondensador=0.06;

static final double mtMax=26.61;// kg/s
static double mt=26.61;
static double m1=0;
static double m2=0;
static double m3=0;
static double m4=0;
static double mTurboGenerador;
static double mTurbobomba;
static double mCaldera;
static double mGas;
static double mFO;

static double ConsumoFO=0;
static double ConsumoGas=0;
```

```
final static double mFOtankHeating=0.2472;
final static double mFOheatingMax=0.1897;

static double potenciaTurbina=0;
static double potenciaTurbinaEnEje=0;
static double potenciaNetaPlanta=0;
static double rendimientoPlanta=0;
static double potenciaAbsorbidaTurbobomba=0;
static double potenciaentregadaTurbobomba=0;
static double potenciaAbsorbidaTurbogenerador=1840;
static double calorAportadoCaldera=0;
static double calorCombustion;


final static double PC_FUELOIL=39710;
static double PC_GAS=39900;


static double temperaturaFocoFrio=25;



static PuntoSaturado caldera=new PuntoSaturado();
static PuntoSaturado extraccion1=new PuntoSaturado();
static PuntoSaturado extraccion2=new PuntoSaturado();
static PuntoSaturado extraccion3=new PuntoSaturado();
static PuntoSaturado condensador=new PuntoSaturado();

static PuntoDiagrama punto1;
static PuntoDiagrama punto2;
static PuntoDiagrama punto2s;
static PuntoDiagrama punto3;
static PuntoDiagrama punto3s;
static PuntoDiagrama punto4;
static PuntoDiagrama punto4s;
static PuntoDiagrama punto5;
static PuntoDiagrama punto5s;
static PuntoDiagrama punto6;
static PuntoDiagrama punto7s;
static PuntoDiagrama punto7;
```

```
    static PuntoDiagrama punto8;
    static PuntoDiagrama punto9;
    static PuntoDiagrama punto10s;
    static PuntoDiagrama punto10;

    static boolean conRecalentamientos=false;
    static double temperaturaMaxima=510;

    static double indiceCarga=1;
    static double proporcionFuelOil=1;


    static boolean combinado=false;


    /**
     * Método que calcula los puntos característicos del ciclo rankine
     */
    public static void calculaPuntos() {

    PuntoSaturado
paux=TablaSaturadoTemperatura.getPuntoTemperatura(Rankine.temperaturaFocoFrio);
    Rankine.PresionCondensador=paux.presion;

    ///Calculo las isóbaras
    caldera=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionCaldera);
    extraccion1=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion1);
    extraccion2=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion2);
    extraccion3=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion3);
    condensador=TablaSaturadoTemperatura.getPuntoPresion(Rankine.PresionCondensador);

    double entalpia=-1;



    //Calculo los puntos del diagrama
    punto1=new PuntoDiagrama();
    PuntoRecalentado p1=TablasRecalentado.getPuntoPresionTemperatura(presionCaldera,
temperaturaMaxima);
```

```
    punto1=p1.getPuntoDiagrama();
    punto1.imprimePunto();

    punto2s=new PuntoDiagrama();
    PuntoRecalentado p2s=TablasRecalentado.getPuntoPresionEntropia(presionExtraccion1,
punto1.entropia);

    punto2s=p2s.getPuntoDiagrama();

    punto2s.imprimePunto();

    punto2= new PuntoDiagrama();
    entalpia=punto1.entalpia-Rankine.rendimientoIsoentropico*(punto1.entalpia-
punto2s.entalpia);
    PuntoRecalentado p2=TablasRecalentado.getPuntoPresionEntalpia(presionExtraccion1,
entalpia);

    punto2=p2.getPuntoDiagrama();

    punto2.imprimePunto();

    punto3s=new PuntoDiagrama();
    PuntoRecalentado p3s=TablasRecalentado.getPuntoPresionEntropia(presionExtraccion2,
punto2.entropia);

    punto3s=p3s.getPuntoDiagrama();

    punto3s.imprimePunto();

    punto3= new PuntoDiagrama();
    entalpia=punto2.entalpia-Rankine.rendimientoIsoentropico*(punto2.entalpia-
punto3s.entalpia);
    PuntoRecalentado p3=TablasRecalentado.getPuntoPresionEntalpia(presionExtraccion2,
entalpia);

    punto3=p3.getPuntoDiagrama();

    punto3.imprimePunto();

    punto4s=new PuntoDiagrama();
```

```
    PuntoRecalentado p4s=TablasRecalentado.getPuntoPresionEntropia(presionExtraccion3,
punto3.entropia);

    punto4s=p4s.getPuntoDiagrama();


    punto4s.imprimePunto();

    punto4= new PuntoDiagrama();
    entalpia=punto3.entalpia-Rankine.rendimientoIsoentropico*(punto3.entalpia-
punto4s.entalpia);
    PuntoRecalentado p4=TablasRecalentado.getPuntoPresionEntalpia(presionExtraccion3,
entalpia);

    punto4=p4.getPuntoDiagrama();

    punto4.imprimePunto();

    punto5s=new PuntoDiagrama();
    PuntoRecalentado p5s=TablasRecalentado.getPuntoPresionEntropia(PresionCondensador,
punto4.entropia);

    punto5s=p5s.getPuntoDiagrama();

    punto5s.imprimePunto();

    punto5= new PuntoDiagrama();
    entalpia=punto4.entalpia-Rankine.rendimientoIsoentropico*(punto4.entalpia-
punto5s.entalpia);
    PuntoRecalentado p5=TablasRecalentado.getPuntoPresionEntalpia(PresionCondensador,
entalpia);

    punto5=p5.getPuntoDiagrama();

    punto5.imprimePunto();


    PuntoSaturado p6=TablaSaturadoTemperatura.getPuntoPresion(p5.presion);
    p6.tituloVapor=0;
    punto6=p6.getPuntoDiagrama();
```

```
    punto6.imprimePunto();

    PuntoSubenfriado p7s =
TablasSubenfriado.getPuntoPresionEntropia(Rankine.presionExtraccion3, punto6.entropia);
    punto7s=p7s.getPuntoDiagrama();

    punto7s.imprimePunto();


    entalpia=(punto7s.entalpia-
punto6.entalpia)/Rankine.rendimientoIsoentropicoBombas+punto6.entalpia;
    PuntoSubenfriado p7=TablasSubenfriado.getPuntoPresionEntalpia(Rankine.presionExtraccion3,
entalpia);
    punto7=p7.getPuntoDiagrama();

    punto7.imprimePunto();


    PuntoSaturado p8=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion3);
    p8.tituloVapor=0;
    punto8=p8.getPuntoDiagrama();


    punto8.imprimePunto();

    PuntoSaturado p9=TablaSaturadoTemperatura.getPuntoPresion(Rankine.presionExtraccion2);
    p9.tituloVapor=0;
    punto9=p9.getPuntoDiagrama();


    punto9.imprimePunto();


    PuntoSubenfriado p10s = TablasSubenfriado.getPuntoPresionEntropia(Rankine.presionCaldera,
punto9.entropia);
    punto10s=p10s.getPuntoDiagrama();

    punto10s.imprimePunto();
```

```
    entalpia=(punto10s.entalpia-
punto9.entalpia)/Rankine.rendimientoIsoentropicoBombas+punto9.entalpia;
    PuntoSubenfriado p10=TablasSubenfriado.getPuntoPresionEntalpia(Rankine.presionCaldera,
entalpia);
    punto10=p10.getPuntoDiagrama();


    punto10.imprimePunto();

    }



    /**
     * Método que calcula los distintos caudales másicos del ciclo rankine
     */
    public static void calcularCaudales() {

    double h1=punto1.entalpia;
    double h2=punto2.entalpia;
    double h3=punto3.entalpia;
    double h4=punto4.entalpia;
    double h5=punto5.entalpia;
    double h6=punto6.entalpia;
    double h7=punto7.entalpia;
    double h8=punto8.entalpia;
    double h9=punto9.entalpia;
    double h10=punto10.entalpia;



    mCaldera=calculaMtMax()*indiceCarga;
    mTurboGenerador=Rankine.potenciaAbsorbidaTurbogenerador/(h1-h5);
    Rankine.potenciaentregadaTurbobomba=(h10-h9)*mCaldera;

    Rankine.potenciaAbsorbidaTurbobomba=potenciaentregadaTurbobomba/Rankine.rendimientoMecanico
Bombas;
    Rankine.mTurbobomba=Rankine.potenciaAbsorbidaTurbobomba/(h1-h5);
    mt=mCaldera-mTurboGenerador-mTurbobomba;
```

```
    m1=Rankine.mFOtankHeating+Rankine.mFOheatingMax*Rankine.indiceCarga*Rankine.proporcionFuelO
il;
    m2=(h9-h8)*mt/(h3-h8);
    m3=((h8-h7)*mt-(h8-h7)*m2)/(h4-h7);
    m4=mt-m2-m3;



    }

    /**
     * Método que permite calcular las potencias y los rendimeintos del cilo rankine
     */
    public static void calcularPotenciaRendimiento() {

    double h1=punto1.entalpia;
    double h2=punto2.entalpia;
    double h3=punto3.entalpia;
    double h4=punto4.entalpia;
    double h5=punto5.entalpia;
    double h6=punto6.entalpia;
    double h7=punto7.entalpia;
    double h8=punto8.entalpia;
    double h9=punto9.entalpia;
    double h10=punto10.entalpia;

    calorAportadoCaldera=(h1-h10)*Rankine.mCaldera;
    potenciaTurbina=(h1-h2)*mt+(h2-h3)*(mt-m1)+(h3-h4)*(m3+m4)+(h4-h5)*m4;
    potenciaTurbinaEnEje=(potenciaTurbina*rendimientoMecanicoTurbinas);
    potenciaAbsorbidaTurbobomba=(h1-h5)*Rankine.mTurbobomba;
    Rankine.calorCombustion=calorAportadoCaldera/rendimientoCaldera;

    if(!combinado) {
    Rankine.rendimientoPlanta=(potenciaTurbinaEnEje)/calorCombustion;
    }else {

    Rankine.rendimientoPlanta=(Brayton.potenciaEje+potenciaTurbinaEnEje)/(Brayton.calorAportado
*Brayton.caudalAire);
```

```
}

}



/**
 * Método que calcula el consumo de cada combustible
 */
public static void calculaConsumoCombustible() {


if(Rankine.combinado==false) {

double x=Rankine.proporcionFuelOil;
double y=1.0-x;
double pcCombinado=Rankine.PC_FUELOIL*x+Rankine.PC_GAS*y;
double masaCombustibleTotal=Rankine.calorCombustion/pcCombinado;
mGas=masaCombustibleTotal*(1-x);
mFO=masaCombustibleTotal*x;


}else {
mFO=0;
mGas=Brayton.calorAportado*Brayton.caudalAire/PC_GAS;
}
}


/**
 * Método que calcula el caudal total
 * @return
 */
public static double calculaMtMax() {

if(Rankine.combinado) {
```

```
    double
calorRecuperado=Brayton.calculaCalorRecuperacion(Rankine.punto10.temperatura)*Rankine.rendimien
toCaldera;
    System.out.println(calorRecuperado);
    double m=calorRecuperado/(Rankine.punto1.entalpia-Rankine.punto10.entalpia);
    return m;

    }else {
    return Rankine.mtMax;
    }



    }




    /**
     * Método auxiliar para redondear un número
     * @param número
     * @return
     */
      public static double redondear(double num) {
       return Math.round(num*100.0)/100.0;
      }


      /**
       * Método que genera un texto descriptivo de los puntos que describen el ciclo.
       * @return
       */
      public static String generaTexto() {


    String mensaje="";

    mensaje=mensaje+Rankine.punto1.generaTexto("Punto 1  ")+"\n";
```

```
    mensaje=mensaje+Rankine.punto2s.generaTexto("Punto 2s ")+"\n";
    mensaje=mensaje+Rankine.punto2.generaTexto("Punto 2  ")+"\n";
    mensaje=mensaje+Rankine.punto3s.generaTexto("Punto 3s ")+"\n";
    mensaje=mensaje+Rankine.punto3.generaTexto("Punto 3  ")+"\n";
    mensaje=mensaje+Rankine.punto4s.generaTexto("Punto 4s ")+"\n";
    mensaje=mensaje+Rankine.punto4.generaTexto("Punto 4  ")+"\n";
    mensaje=mensaje+Rankine.punto5s.generaTexto("Punto 5s ")+"\n";
    mensaje=mensaje+Rankine.punto5.generaTexto("Punto 5  ")+"\n";
    mensaje=mensaje+Rankine.punto6.generaTexto("Punto 6  ")+"\n";
    mensaje=mensaje+Rankine.punto7s.generaTexto("Punto 7  ")+"\n";
    mensaje=mensaje+Rankine.punto7.generaTexto("Punto 7s ")+"\n";
    mensaje=mensaje+Rankine.punto8.generaTexto("Punto 8  ")+"\n";
    mensaje=mensaje+Rankine.punto9.generaTexto("Punto 9  ")+"\n";
    mensaje=mensaje+Rankine.punto10s.generaTexto("Punto 10s")+"\n";
    mensaje=mensaje+Rankine.punto10.generaTexto("Punto 10 ")+"\n";


    String texto1="";
    texto1=texto1+("\nCaudal mt = "+redondear(mt)+" kg/s");
    texto1=texto1+("\nCaudal m1 = "+redondear(m1)+" kg/s");
    texto1=texto1+("\nCaudal m2 = "+redondear(m2)+" kg/s");
    texto1=texto1+("\nCaudal m3 = "+redondear(m3)+" kg/s");
    texto1=texto1+("\nCaudal m4 = "+redondear(m4)+" kg/s");
    texto1=texto1+("\nCaudal t.generador = "+redondear(mTurboGenerador)+" kg/s");
    texto1=texto1+("\nCaudal t.bomba = "+redondear(mTurbobomba)+" kg/s");
    texto1=texto1+("\nCaudal Caldera = "+redondear(mCaldera)+" kg/s\n");

    String texto2="";
    if(combinado==false) {

    texto2=texto2+("\nCalor de la combustión =              "+redondear(calorCombustion)+"
kw\n");
    texto2=texto2+("Calor aportado por la caldera =        "+redondear(calorAportadoCaldera)+"
kw\n");
    texto2=texto2+("Potencia turbina =                     "+redondear(potenciaTurbina)+"
kw\n");
    texto2=texto2+("Potencia turbina en el eje =           "+redondear(potenciaTurbinaEnEje)+"
kw\n");
    texto2=texto2+("Potencia absorbida turbobomba =
"+redondear(potenciaAbsorbidaTurbobomba)+" kw\n");
```

```
    texto2=texto2+("Potencia absorbida turbogenerador =
"+redondear(potenciaAbsorbidaTurbogenerador)+" kw\n");
    texto2=texto2+("Rendimiento neto Planta =
"+redondear(rendimientoPlanta)*100.0+" %\n");
    texto2=texto2+("Consumo de Fuel Oil =                 "+redondear(mFO)+" kg/s\n");
    texto2=texto2+("Consumo de Gas natural =              "+redondear(mGas)+" kg/s\n");

    }else {

    texto2=texto2+("\nCalor de la recuperación =            "+redondear(calorCombustion)+"
kw\n");
    texto2=texto2+("Calor aprovechado recuperación =      "+redondear(calorAportadoCaldera)+"
kw\n");
    texto2=texto2+("Potencia en el eje turbina vapor =    "+redondear(potenciaTurbina)+"
kw\n");
    texto2=texto2+("Potencia en el eje turbina de gas =   "+redondear(Brayton.potenciaEje)+"
kw\n");
    texto2=texto2+("Potencia neta de la planta COGES =
"+redondear(potenciaTurbinaEnEje+Brayton.potenciaEje)+" kw\n");
    texto2=texto2+("Potencia absorbida turbobomba =
"+redondear(potenciaAbsorbidaTurbobomba)+" kw\n");
    texto2=texto2+("Potencia absorbida turbogenerador =
"+redondear(potenciaAbsorbidaTurbogenerador)+" kw\n");
    texto2=texto2+("Rendimiento neto Planta COGES=
"+redondear(rendimientoPlanta)*100.0+" %\n");
    texto2=texto2+("Consumo de Gas natural =              "+redondear(mGas)+" kg/s\n");

    }

    mensaje=mensaje+repartirEnColumnas(texto2, texto1);

    return "Puntos significativos del ciclo Rankine\n----------------------------------------
-----------------------"
    + "\n"+mensaje +"\n";

      }


      /**
       * Método auxiliar que estructura el texto en columnas para que pueda
```

```
   * mostrarse correctamente en la pantalla
   * @param texto1
   * @param texto2
   * @return
   */
  public static String repartirEnColumnas(String texto1, String texto2) {


String[]  lista1=texto1.split("\n");
String[]  lista2=texto2.split("\n");
String retornar="";

boolean finLista1=false;
boolean finLista2=false;
int ultimaJ=0;
int ultimaI=0;

for(int i=0, j=0; i<lista1.length&&j<lista2.length; i++, j++) {
while(lista1[i].equals("")) {
i++;
if(lista1.length==i) {
finLista1=true;
ultimaJ=j;
ultimaI=i;
break;
}
}
while(lista2[j].equals("")) {
j++;
if(lista2.length==j) {
finLista2=true;
ultimaJ=j;
ultimaI=i;
break;
}
}

if(!finLista1&&!finLista2) {
retornar=retornar+"\n"+completarConEspacios(lista1[i], 70)+lista2[j];
}else {
```

```
ultimaJ=j;
ultimaI=i;
break;
}

if(i==lista1.length-1){
finLista1=true;
}

if(j==lista2.length-1){
finLista2=true;
}

ultimaJ=j;
ultimaI=i;
}


if(finLista1&&finLista2) {
return retornar;
}else if(finLista2) {
for(int i=ultimaI+1; i<lista1.length; i++) {
retornar=retornar+"\n"+lista1[i];
}
}else if(finLista1){

for(int i=ultimaJ+1; i<lista2.length; i++) {
retornar=retornar+"\n"+completarConEspacios("", 70)+lista2[i];
}

}


return retornar;

  }


  /**
   * Método auxiliar para completar una cadena e texto con espacios
```

```
    * @param completar
    * @param longitud
    * @return
    */
    public static String completarConEspacios(String completar, int longitud) {

    int inicial=completar.length();
    int anadir=longitud-inicial;
    String retornar=""+completar;

    for(int i=0; i<anadir; i++) {
    retornar=retornar+" ";
    }

return retornar;

    }


}
```

```
/**
 * Clase que representa y realiza la lectura de las tablas de vapor recalentado
 * @author El autor de este TFG
 *
 */
public class TablaRecalentado {

double presion;
ArrayList<PuntoRecalentado> listaPuntos =new ArrayList<PuntoRecalentado>();

public void imprimeTabla() {

System.out.println("TablaRecalentado (Presion="+this.presion+")");
for(int i=0; i<listaPuntos.size(); i++) {
listaPuntos.get(i).imprimePunto();
```

```
    }

    }

    /**
     * Método para leer el punto correspondiente a una temperatura concreta
     * @param temperatura
     * @return
     */
    public PuntoRecalentado getPuntoTemperatura(double temperatura) {

    PuntoRecalentado anterior=listaPuntos.get(0);
    PuntoRecalentado actual=listaPuntos.get(0);

    for(int i=0; i<listaPuntos.size(); i++) {
    anterior=actual;
    actual=this.listaPuntos.get(i);
    if(actual.temperatura>temperatura) {
    break;
    }
    }

    PuntoRecalentado p=new PuntoRecalentado();
    double factorInterpolacion=(temperatura-anterior.temperatura)/(actual.temperatura-
anterior.temperatura);
    p.entalpia=anterior.entalpia+(actual.entalpia-anterior.entalpia)*factorInterpolacion;
    p.entropia=anterior.entropia+(actual.entropia-anterior.entropia)*factorInterpolacion;
    p.volumen=anterior.volumen+(actual.volumen-anterior.volumen)*factorInterpolacion;
    p.temperatura=temperatura;
    p.presion=this.presion;

    return p;


    }


    /**
     * Método para leer el punto correspondiente a una entropía concreta
```

```
     * @param entropía
     * @return
     */
    public PuntoRecalentado getPuntoEntropia(double entropia) {
    PuntoRecalentado anterior=listaPuntos.get(0);
    PuntoRecalentado actual=listaPuntos.get(0);

    for(int i=0; i<listaPuntos.size(); i++) {
    anterior=actual;
    actual=this.listaPuntos.get(i);
    if(actual.entropia>entropia) {
    break;
    }
    }

    PuntoRecalentado p=new PuntoRecalentado();
    double factorInterpolacion=(entropia-anterior.entropia)/(actual.entropia-
anterior.entropia);
    p.entalpia=anterior.entalpia+(actual.entalpia-anterior.entalpia)*factorInterpolacion;
    p.entropia=anterior.entropia+(actual.entropia-anterior.entropia)*factorInterpolacion;
    p.volumen=anterior.volumen+(actual.volumen-anterior.volumen)*factorInterpolacion;
    p.temperatura=anterior.temperatura+(actual.temperatura-
anterior.temperatura)*factorInterpolacion;;
    p.presion=this.presion;

    return p;
    }


    /**
     * Método para leer el punto correspondiente a una entalpia concreta
     * @param entalpia
     * @return
     */
    public PuntoRecalentado getPuntoEntalpia(double entalpia) {
    PuntoRecalentado anterior=listaPuntos.get(0);
    PuntoRecalentado actual=listaPuntos.get(0);

    for(int i=0; i<listaPuntos.size(); i++) {
    anterior=actual;
```

```
    actual=this.listaPuntos.get(i);
    if(actual.entalpia>entalpia) {
    break;
    }
    }

    PuntoRecalentado p=new PuntoRecalentado();
    double factorInterpolacion=(entalpia-anterior.entalpia)/(actual.entalpia-
anterior.entalpia);
    p.entalpia=anterior.entalpia+(actual.entalpia-anterior.entalpia)*factorInterpolacion;
    p.entropia=anterior.entropia+(actual.entropia-anterior.entropia)*factorInterpolacion;
    p.volumen=anterior.volumen+(actual.volumen-anterior.volumen)*factorInterpolacion;
    p.temperatura=anterior.temperatura+(actual.temperatura-
anterior.temperatura)*factorInterpolacion;;
    p.presion=this.presion;

    return p;
    }



    }



    /**
     * Clase que representa y realiza la lectura de las tablas de vapor saturado ordenada por
temperaturas
     * @author El autor de este TFG
     *
     */
    public class TablaSaturadoTemperatura {

    public static final String SEPARATOR=";";
    public static final String QUOTE="\"";

    final static int PRESION=1;
    final static int TEMPERATURA=0;
```

```
    final static int VOLUMEN_LIQUIDO=2;
    final static int VOLUMEN_VAPOR=3;
    final static int ENTALPIA_LIQUIDO=6;
    final static int ENTALPIA_VAPOR=8;
    final static int ENTROPIA_LIQUIDO=9;
    final static int ENTROPIA_VAPOR=10;

    final static int primera_linea=6;


    public static ArrayList<PuntoSaturado> listaPuntos=new ArrayList<PuntoSaturado>();

    /**
     * Metodo para leer la tabla
     */
    public static void leerTabla(){

          BufferedReader br = null;

          try {

             br =new BufferedReader(new FileReader("src\\temperaturaSaturado.csv"));
             String line = br.readLine();
             int contadorLineas=1;
             while (null!=line) {
                  contadorLineas++;
                  if(contadorLineas>primera_linea) {

                String [] fields = line.split(SEPARATOR);

                fields = removeTrailingQuotes(fields);



                PuntoSaturado punto=new PuntoSaturado();

punto.entalpiaLiquido=Double.parseDouble(fields[ENTALPIA_LIQUIDO].replaceAll(",", "."));
                punto.entalpiaVapor=Double.parseDouble(fields[ENTALPIA_VAPOR].replaceAll(",",
"."));
```

```
punto.entropiaLiquido=Double.parseDouble(fields[ENTROPIA_LIQUIDO].replaceAll(",", "."));
                punto.entropiaVapor=Double.parseDouble(fields[ENTROPIA_VAPOR].replaceAll(",",
".."));
                punto.presion=Double.parseDouble(fields[PRESION].replaceAll(",", "."));
                punto.temperatura=Double.parseDouble(fields[TEMPERATURA].replaceAll(",", "."));
                punto.volumenLiquido=Double.parseDouble(fields[VOLUMEN_LIQUIDO].replaceAll(",",
".."));
                punto.volumenVapor=Double.parseDouble(fields[VOLUMEN_VAPOR].replaceAll(",",
".."));
                listaPuntos.add(punto);


                  }


                  line = br.readLine();
              }

           } catch (Exception e) {
              e.printStackTrace();
           } finally {
              if (null!=br) {
                 try {
    br.close();
    } catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
    }
              }
           }
    }


    /**
     * Metodo para mostrar toda la tabla por consola.
     */
    public static void imprimeTabla(){

    System.out.println("TABLA VAPOR SATURADO (Ordenado por temperaturas)\n----------------------
----------------------------------------------------");
```

```
for(int i=0; i<listaPuntos.size(); i++) {
listaPuntos.get(i).imprime();
}
System.out.println("\n\n");

}



/**
 * Método para obter un punto saturado para una presion dada
 * @param presion
 * @return
 */
public static PuntoSaturado getPuntoPresion(double presion) {


IF97 tablas=new IF97(UnitSystem.SI);
PuntoSaturado p=new PuntoSaturado();

p.entalpiaLiquido=tablas.specificEnthalpyPX(presion*100000, 0)/1000;
p.entalpiaVapor=tablas.specificEnthalpyPX(presion*100000, 1)/1000;
p.entropiaLiquido=tablas.specificEntropyPX(presion*100000, 0)/1000;
p.entropiaVapor=tablas.specificEntropyPX(presion*100000, 1)/1000;
p.presion=presion;
p.temperatura=tablas.saturationTemperatureP(presion*100000)-273.15;
p.tituloVapor=-1;
p.volumenLiquido=tablas.specificVolumePX(presion*100000, 0);
p.volumenVapor=tablas.specificVolumePX(presion*100000, 1);

return p;


}


/**
 * Método para obter un punto saturado para una entropia y presion dadas
 * @param presion
 * @return
```

```
 */
public static PuntoSaturado getPuntoPresionEntropia(double presion, double entropia) {


IF97 tablas=new IF97(UnitSystem.SI);
PuntoSaturado p=new PuntoSaturado();


p.tituloVapor=tablas.vapourFractionPS(presion*100000, entropia*1000);
p.entalpiaLiquido=tablas.specificEnthalpyPX(presion*100000, 0)/1000;
p.entalpiaVapor=tablas.specificEnthalpyPX(presion*100000, 1)/1000;
p.entropiaLiquido=tablas.specificEntropyPX(presion*100000, 0)/1000;
p.entropiaVapor=tablas.specificEntropyPX(presion*100000, 1)/1000;
p.presion=presion;
p.temperatura=tablas.saturationTemperatureP(presion*100000)-273.15;

p.volumenLiquido=tablas.specificVolumePX(presion*100000, 0);
p.volumenVapor=tablas.specificVolumePX(presion*100000, 1);

return p;


}

/**
 * Método auxiliar para eliinar las comillas e una cadena de texto
 */
private static String[] removeTrailingQuotes(String[] fields) {

      String result[] = new String[fields.length];

      for (int i=0;i<result.length;i++){
         result[i] = fields[i].replaceAll("^"+QUOTE, "").replaceAll(QUOTE+"$", "");
      }
      return result;
}


/**
 * Método para realizar la lectura de un punto en base a la presión y la entalpía
```

```
 * @param presion
 * @param entalpia
 * @return
 */
public static PuntoSaturado getPuntoPresionEntalpia(double presion, double entalpia) {

IF97 tablas=new IF97(UnitSystem.SI);
PuntoSaturado p=new PuntoSaturado();


p.tituloVapor=tablas.vapourFractionPH(presion*100000, entalpia*1000);
p.entalpiaLiquido=tablas.specificEnthalpyPX(presion*100000, 0)/1000;
p.entalpiaVapor=tablas.specificEnthalpyPX(presion*100000, 1)/1000;
p.entropiaLiquido=tablas.specificEntropyPX(presion*100000, 0)/1000;
p.entropiaVapor=tablas.specificEntropyPX(presion*100000, 1)/1000;
p.presion=presion;
p.temperatura=tablas.saturationTemperatureP(presion*100000)-273.15;

p.volumenLiquido=tablas.specificVolumePX(presion*100000, 0);
p.volumenVapor=tablas.specificVolumePX(presion*100000, 1);

return p;

}


/**
 * Método para realizar la lectura de un punto saturado a aprtir de su temperatura
 * @param temperatura
 * @return
 */
public static PuntoSaturado getPuntoTemperatura(double temperatura) {
IF97 tablas=new IF97(UnitSystem.SI);
PuntoSaturado p=new PuntoSaturado();

p.entalpiaLiquido=tablas.specificEnthalpyTX(temperatura+273.15, 0)/1000;
p.entalpiaVapor=tablas.specificEnthalpyTX(temperatura+273.15, 1)/1000;
p.entropiaLiquido=tablas.specificEntropyTX(temperatura+273.15, 0)/1000;
p.entropiaVapor=tablas.specificEntropyTX(temperatura+273.15, 1)/1000;
p.presion=tablas.saturationPressureT(temperatura+273.15)/100000;
```

```
p.temperatura=temperatura;
p.tituloVapor=-1;
p.volumenLiquido=tablas.specificVolumeTX(temperatura+273.15, 0);
p.volumenVapor=tablas.specificVolumeTX(temperatura+273.15, 1);

return p;
}


}



/**
 * Calse que agrupa todas las tablas de vapor recalentado y
 * permite hacer busquedas de puntos concretos entre ellas
 * @author USER
 *
 */
public class TablasRecalentado {

public static final String SEPARATOR=";";
public static final String QUOTE="\"";

final static int TEMPERATURA=0;
final static int VOLUMEN=1;
final static int ENTALPIA=3;
final static int ENTROPIA=4;

final static int primera_linea=6;



public static ArrayList<TablaRecalentado> listaTablas=new ArrayList<TablaRecalentado>();


/**
 * Método para mostrar las tablas por consola
 */
```

```
    public static void imprimeTablas() {
    for(int i=0; i<listaTablas.size();i++) {
    listaTablas.get(i).imprimeTabla();
    System.out.println("\n");
    }

    }


    /**
     * Método para obtener un punto recalentado en base a la presión  y la temperatura
     * @param presion
     * @param temperatura
     * @return
     */
    public static PuntoRecalentado getPuntoPresionTemperatura(double presion, double
temperatura) {


    IF97 tablas=new IF97(UnitSystem.SI);
    PuntoRecalentado p=new PuntoRecalentado();

    p.entalpia=tablas.specificEnthalpyPT(presion*100000, temperatura+273.15)/1000;
    p.entropia=tablas.specificEntropyPT(presion*100000, temperatura+273.15)/1000;
    p.presion=presion;
    p.temperatura=temperatura;
    p.volumen=tablas.specificVolumePT(presion*100000.0, temperatura+273.15);

    return p;

    }

    /**
     * Método para obtener un punto recalentado a partir de la presion y la entropía
     * @param presion
     * @param entropia
     * @return
     */
    public static PuntoRecalentado getPuntoPresionEntropia(double presion, double entropia) {
```

```
IF97 tablas=new IF97(UnitSystem.SI);
PuntoRecalentado p=new PuntoRecalentado();

p.entalpia=tablas.specificEnthalpyPS(presion*100000, entropia*1000)/1000;
p.entropia=entropia;
p.presion=presion;
p.temperatura=tablas.temperaturePS(presion*100000, entropia*1000)-273.15;
p.volumen=tablas.specificVolumePS(presion*100000.0, entropia*1000);

return p;


}

/**
 * Método para obtener un punto recalentado a partir de la presión y la entalpía
 * @param presion
 * @param entalpia
 * @return
 */
public static PuntoRecalentado getPuntoPresionEntalpia(double presion, double entalpia) {



IF97 tablas=new IF97(UnitSystem.SI);
PuntoRecalentado p=new PuntoRecalentado();

p.entalpia=entalpia;
p.entropia=tablas.specificEntropyPH(presion*100000, entalpia*1000)/1000;
p.presion=presion;
p.temperatura=tablas.temperaturePH(presion*100000, entalpia*1000)-273.15;
p.volumen=tablas.specificVolumePH(presion*100000.0, entalpia*1000);

return p;


}
```

```
/**
 * Método para leeer las tablas de vapor recalentado
 */
public static void leeTablas() {

BufferedReader br = null;

      try {

         br =new BufferedReader(new FileReader("src\\sobrecalentado.csv"));
         String line = br.readLine();
         TablaRecalentado tabla=null;

         while (null!=line) {



            String [] fields = line.split(SEPARATOR);

                  if(fields.length==0) {
                        continue;
                  }

            fields = removeTrailingQuotes(fields);

            if(fields[0].contains("bar")) {
             //anado la tabla anterior siempre que exista
             if(tabla!=null) {
                   listaTablas.add(tabla);
             }

             tabla = new TablaRecalentado();
             tabla.presion=Double.parseDouble(fields[1].replaceAll(",", "."));
            }else {

             PuntoRecalentado p=new PuntoRecalentado();
             p.temperatura=Double.parseDouble(fields[TEMPERATURA].replaceAll(",","."));
             p.volumen=Double.parseDouble(fields[VOLUMEN].replaceAll(",","."));
             p.entalpia=Double.parseDouble(fields[ENTALPIA].replaceAll(",","."));
             p.entropia=Double.parseDouble(fields[ENTROPIA].replaceAll(",","."));
```

```
                p.presion=tabla.presion;
                tabla.listaPuntos.add(p);

               }


                line = br.readLine();
            }

            //anado al ultima tabla a la lista de tablas
            listaTablas.add(tabla);

         } catch (Exception e) {
           e.printStackTrace();
         } finally {
            if (null!=br) {
               try {
br.close();
} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
            }
         }
}

/***
 * Método auxiliar para eliminar las comillas de una cadena de texto
 * @param fields
 * @return
 */
private static String[] removeTrailingQuotes(String[] fields) {

      String result[] = new String[fields.length];

      for (int i=0;i<result.length;i++){
         result[i] = fields[i].replaceAll("^"+QUOTE, "").replaceAll(QUOTE+"$", "");
      }
      return result;
}
```

```
    }


    /**
     * Clase que representas las tablas de líquido subenfriado y realiza la lectura de los
puntos de las mismas
     * @author El autor de este TFG
     *
     */
    public class TablasSubenfriado {


    /**
     * Método para obtener un punto subefriado correspondiente a una presion y temperatura dadas
     * @param presion
     * @param temperatura
     * @return
     */
    public static PuntoSubenfriado getPuntoPresionTemperatura(double presion, double
temperatura) {

    IF97 tablas=new IF97(UnitSystem.SI);
    PuntoSubenfriado p=new PuntoSubenfriado();

    p.entalpia=tablas.specificEnthalpyPT(presion*100000, temperatura+273.15)/1000;
    p.entropia=tablas.specificEntropyPT(presion*100000, temperatura+273.15)/1000;
    p.presion=presion;
    p.temperatura=temperatura;
    p.volumen=tablas.specificVolumePT(presion*100000.0, temperatura+273.15);

    return p;

    }

    /**
     * Método para obtener un punto subefriado correspondiente a una presion y entropía dadas
     * @param presion
```

```
 * @param entropia
 * @return
 */
public static PuntoSubenfriado getPuntoPresionEntropia(double presion, double entropia) {

IF97 tablas=new IF97(UnitSystem.SI);
PuntoSubenfriado p=new PuntoSubenfriado();

p.entalpia=tablas.specificEnthalpyPS(presion*100000, entropia*1000)/1000;
p.entropia=entropia;
p.presion=presion;
p.temperatura=tablas.temperaturePS(presion*100000, entropia*1000)-273.15;
p.volumen=tablas.specificVolumePS(presion*100000.0, entropia*1000);

return p;

}

/**
 * Método para obtener un punto subefriado correspondiente a una presion y entalpía dadas
 * @param presion
 * @param entalpia
 * @return
 */
public static PuntoSubenfriado getPuntoPresionEntalpia(double presion, double entalpia) {

IF97 tablas=new IF97(UnitSystem.SI);
PuntoSubenfriado p=new PuntoSubenfriado();

p.entalpia=entalpia;
p.entropia=tablas.specificEntropyPH(presion*100000, entalpia*1000)/1000;
p.presion=presion;
p.temperatura=tablas.temperaturePH(presion*100000, entalpia*1000)-273.15;
p.volumen=tablas.specificVolumePH(presion*100000.0, entalpia*1000);

return p;

}
}
```

```
/**
 * Clase que representa y calcula la sucesión de puntos de una trnsformación termodinámica
 * @author El autor de este TFG
 */
public class Transformacion {
ArrayList<PuntoDiagrama> puntos=new ArrayList<PuntoDiagrama>();
PuntoDiagrama primero;
PuntoDiagrama ultimo;


/**
 * Método para generar los puntos que componen una tranformación isoentrópica
 * @param numPuntos
 */
public void generaIsoentropica(int numPuntos) {
puntos.add(primero.clone());
puntos.add(ultimo.clone());
}



/**
 * Método para generar los puntos que componen una tranformación isóbara
 * @param numPuntos
 */
public void generaIsobara(int numPuntos){

double intervalo=ultimo.entropia-primero.entropia;
double salto=intervalo/numPuntos;
double entropia=primero.entropia;

while(entropia<ultimo.entropia) {
PuntoSubenfriado aux=TablasSubenfriado.getPuntoPresionEntropia(primero.presion, entropia);
if(aux!=null) {

PuntoDiagrama p=aux.getPuntoDiagrama();
puntos.add(p);
```

```
}
    entropia+=salto;
}

}

/**
 * Método para pintar una transformación del diagrama TS
 * @param g
 */
public void pintaTemperaturaEntropia(Grafica g) {

Double min=GeneraGraficas.min;
Double max=GeneraGraficas.max;


for(int i=0; i<puntos.size(); i++) {
        PuntoDiagrama aux= puntos.get(i);
        if(aux!=null&&aux.entropia>min&&aux.entropia<max) {
              g.inserta(aux.entropia, aux.temperatura);
        }
      }
}

/**
 * Método para pintar una transformación del diagrama HS
 * @param g
 */
public void pintaEntalpiaEntropia(Grafica g) {

Double min=GeneraGraficas.min;
Double max=GeneraGraficas.max;
      for(int i=0; i<puntos.size(); i++) {
        PuntoDiagrama aux= puntos.get(i);
        if(aux!=null&&aux.entropia>min&&aux.entropia<max) {
              g.inserta(aux.entropia, aux.entalpia);
        }
      }
}
```

```
}



/**
 * Clase que representa la ventana de cnofiguración de la panta de vapor
 * @author El autor de este TFG
 *
 */
public class Ventana1 extends JFrame {

static int alto;
static int ancho;
static int altoVentana;
static int anchoVentana;
static int anchoPanel;
static int altoPanel;
static int xDiagrama;

static final int xDiagramaMin=375;



/**
 * Método constructor de la interfaz
 */
public Ventana1() {

Toolkit t = Toolkit.getDefaultToolkit();
Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
alto=screenSize.height;
ancho=screenSize.width;
altoVentana=(int) (alto*0.95);
anchoVentana=(int) (ancho);

setSize(anchoVentana, altoVentana);
setTitle("Planta de Propulsión");

this.setResizable(false);
```

```
this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


Panel1 p=new Panel1();
add(p);


}









}

class Panel1 extends JPanel implements ActionListener{

int altoImagen;
int anchoImagen;
private Image imagen;


JButton calcular;
JButton TS;
JButton HS;

JTextField PresionCaldera;
JTextField PresionExtraccion1;
JTextField PresionExtraccion2;
JTextField PresionExtraccion3;
JTextField rendimientoCaldera;
JTextField inicio;
JTextField fin;
JTextField temperaturaMin;
JTextField temperaturaMax;
JTextField indiceCarga;
JTextField proporcionFuel;
JTextField rendimientoIsoentropico;
```

```
JTextField rendimientoMecanico;

private JComboBox<String> comboGas;



JLabel calderaLabel;
JLabel ext1;
JLabel ext2;
JLabel ext3;
JLabel rendimientoCalderaLabel;
JLabel centrarGrafica;
JLabel temperaturaMinText;
JLabel temperaturaMaxText;
JLabel indiceCargaText;
JLabel proporcionFuelText;
JLabel rendimientoIsoentropicoText;
JLabel rendimientoMecanicoText;
JLabel terminalGasText;




public Panel1() {

//////////////////////Creo los Botones

this.setLayout(null);
calcular=new JButton("Calcular");
HS=new JButton("Diagrama HS");
TS=new JButton("Diagrama TS");

calcular.setBounds(85, 580, 200, 40);
HS.setBounds(20, 480, 150, 30);
TS.setBounds(20, 520, 150, 30);

add(calcular);
add(HS);
add(TS);
```

```
calcular.addActionListener(this);
HS.addActionListener(this);
TS.addActionListener(this);


//////////////////Creo las cajas de texto
PresionCaldera=new JTextField();
PresionExtraccion1=new JTextField();
PresionExtraccion2=new JTextField();
PresionExtraccion3=new JTextField();
rendimientoCaldera=new JTextField();
inicio=new JTextField();
fin=new JTextField();
temperaturaMin=new JTextField();
temperaturaMax=new JTextField();
indiceCarga=new JTextField();
proporcionFuel=new JTextField();
proporcionFuel=new JTextField();
rendimientoIsoentropico=new JTextField();
rendimientoMecanico=new JTextField();
comboGas=new JComboBox();


PresionCaldera.setBounds(250,20, 100, 20);
PresionExtraccion1.setBounds(250,50, 100, 20);
PresionExtraccion2.setBounds(250,80, 100, 20);
PresionExtraccion3.setBounds(250,110, 100, 20);
rendimientoCaldera.setBounds(250,140, 100, 20);
temperaturaMin.setBounds(250,170, 100, 20);
temperaturaMax.setBounds(250,200, 100, 20);
indiceCarga.setBounds(250,230, 100, 20);
proporcionFuel.setBounds(250,260, 100, 20);
rendimientoIsoentropico.setBounds(250,290, 100, 20);
rendimientoMecanico.setBounds(250,320, 100, 20);
comboGas.setBounds(200,350, 150, 20);


inicio.setBounds(205,510, 50, 20);
fin.setBounds(260,510, 50, 20);
```

```
comboGas.addItem("Australia-NWS");
comboGas.addItem("Australia-Darwin");
comboGas.addItem("Argelia-Skilkda");
comboGas.addItem("Argelia-Bethloua");
comboGas.addItem("Argelia-Arzew");
comboGas.addItem("Brunei");
comboGas.addItem("Egipto-Izku");
comboGas.addItem("Egipto-Damietta");
comboGas.addItem("Guinea Ecuatorial");
comboGas.addItem("Indonesia-Arun");
comboGas.addItem("Indonesia-Badak");
comboGas.addItem("Indonesia-Tangguh");
comboGas.addItem("Libia");
comboGas.addItem("Malasia");
comboGas.addItem("Nigeria");
comboGas.addItem("Noruega");
comboGas.addItem("Oman");
comboGas.addItem("Peru");
comboGas.addItem("Catar");
comboGas.addItem("Russia-Sakhalin");
comboGas.addItem("Trinidad");
comboGas.addItem("USA-Alaska");
comboGas.addItem("Yemen");


add(this.PresionCaldera);
add(this.PresionExtraccion1);
add(this.PresionExtraccion2);
add(this.PresionExtraccion3);
add(this.rendimientoCaldera);
add(this.inicio);
add(fin);
add(this.temperaturaMax);
add(this.temperaturaMin);
add(this.indiceCarga);
add(this.proporcionFuel);
add(this.rendimientoIsoentropico);
add(this.rendimientoMecanico);
```

```
add(comboGas);

this.PresionCaldera.setText("6.0");
this.PresionExtraccion1.setText("2.0");
this.PresionExtraccion2.setText("0.7");
this.PresionExtraccion3.setText("0.15");
this.rendimientoCaldera.setText("80");
this.inicio.setText("0");
this.fin.setText("10");
this.indiceCarga.setText("1");
this.temperaturaMin.setText("25");
this.temperaturaMax.setText("510");
this.proporcionFuel.setText("80");
this.rendimientoMecanico.setText("95");
this.rendimientoIsoentropico.setText("97");

//////////////////Creo los labels

calderaLabel = new JLabel();
ext1 = new JLabel();
ext2 = new JLabel();
ext3 = new JLabel();
rendimientoCalderaLabel = new JLabel();
this.centrarGrafica=new JLabel();
temperaturaMinText=new JLabel();
temperaturaMaxText=new JLabel();;
indiceCargaText=new JLabel();;
proporcionFuelText=new JLabel();;
rendimientoIsoentropicoText=new JLabel();;
rendimientoMecanicoText=new JLabel();;
terminalGasText=new JLabel();

this.calderaLabel.setBounds(20,20, 200, 20);
this.ext1.setBounds(20,50, 200, 20);
this.ext2.setBounds(20,80, 200, 20);
this.ext3.setBounds(20,110, 200, 20);
this.rendimientoCalderaLabel.setBounds(20,140, 200, 20);
temperaturaMinText.setBounds(20,170, 200, 20);
temperaturaMaxText.setBounds(20,200, 200, 20);
indiceCargaText.setBounds(20,230, 200, 20);
```

```
proporcionFuelText.setBounds(20,260, 200, 20);
rendimientoIsoentropicoText.setBounds(20,290, 200, 20);
rendimientoMecanicoText.setBounds(20,320, 200, 20);
terminalGasText.setBounds(20,350, 200, 20);
centrarGrafica.setBounds(190, 480, 200, 20);



this.calderaLabel.setText("Presión de la Caldera          (MPa)");
this.ext1.setText("Presión de la extracción 1   (MPa)");
this.ext2.setText("Presión de la extracción 2   (MPa)");
this.ext3.setText("Presión de la extracción 3   (MPa)");
this.rendimientoCalderaLabel.setText("Rendimiento caldera              (%)");
this.centrarGrafica.setText("Representar intervalo (eje x)");
temperaturaMinText.setText("Temperatura agua del mar     (Cº)");
temperaturaMaxText.setText("Temperatura máxima             (Cº)");
indiceCargaText.setText("Indice de carga");
proporcionFuelText.setText("Proporción Fuel Oil               (%)");
rendimientoIsoentropicoText.setText("Rendimeinto isoentrópico    (%)");
rendimientoMecanicoText.setText("Rendimeinto mecánico         (%)");
terminalGasText.setText("Terminal de Gas");

add(this.calderaLabel);
add(this.ext1);
add(this.ext2);
add(this.ext3);
add(this.rendimientoCalderaLabel);
add(this.centrarGrafica);
add(this.temperaturaMaxText);
add(this.temperaturaMinText);
add(this.indiceCargaText);
add(this.rendimientoIsoentropicoText);
add(this.rendimientoMecanicoText);
add(this.proporcionFuelText);
add(terminalGasText);



}

public void paintComponent(Graphics g) {
```

```
    super.paintComponent(g);

    File FicheroImagen=new File("src\\esquemaPlanta.PNG");
    try {
    imagen=ImageIO.read(FicheroImagen);
    } catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
    }


    altoImagen=imagen.getHeight(null);
    anchoImagen=imagen.getWidth(null);
    int posicionY=0;

    int altoImagenPintada=this.getHeight();
    int anchoImagenPintada=anchoImagen*Ventana1.altoVentana/altoImagen;
    int PosicionXImagenPintada=Ventana1.anchoVentana-anchoImagenPintada;
    if(Ventana1.xDiagramaMin>PosicionXImagenPintada) {
    PosicionXImagenPintada=Ventana1.xDiagramaMin;
    anchoImagenPintada=Ventana1.anchoVentana-PosicionXImagenPintada;

    altoImagenPintada=altoImagenPintada*anchoImagenPintada/(anchoImagen*Ventana1.altoVentana/al
toImagen);
    posicionY=(Ventana1.altoVentana-altoImagenPintada)/3;
    }


    g.drawImage(imagen, PosicionXImagenPintada, posicionY, anchoImagenPintada,
altoImagenPintada, null);

    Ventana1.altoPanel=altoImagenPintada;
    Ventana1.anchoPanel=this.getWidth();
    Ventana1.xDiagrama=PosicionXImagenPintada;

    }

    @Override
    public void actionPerformed(ActionEvent e) {
```

```
Object botonPulsado=e.getSource();
if(botonPulsado==calcular) {
leePuntosInterfaz();
Rankine.calculaPuntos();
Rankine.calcularCaudales();
	Rankine.calcularPotenciaRendimiento();
	Rankine.calculaConsumoCombustible();

	//Creo interfaz de ventana 2
Ventana2 v=new Ventana2();
v.setVisible(true);



}else if(botonPulsado==HS){
leeinicioInterfaz();
leePuntosInterfaz();
Rankine.calculaPuntos();
if((GeneraGraficas.max-GeneraGraficas.min)<2) {
Mensaje m=new Mensaje();
m.escribe("El intervalo seleccionado es demasiado pequeño");
}else {
HiloHS h1=new HiloHS();
h1.start();
}

}else if(botonPulsado==TS) {
leeinicioInterfaz();
leePuntosInterfaz();
Rankine.calculaPuntos();
if((GeneraGraficas.max-GeneraGraficas.min)<1) {
Mensaje m=new Mensaje();
m.escribe("El intervalo seleccionado es emasiado pequeño");
}else {
HiloTS h2=new HiloTS();
h2.start();
}
}
```

```
    }


    /**
     * Método para leer los parámetros de la interfaz
     */
    public void leePuntosInterfaz() {
    Rankine.presionCaldera=10*Double.parseDouble(this.PresionCaldera.getText());
    Rankine.presionExtraccion1=10*Double.parseDouble(this.PresionExtraccion1.getText());
    Rankine.presionExtraccion2=10*Double.parseDouble(this.PresionExtraccion2.getText());
    Rankine.presionExtraccion3=10*Double.parseDouble(this.PresionExtraccion3.getText());
    Rankine.rendimientoCaldera=Double.parseDouble(this.rendimientoCaldera.getText())/100.0;
    Rankine.PC_GAS=leerPCIGas();

    Rankine.temperaturaFocoFrio=Double.parseDouble(this.temperaturaMin.getText());
        Rankine.temperaturaMaxima=Double.parseDouble(this.temperaturaMax.getText());

    Rankine.rendimientoIsoentropico=Double.parseDouble(this.rendimientoIsoentropico.getText())/
100.0;

Rankine.rendimientoIsoentropicoBombas=Double.parseDouble(this.rendimientoIsoentropico.getText()
)/100.0;

    Rankine.rendimientoMecanicoBombas=Double.parseDouble(this.rendimientoMecanico.getText())/10
0.0;

    Rankine.rendimientoMecanicoTurbinas=Double.parseDouble(this.rendimientoMecanico.getText())/
100.0;
    Rankine.indiceCarga=Double.parseDouble(this.indiceCarga.getText());
        Rankine.proporcionFuelOil=Double.parseDouble(this.proporcionFuel.getText())/100.0;

    }

    /**
     * Método para leeer la terminal de gas seleccionada
     * @return
     */
    public double leerPCIGas() {
```

```
String terminal=comboGas.getSelectedItem().toString();
double pci;

 switch (terminal)
        {

    case "Australia-NWS":  pci = 40000;
            break;
    case "Australia-Darwin":  pci = 40000;
            break;
    case "Argelia-Skilkda":  pci = 40000;
            break;
    case "Argelia-Bethloua":  pci = 40000;
            break;
    case "Argelia-Arzew":  pci = 40000;
            break;
    case "Brunei":  pci = 40000;
            break;
    case "Egipto-Izku":  pci = 40000;
            break;
    case "Egipto-Damietta":  pci = 40000;
                break;
    case "Guinea Ecuatorial":  pci = 40000;
                break;
    case "Indonesia-Arun":  pci = 40000;
              break;
    case "Indonesia-Badak":  pci = 40000;
              break;
    case "Indonesia-Tangguh":  pci = 40000;
              break;
    case "Libia":  pci = 40000;
              break;
    case "Malasia":  pci = 40000;
              break;
    case "Nigeria":  pci = 40000;
                 break;
    case "Noruega":  pci = 40000;
            break;
 case "Oman":  pci = 40000;
```

```
            break;
 case "Peru":  pci = 40000;
            break;
 case "Catar":  pci = 40000;
                break;
 case "Russia-Sakhalin":  pci = 40000;
                break;
 case "Trinidad":  pci = 40000;
                break;
 case "USA-Alaska":  pci = 40000;
                break;
 case "Yemen":  pci = 40000;
                break;
 default: pci = 40000;
                break;

        }

return pci;
}

/**
 * Método para leer el rango de representación de la gráfica
 */
public void leeinicioInterfaz() {


String inicio=this.inicio.getText();
String fin=this.fin.getText();
GeneraGraficas.min=Double.parseDouble(inicio);
GeneraGraficas.max=Double.parseDouble(fin);



}
}




/**
```

```
 * Clase que representa la pantalla de resultados del programa
 * @author El autor de este TFG
 */
public class Ventana2 extends JFrame{

static int alto;
static int ancho;
static int altoVentana;
static int anchoVentana;
static int anchoPanel;
static int altoPanel;
static int xDiagrama;



/**
 * Método constructor de la clase
 */
public Ventana2() {

Toolkit t = Toolkit.getDefaultToolkit();
Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
alto=screenSize.height;
ancho=screenSize.width;
altoVentana=(int) (alto*0.95);
anchoVentana=(int) (ancho*2/3);

setSize(anchoVentana, altoVentana);
this.setLocation(ancho/3,0);
setTitle("Planta de Propulsión");



this.setResizable(false);

Panel2 p=new Panel2();
add(p);

}



}
```

```
class Panel2 extends JPanel implements ActionListener{


JTextPane resultados;
JScrollPane scrollPane;
JScrollPane scrollPane2;
JScrollBar barra;
private BufferedImage image;

public Panel2() {
this.setLayout(null);
resultados=new JTextPane();
scrollPane = new JScrollPane(resultados);


resultados.setBounds(0, 0, Ventana2.anchoVentana, Ventana2.altoVentana);
scrollPane.setBounds(0, 0, Ventana2.anchoVentana-15, Ventana2.altoVentana*1/2);
scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
scrollPane.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);



resultados.setAlignmentY(TOP_ALIGNMENT);

String mensaje=Rankine.generaTexto();


    resultados.setMargin(new Insets(20, 20, 20, 20));

resultados.setText(mensaje);
Font font = new Font("Consolas", Font.PLAIN, 13);
resultados.setFont(font);
add(scrollPane);


 try {
          image = ImageIO.read(new File("src\\esquema.PNG"));
       } catch (IOException ex) {
```

```
                // handle exception...
            }


    }


    @Override
    public void actionPerformed(ActionEvent e) {
    // TODO Auto-generated method stub

    }

     @Override
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            g.drawImage(image, 0, Ventana2.altoVentana*1/2, (int) (Ventana2.altoVentana*1.2/2),
(int) (Ventana2.altoVentana*1/2-0.05*Ventana2.altoVentana), null);

        }

    }


    /**
     * Clase que representa la ventana de cnofiguración de la panta de ciclo combinado
     * @author El autor de este TFG
     *
     */
    public class Ventana3 extends JFrame {


    private static final long serialVersionUID = 1L;
    static int alto;
    static int ancho;
    static int altoVentana;
    static int anchoVentana;
    static int anchoPanel;
    static int altoPanel;
    static int xDiagrama;
```

```
static final int xDiagramaMin=375;


/**
 * Método constructor de la clase
 */
public Ventana3() {

Toolkit t = Toolkit.getDefaultToolkit();
Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
alto=screenSize.height;
ancho=screenSize.width;
altoVentana=(int) (alto*0.95);
anchoVentana=(int) (ancho);

setSize(anchoVentana, altoVentana);
setTitle("Planta de Propulsión");

this.setResizable(false);

this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


Panel3 p=new Panel3();
add(p);

}

}

class Panel3 extends JPanel implements ActionListener{

int altoImagen;
int anchoImagen;
private Image imagen;


JButton calcular;
JButton TS;
```

```
JButton HS;


JTextField PresionCaldera;
JTextField PresionExtraccion1;
JTextField PresionExtraccion2;
JTextField PresionExtraccion3;
JTextField rendimientoCaldera;
JTextField inicio;
JTextField fin;
JTextField temperaturaMin;
JTextField temperaturaMax;
JTextField indiceCarga;
JTextField proporcionFuel;
JTextField rendimientoIsoentropico;
JTextField rendimientoMecanico;
private JComboBox<String> comboGas;


JLabel calderaLabel;
JLabel ext1;
JLabel ext2;
JLabel ext3;
JLabel rendimientoCalderaLabel;
JLabel centrarGrafica;
JLabel temperaturaMinText;
JLabel temperaturaMaxText;
JLabel indiceCargaText;
JLabel proporcionFuelText;
JLabel rendimientoIsoentropicoText;
JLabel rendimientoMecanicoText;
JLabel terminalGasText;


public Panel3() {


//////////////////////Creo los Botones

this.setLayout(null);
```

```
calcular=new JButton("Calcular");
HS=new JButton("Diagrama HS");
TS=new JButton("Diagrama TS");

calcular.setBounds(85, 580, 200, 40);
HS.setBounds(20, 480, 150, 30);
TS.setBounds(20, 520, 150, 30);

add(calcular);
add(HS);
add(TS);



calcular.addActionListener(this);
HS.addActionListener(this);
TS.addActionListener(this);


//////////////////Creo las cajas de texto
PresionCaldera=new JTextField();
PresionExtraccion1=new JTextField();
PresionExtraccion2=new JTextField();
PresionExtraccion3=new JTextField();
rendimientoCaldera=new JTextField();
inicio=new JTextField();
fin=new JTextField();
temperaturaMin=new JTextField();
temperaturaMax=new JTextField();
indiceCarga=new JTextField();
proporcionFuel=new JTextField();
proporcionFuel=new JTextField();
rendimientoIsoentropico=new JTextField();
rendimientoMecanico=new JTextField();
comboGas=new JComboBox();


PresionCaldera.setBounds(250,20, 100, 20);
PresionExtraccion1.setBounds(250,50, 100, 20);
PresionExtraccion2.setBounds(250,80, 100, 20);
PresionExtraccion3.setBounds(250,110, 100, 20);
```

```
rendimientoCaldera.setBounds(250,140, 100, 20);
temperaturaMin.setBounds(250,170, 100, 20);
temperaturaMax.setBounds(250,200, 100, 20);
indiceCarga.setBounds(250,230, 100, 20);
proporcionFuel.setBounds(250,260, 100, 20);
rendimientoIsoentropico.setBounds(250,290, 100, 20);
rendimientoMecanico.setBounds(250,320, 100, 20);
comboGas.setBounds(200,350, 150, 20);


inicio.setBounds(205,510, 50, 20);
fin.setBounds(260,510, 50, 20);


comboGas.addItem("Australia-NWS");
comboGas.addItem("Australia-Darwin");
comboGas.addItem("Argelia-Skilkda");
comboGas.addItem("Argelia-Bethloua");
comboGas.addItem("Argelia-Arzew");
comboGas.addItem("Brunei");
comboGas.addItem("Egipto-Izku");
comboGas.addItem("Egipto-Damietta");
comboGas.addItem("Guinea Ecuatorial");
comboGas.addItem("Indonesia-Arun");
comboGas.addItem("Indonesia-Badak");
comboGas.addItem("Indonesia-Tangguh");
comboGas.addItem("Libia");
comboGas.addItem("Malasia");
comboGas.addItem("Nigeria");
comboGas.addItem("Noruega");
comboGas.addItem("Oman");
comboGas.addItem("Peru");
comboGas.addItem("Catar");
comboGas.addItem("Russia-Sakhalin");
comboGas.addItem("Trinidad");
comboGas.addItem("USA-Alaska");
comboGas.addItem("Yemen");


add(this.PresionCaldera);
```

```
add(this.PresionExtraccion1);
add(this.PresionExtraccion2);
add(this.PresionExtraccion3);
add(this.rendimientoCaldera);
add(this.inicio);
add(fin);
add(this.temperaturaMax);
add(this.temperaturaMin);
add(this.indiceCarga);
add(this.proporcionFuel);
add(this.rendimientoIsoentropico);
add(this.rendimientoMecanico);
add(comboGas);

this.PresionCaldera.setText("6.0");
this.PresionExtraccion1.setText("2.0");
this.PresionExtraccion2.setText("0.7");
this.PresionExtraccion3.setText("0.15");
this.rendimientoCaldera.setText("80");
this.inicio.setText("0");
this.fin.setText("10");
this.indiceCarga.setText("1");
this.temperaturaMin.setText("25");
this.temperaturaMax.setText("510");
this.proporcionFuel.setText("80");
this.rendimientoMecanico.setText("95");
this.rendimientoIsoentropico.setText("97");

//////////////////Creo los labels

calderaLabel = new JLabel();
ext1 = new JLabel();
ext2 = new JLabel();
ext3 = new JLabel();
rendimientoCalderaLabel = new JLabel();
this.centrarGrafica=new JLabel();
temperaturaMinText=new JLabel();
temperaturaMaxText=new JLabel();;
indiceCargaText=new JLabel();;
proporcionFuelText=new JLabel();;
```

```
rendimientoIsoentropicoText=new JLabel();;
rendimientoMecanicoText=new JLabel();;
terminalGasText=new JLabel();

this.calderaLabel.setBounds(20,20, 200, 20);
this.ext1.setBounds(20,50, 200, 20);
this.ext2.setBounds(20,80, 200, 20);
this.ext3.setBounds(20,110, 200, 20);
this.rendimientoCalderaLabel.setBounds(20,140, 200, 20);
temperaturaMinText.setBounds(20,170, 200, 20);
temperaturaMaxText.setBounds(20,200, 200, 20);
indiceCargaText.setBounds(20,230, 200, 20);
proporcionFuelText.setBounds(20,260, 200, 20);
rendimientoIsoentropicoText.setBounds(20,290, 200, 20);
rendimientoMecanicoText.setBounds(20,320, 200, 20);
terminalGasText.setBounds(20,350, 200, 20);
centrarGrafica.setBounds(190, 480, 200, 20);



this.calderaLabel.setText("P. Caldera Recuperación      (MPa)");
this.ext1.setText("Presión de la extracción 1   (MPa)");
this.ext2.setText("Presión de la extracción 2   (MPa)");
this.ext3.setText("Presión de la extracción 3   (MPa)");
this.rendimientoCalderaLabel.setText("Rendimiento HRSG                  (%)");
this.centrarGrafica.setText("Representar intervalo (eje x)");
temperaturaMinText.setText("Temperatura agua del mar     (Cº)");
temperaturaMaxText.setText("Temperatura máxima              (Cº)");
indiceCargaText.setText("Indice de carga");
proporcionFuelText.setText("Proporción Fuel Oil                (%)");
rendimientoIsoentropicoText.setText("Rendimeinto isoentrópico    (%)");
rendimientoMecanicoText.setText("Rendimeinto mecánico         (%)");
terminalGasText.setText("Terminal de Gas");

add(this.calderaLabel);
add(this.ext1);
add(this.ext2);
add(this.ext3);
add(this.rendimientoCalderaLabel);
add(this.centrarGrafica);
```

```
    add(this.temperaturaMaxText);
    add(this.temperaturaMinText);
    add(this.indiceCargaText);
    add(this.rendimientoIsoentropicoText);
    add(this.rendimientoMecanicoText);
    add(this.proporcionFuelText);
    add(terminalGasText);




    }

    public void paintComponent(Graphics g) {

    super.paintComponent(g);

    File FicheroImagen=new File("src\\esquemaPlantaCombinado.PNG");
    try {
    imagen=ImageIO.read(FicheroImagen);
    } catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
    }



    altoImagen=imagen.getHeight(null);
    anchoImagen=imagen.getWidth(null);
    int posicionY=0;

    int altoImagenPintada=this.getHeight();
    int anchoImagenPintada=anchoImagen*Ventana3.altoVentana/altoImagen;
    int PosicionXImagenPintada=Ventana3.anchoVentana-anchoImagenPintada;
    if(Ventana3.xDiagramaMin>PosicionXImagenPintada) {
    PosicionXImagenPintada=Ventana3.xDiagramaMin;
    anchoImagenPintada=Ventana3.anchoVentana-PosicionXImagenPintada;

    altoImagenPintada=altoImagenPintada*anchoImagenPintada/(anchoImagen*Ventana3.altoVentana/a
ltoImagen);
    posicionY=(Ventana3.altoVentana-altoImagenPintada)/3;
    }
```

```
    g.drawImage(imagen, PosicionXImagenPintada, posicionY, anchoImagenPintada,
altoImagenPintada, null);

    Ventana3.altoPanel=altoImagenPintada;
    Ventana3.anchoPanel=this.getWidth();
    Ventana3.xDiagrama=PosicionXImagenPintada;

    }

    @Override
    public void actionPerformed(ActionEvent e) {

    Object botonPulsado=e.getSource();
    if(botonPulsado==calcular) {
    leePuntosInterfaz();
    Brayton.calcular();
    Brayton.calculaPoenciasRendimientos();
    Rankine.calculaPuntos();
    Rankine.calcularCaudales();
           Rankine.calcularPotenciaRendimiento();
           Rankine.calculaConsumoCombustible();

           //Creo interfaz de ventana 2
    Ventana4 v=new Ventana4();
    v.setVisible(true);


    }else if(botonPulsado==HS){
    leeinicioInterfaz();
    leePuntosInterfaz();
    Rankine.calculaPuntos();
    if((GeneraGraficas.max-GeneraGraficas.min)<2) {
    Mensaje m=new Mensaje();
    m.escribe("El intervalo seleccionado es demasiado pequeño");
    }else {
    HiloHS h1=new HiloHS();
    h1.start();
    }
```

```
    }else if(botonPulsado==TS) {
    leeinicioInterfaz();
    leePuntosInterfaz();
    Rankine.calculaPuntos();
    if((GeneraGraficas.max-GeneraGraficas.min)<1) {
    Mensaje m=new Mensaje();
    m.escribe("El intervalo seleccionado es emasiado pequeño");
    }else {

    Brayton.calcular();
                Rankine.calculaPuntos();
    HiloTSBrayton h2=new HiloTSBrayton();
    h2.start();
    }
    }




    }

    /**
     * Método para leer los parámetros de la interfaz
     */
    public void leePuntosInterfaz() {
    Rankine.presionCaldera=10*Double.parseDouble(this.PresionCaldera.getText());
    Rankine.presionExtraccion1=10*Double.parseDouble(this.PresionExtraccion1.getText());
    Rankine.presionExtraccion2=10*Double.parseDouble(this.PresionExtraccion2.getText());
    Rankine.presionExtraccion3=10*Double.parseDouble(this.PresionExtraccion3.getText());
    Rankine.rendimientoCaldera=Double.parseDouble(this.rendimientoCaldera.getText())/100.0;
    Rankine.PC_GAS=leerPCIGas();

    Rankine.temperaturaFocoFrio=Double.parseDouble(this.temperaturaMin.getText());
        Rankine.temperaturaMaxima=Double.parseDouble(this.temperaturaMax.getText());

    Rankine.rendimientoIsoentropico=Double.parseDouble(this.rendimientoIsoentropico.getText())
/100.0;
```

```
Rankine.rendimientoIsoentropicoBombas=Double.parseDouble(this.rendimientoIsoentropico.getText(
))/100.0;

    Rankine.rendimientoMecanicoBombas=Double.parseDouble(this.rendimientoMecanico.getText())/1
00.0;

    Rankine.rendimientoMecanicoTurbinas=Double.parseDouble(this.rendimientoMecanico.getText())
/100.0;
    Rankine.indiceCarga=Double.parseDouble(this.indiceCarga.getText());
        Rankine.proporcionFuelOil=Double.parseDouble(this.proporcionFuel.getText())/100.0;

    }


    /**
     * Método para leer la terminal de gas seleccionada
     * @return
     */
    public double leerPCIGas() {

    String terminal=comboGas.getSelectedItem().toString();
    double pci;


     switch (terminal)
            {

       case "Australia-NWS":  pci = 40000;
                break;
       case "Australia-Darwin":  pci = 40000;
                break;
       case "Argelia-Skilkda":  pci = 40000;
                break;
       case "Argelia-Bethloua":  pci = 40000;
                break;
       case "Argelia-Arzew":  pci = 40000;
                break;
       case "Brunei":  pci = 40000;
                break;
```

```
 case "Egipto-Izku":  pci = 40000;
          break;
 case "Egipto-Damietta":  pci = 40000;
          break;
 case "Guinea Ecuatorial":  pci = 40000;
          break;
 case "Indonesia-Arun":  pci = 40000;
          break;
 case "Indonesia-Badak":  pci = 40000;
          break;
 case "Indonesia-Tangguh":  pci = 40000;
          break;
 case "Libia":  pci = 40000;
          break;
 case "Malasia":  pci = 40000;
          break;
 case "Nigeria":  pci = 40000;
          break;
 case "Noruega":  pci = 40000;
          break;
 case "Oman":  pci = 40000;
          break;
 case "Peru":  pci = 40000;
          break;
 case "Catar":  pci = 40000;
          break;
 case "Russia-Sakhalin":  pci = 40000;
          break;
 case "Trinidad":  pci = 40000;
          break;
  case "USA-Alaska":  pci = 40000;
          break;
  case "Yemen":  pci = 40000;
          break;
  default: pci = 40000;
          break;

        }

return pci;
```

```
}

/**
 * Método para leer el rango de representación de la gráfica
 */
public void leeinicioInterfaz() {


String inicio=this.inicio.getText();
String fin=this.fin.getText();
GeneraGraficas.min=Double.parseDouble(inicio);
GeneraGraficas.max=Double.parseDouble(fin);



}
}


/**
 * Clase que representa la pantalla de resultados del programa
 * @author El autor de este TFG
 *
 */
public class Ventana4 extends JFrame{

static int alto;
static int ancho;
static int altoVentana;
static int anchoVentana;
static int anchoPanel;
static int altoPanel;
static int xDiagrama;


/**
 * Método constructor de la clase
 */
public Ventana4() {

Toolkit t = Toolkit.getDefaultToolkit();
```

```
Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
alto=screenSize.height;
ancho=screenSize.width;
altoVentana=(int) (alto*0.95);
anchoVentana=(int) (ancho*2/3);

setSize(anchoVentana, altoVentana);
this.setLocation(ancho/3,0);
setTitle("Planta de Propulsión");


this.setResizable(false);

Panel4 p=new Panel4();
add(p);

}


}


class Panel4 extends JPanel implements ActionListener{



JTextPane resultados;
JScrollPane scrollPane;
JScrollPane scrollPane2;
JScrollBar barra;
private BufferedImage image;

public Panel4() {
this.setLayout(null);
resultados=new JTextPane();
scrollPane = new JScrollPane(resultados);


resultados.setBounds(0, 0, Ventana4.anchoVentana, Ventana4.altoVentana);
scrollPane.setBounds(0, 0, Ventana4.anchoVentana-15, Ventana4.altoVentana*1/2);
```

```
    scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
    scrollPane.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);



    resultados.setAlignmentY(TOP_ALIGNMENT);

    String mensaje=Brayton.generaTexto()+Rankine.generaTexto();


        resultados.setMargin(new Insets(20, 20, 20, 20));

    resultados.setText(mensaje);
    Font font = new Font("Consolas", Font.PLAIN, 13);
    resultados.setFont(font);
    add(scrollPane);


     try {
              image = ImageIO.read(new File("src\\esquemaCombinado.PNG"));
           } catch (IOException ex) {
                // handle exception...
           }

    }


    @Override
    public void actionPerformed(ActionEvent e) {
    // TODO Auto-generated method stub

    }

     @Override
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            g.drawImage(image, 0, Ventana4.altoVentana*1/2, (int) (Ventana4.anchoVentana),
(int) (Ventana4.altoVentana*1/2-0.05*Ventana4.altoVentana), null);

        }
```

```
}


/**
 * Clase que representa la ventana de inicio del programa
 * @author El autor de este TFG
 *
 */
public class VentanaInicio extends JFrame{

public VentanaInicio() {

Toolkit t = Toolkit.getDefaultToolkit();
Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
int alto=screenSize.height;
int ancho=screenSize.width;
this.setBounds(ancho/2-150, alto/2-50, 300, 100);


setTitle("Planta de Propulsión");

this.setResizable(false);


PanelInicio p=new PanelInicio();
p.v=this;
add(p);
}
}

class PanelInicio extends JPanel implements ActionListener{

JButton plantaVapor;
JButton cicloCombinado;
JLabel seleccion;

VentanaInicio v;
```

```
public PanelInicio() {
seleccion=new JLabel("Seleccione un modo de programa");
add(seleccion);

plantaVapor=new JButton("Planta de Vapor");
cicloCombinado=new JButton("Ciclo combinado");
add(plantaVapor);
add(cicloCombinado);
plantaVapor.addActionListener(this);
cicloCombinado.addActionListener(this);



}


@Override
public void actionPerformed(ActionEvent e) {
Object botonPulsado=e.getSource();
if(botonPulsado==plantaVapor) {

Rankine.combinado=false;
Ventana1 v=new Ventana1();
v.setVisible(true);
            Rankine.calculaPuntos();
            Rankine.calcularCaudales();
            Rankine.calcularPotenciaRendimiento();
            Rankine.calculaConsumoCombustible();
            this.v.setVisible(false);


}else {

Rankine.combinado=true;
Ventana3 v=new Ventana3();
v.setVisible(true);
            this.v.setVisible(false);

}

}
```

```
public void paintComponent(Graphics g) {

super.paintComponent(g);
}

}
```

PLIEGO DE CONDICIONES

1. OBJETIVO

El objetivo del Pliego de Condiciones, el cual se muestra a continuación, es acotar los requisitos y necesidades a satisfacer durante la ejecución del proyecto de desarrollo del software de cálculo, además de ajustarse a las condiciones técnicas y controlar la calidad del código producido. El proyecto consiste en la construcción de un software para el caculo de parámetros operativos y de funcionamiento relativos a la planta propulsora de un buque LNG, con el fin de simplificar el estudio del funcionamiento de estas instalaciones, así como la repercusión que los cambios en sus variables de funcionamiento tienen entre sí. Se contemplarán dos posibles plantas propulsoras, siendo la primera de ellas una planta propulsora convencional de vapor y la segunda una planta de ciclo combinado.

1.2 CONDICIONES GENERALES

Seguridad en el trabajo

Se habrá de realizar un Plan en materia de seguridad y salud, que cumpla la siguiente normativa:

- Ordenanza General de Seguridad e Higiene en el Trabajo, aprobada según el Decreto del 11 de Marzo de 1971.

Códigos y normas

Además de las condiciones mencionadas en este escrito, se deberá cumplir la Constitución Española del 27/12/78, donde se tiene:

- Ley 8/1980 del 10 de Marzo de 1980, Estatuto de los trabajadores:

- art. 6: Trabajo de menores

- art. 19: Seguridad e Higiene en el Trabajo
- art. 34: Jornada laboral

Gestión medioambiental

Todas las obras del proyecto se ejecutarán garantizando el cumplimiento de la legislación medioambiental aplicable.

Condiciones para la ejecución del proyecto

La contratada para el desarrollo del software está obligada al cumplimiento de la Reglamentación del Trabajo correspondiente, la contratación del Seguro Obligatorio, Subsidio Familiar y de Vejez, Seguro de Enfermedad y todas aquellas reglamentaciones de carácter social vigentes.

Sociedad de clasificación

El Proyecto se desarrollará y ejecutará de acuerdo con lo dispuesto por la sociedad de clasificación DNV, realizando un cumplimiento estricto de los requisitos establecidos en las reglas para la clasificación [2], especialmente atendiendo a la parte 4 “Sistemas y componentes” en su capítulo 9, “sistemas de control y de monitorización”.

2. CONDICIONES TÉCNICAS

2.1 REQUISITOS FUNCIONALES DEL PROGRAMA

El programa de cálculo desarrollado debe cumplir con una serie de requisitos de funcionamiento, debiendo estos ser respetados e implementados en la aplicación por parte del DESARROLLADOR.

2.1.1 ESCENARIOS DE FUNCIONAMIENTO

El programa permitirá realizar el cálculo de los parámetros operativos de dos tipos de planta propulsora distintos, por una parte, una planta propulsora de vapor y por otra una planta de ciclo combinado. Cada una de estas plantas propulsoras será modelada en el programa de acuerdo a lo descrito en el documento de memoria, en el cual se realiza una descripción detallada de las mismas.

El programa permitirá la selección de planta propulsora mediante una ventana de inicio en la que el usuario podrá seleccionar una de las dos opciones.

2.1.2 PARÁMETROS DE PROGRAMA

El programa de cálculo tomará como parámetros aportados por el usuario los siguientes datos:

-Presión de la caldera

-Presión de la primera extracción

-Presión de la segunda extracción

-Presión de la tercera extracción

-Rendimiento de la caldera

-Temperatura del agua de mar

-Temperatura máxima

-Índice de carga

-Proporción de fuel oil

-Rendimiento Isoentrópico

-Rendimiento mecánico

-Terminal de origen del gas [8]

En el caso de la planta propulsora de ciclo combinado se suprimirá los parámetros “proporción de fuel oil” y “rendimiento de la caldera” y se añadirán los siguientes:

-Rendimiento de la caldera de recuperación

-Temperatura ambiente

Por otra parte, el parámetro “temperatura máxima” se considerará que hace ahora referencia a la temperatura máxima alcanzada en el ciclo Brayton a la entrada de la turbina de gas, mientras que la temperatura máxima del clico Rankine se fijará ahora a partir de la temperatura de los gases de escape de la turbina de gas y el rendimiento de la caldera de recuperación.

2.1.3 PANTALLA DE RESULTADOS

El programa de cálculo, en su pantalla de configuración dispondrá de un botón con el texto “calcular” para realizar el cálculo de la planta a partir de los parámetros introducidos por el usuario y explicados ya en el apartado anterior.

Al pulsar dicho botón el programa realizará todos los cálculos y operaciones necesarias para determinar el rendimiento, potencia neta de la planta y consumo de combustible de acuerdo a lo indicado en el documento ANEXO: CÁLCULOS del presente proyecto.

Inmediatamente después de realizado el cálculo de la planta, el programa abrirá automáticamente una nueva ventana (pantalla de resultados) donde se mostrarán los siguientes valores y en el siguiente orden:

-Temperatura, presión, entalpía específica, entropía específica y volumen específico de cada punto del diagrama Rankine.

-Caudales másicos de cada extracción, caudal total en la caldera y caudales másicos en la turbobomba y el turbogenerador.

-Calor de la combustión

-Calor aportado por la caldera

-Potencia de la turbina

-Potencia en el eje

-Potencia absorbida por la turbobomba

-Potencia absorbida por el turbogenerador.

-Rendimiento neto de la planta

-Consumo de fuel oil

-Consumo de gas natural

En el caso de la planta de ciclo combinado, se mostrarán además los siguientes:

-temperatura y presión de cada punto del diagrama de ciclo Brayton.

-Potencia de la turbina

-Potencia en el eje

-Rendimiento

-Calor cedido al foco frío (gases de escape)

-Caudal másico

Conjuntamente con los valores anteriores se mostrará en la misma ventana un esquema simplificado con el ciclo Brayton y el ciclo Rankine empleados y una leyenda identificativa de todos sus puntos.

2.1.4 GENERACIÓN DE DIAGRAMAS

El software debe incluir una función para generar y representar los diagramas Entalpía-Entropía, y Temperatura-Entropía correspondientes a la planta de propulsión seleccionada por el usuario y representándolos de acuerdo a los parámetros de configuración establecidos por el mismo.

En el caso de la planta de ciclo combinado únicamente será necesario la representación el diagrama Temperatura-Entropía, representándose conjuntamente en el mismo el ciclo Rankine.

La aplicación permitirá al usuario seleccionar un rango de representación para el diagrama permitiendo centrar la representación del eje de abscisas en secciones concretas y acotadas del mismo.

2.2 REQUISITOS NO FUNCIONALES

Además de los requisitos relativos a la funcionalidad del programa, ya enunciados, siguiendo los criterios habitualmente aplicados a la ingeniería de software, a continuación, se indican de forma independiente los requisitos y condiciones no funcionales que el programa deberá cumplir, y por tanto el DESARROLLADOR deberá aplicar en la construcción del mismo.

2.2.1 TECNOLOGÍAS Y LENGUAJE DE PROGRAMACIÓN

El software será desarrollado íntegramente en el lenguaje de programación Java, en su versión 8 [12], teniendo el programa resultante la forma de aplicación de escritorio.

Para el desarrollo de las interfaces gráficas de la aplicación se hará uso de la librería Swing. Por otra parte, en lo referente a la representación de diagramas, emplearemos el paquete Fundamentos [9], desarrollado por los profesores de la Universidad de Cantabria, Michael González Harbour y Mariano Benito Hoz.

2.2.2 ESTRUCTURA DE LA INTERFAZ

La interfaz del programa se dividirá en varias ventanas entre las cuales el usuario podrá navegar dando acceso unas a otras. A continuación, se enumera y describe cada una de ellas.

2.2.2.1 VENTANA DE INICIO

Esta ventana será la primera en abrirse cuando el usuario inicie el programa. Será de un tamaño reducido, entendiéndose como tal el mínimo necesario para albergar 2 botones que permitirán al usuario seleccionar el tipo de planta propulsora, tal como ya se ha explicado en el apartado de requisitos funciónales.

La disposición de los botones será en horizontal y tendrán el texto “Planta de vapor” y “Ciclo combinado” respectivamente. Sobre los dos botones se mostrará una etiqueta de texto con el mensaje “Seleccione un modo de programa”.

2.2.2.2 VENTANA DE CONFIGURACIÓN

La ventana de configuración será aquella en la que el usuario realice la configuración de la planta e introducción de los parámetros necesarios para el cálculo de la mima.

Esta ventana tendrá un tamaño correspondiente a la resolución de la pantalla del equipo en el que se ejecute.

Los dos tercios de la anchura de la ventana situados más a la derecha de la pantalla se emplearán para mostrar al usuario un esquema de la planta propulsora en el que se muestren todos sus elementos de acuerdo a lo indicado en el apartado ANEXO: PLANOS

El tercio de la anchura de la ventana de configuración situado más a la izquierda se destinará a los parámetros de configuración del programa, siendo la forma de introducirlos la de una caja de texto excepto en el caso de las terminales de Gas [8] que se hará mediante un combo box.

Inmediatamente debajo de los parámetros de configuración se mostrarán los botones para la generación de diagramas y el botón de calcular, así como dos cajas de texto para que el usuario indique el rango del diagrama sobre el que quiere centrar la representación.

2.2.2.3 VENTANA DE RESULTADOS

Esta ventana será aquella en la que se muestren los resultados del cálculo, siguiendo el proceso que se ha indicado en el apartado de requisitos funcionales.

En lo referente al tamaño de la ventana, será tal, que se superpondrá al esquema de la planta mostrado en la ventana de configuración situándose la venta de resultados encima de esta en el momento que se genere.

Los resultados se mostrarán en los dos tercios superiores de la ventana haciendo uso de un área de texto que pueda desplazarse mediante una barra de scroll.

El tercio inferior de la ventana se destinará a mostrar el esquema del ciclo Rankine junto con su leyenda de puntos, y en su caso, si se trata de la planta de ciclo combinado, también se mostrará el ciclo Brayton, situándose este a la izquierda del ciclo Rankine.

2.2.2.4 VENTANA DE REPRESENTACIÓN DE DIAGRAMA

Esta ventana será aquella en la que se represente el diagrama seleccionado por el usuario. Su estructura será la que viene determinada por defecto en el paquete Fundamentos [9].

En el caso del ciclo Rankine se representará el diagrama seleccionado (T-S o H-S), mostrándose el ciclo completo junto con todas sus extracciones en color negro, conjuntamente con la curva de saturación en color azul.

En el caso del ciclo Brayton, solo será incluido en el diagrama Temperatura-Entropía.

Los procesos isoentrópicos ideales serán representados en color rojo.

La representación del diagrama quedará acotada y restringida al intervalo indicado por el usuario en la ventana de configuración.

2.3 PRUEBAS DE FUNCIONAMIENTO DEL PROGRAMA

Todas y cada una de las funciones que componen el código deberá ser comprobada al menos mediante ensayos de caja negra implementados mediante el desarrollo de tests específicos para la comprobación del programa. Esto se hará a fin de detectar posibles errores y deficiencias de código desarrollado que puedan desembocar en un mal funcionamiento del

mismo o un fallo no esperado, asegurando así en la medida de lo posible que el comportamiento del programa sea correcto independientemente de cuales hayan sido los parámetros de entrada seleccionados por el usuario.

3. CONDICIONES CONTRACTUALES

3.1 INTRODUCCIÓN Y BASES FUNDAMENTALES

Se ha elaborado una oferta comercial para este proyecto, cuyo objetivo es valorar y establecer las condiciones económicas para el desarrollo e implantación de una aplicación informática para el cálculo de los parámetros operativos y de funcionamiento de la planta propulsora de un buque LNG. De aquí en adelante se denominará CLIENTE a la parte contratante que solicita el desarrollo de dicho software. Por otra parte, se denominará como DESARROLLADOR a la parte contratante encargada del desarrollo del software. La oferta se ha realizado en base a la documentación disponible en el momento de dicha oferta y forma parte de la documentación contractual, siguiendo el siguiente orden de prioridades:

1º La carta de intención del CLIENTE

2º La oferta del DESARROLLADOR

3º La petición de oferta de la CLIENTE Esta propuesta será válida siempre que se firme un contrato que resulte satisfactorio para ambas partes.

3.2 ALCANCE DEL SERVICIO

El alcance del trabajo y los servicios llevados a cabo por el DESARROLLADOR comprenderán lo descrito en el presente documento. No se incluyen en la propuesta, ni actualizaciones futuras, ni mantenimiento de software ni de los equipos informáticos una vez transcurrido el periodo de

garantía del producto. Una vez adjudicado el contrato, el CLIENTE podrá incluir en la oferta los repuestos y el contrato de mantenimiento de la aplicación, con objeto de conseguir las mejores condiciones para el CLIENTE.

3.3 PRECIO DEL PROYECTO

El precio total correspondiente al suministro asciende a 11558,77 €, con IVA excluido. Dicho precio no incluye ningún costo de financiación durante el desarrollo e implantación de software y es válido únicamente para la contratación de la totalidad del proyecto.

Plazo de validez de la oferta

La oferta es válida durante un periodo de cuatro meses. El precio ofertado es fijo durante dicho plazo y transcurrido el periodo, el incremento del precio será de 0.4 % por mes.

Condiciones del precio

El precio incluye: costes de ingeniería, material y mano de obra, instalación, montaje, pruebas, puesta en servicio y documentación del software y los equipos que cubre el proyecto. No se incluyen en el precio el IVA, que será repercutido al CLIENTE al tipo aplicable en el momento de facturación, así como los visados de proyecto, tasas municipales y cualquier otro impuesto de las Administraciones locales, autonómicas y central que se requieran para la tramitación de permisos y legalizaciones. No se revisará el precio por cambios en el precio de mano de obra y equipos, salvo retraso importante en el plazo de entrega por causas no imputables al DESARROLLADOR. Pero cualquier variación por causas no imputables exclusivamente a éste, se facturará aparte, previo acuerdo de los precios con el CLIENTE.

Condiciones de facturación

Las condiciones de pago acordadas serán las siguientes:

- 25 % en el pedido:
- 75 % a la puesta en marcha e implantación el software.

El CLIENTE efectuará los pagos en un plazo de 30 días desde la fecha de facturación mediante transferencia bancaria a la entidad que indique el DESARROLLADOR. Cualquier retraso en el pago, supondrá un interés del 1% mensual.

3.4 EJECUCIÓN DEL PROYECTO

Plazos

La aplicación será entregada al CLIENTE e implantada en sus equipos en el plazo de 5 meses a partir de la firma del contrato.

El software se considerará entregado, a efectos de pago, si ha sido implantado en los equipos del CLIENTE y ha superado una revisión a cargo de personal autorizado por el CLIENTE.

Recepción provisional

El programa se pone a prueba durante 4 semanas en las que el cliente hará uso del mismo a fin de poder estudiar y comprobar con detalle el correcto funcionamiento y desempeño de las funciones acordadas por parte del programa. Si en ese periodo se produce algún fallo atribuible al DESARROLLADOR, deberá empezarse de nuevo con las 4 semanas sin interrupción. El responsable del software ante fallos de ejecución del programa u otras causas será el DESARROLLADOR, hasta la firma del acta de Recepción Provisional.

Superadas las pruebas, se realiza el traspaso de titularizad del software al CLIENTE.

Retraso en pruebas

Si por motivos ajenos al DESARROLLADOR, no se puede realizar cualquiera de las pruebas cuando se informa que la instalación está disponible para ello, se considerará que se han cumplido los plazos. El DESARROLLADOR deberá pasar las pruebas cuando el software esté implantado en los equipos del CLIENTE, pero transcurridos seis meses desde la notificación inicial, esta obligación cesará.

Recepción definitiva

6 meses después de la firma del Acta de Recepción Provisional, se producirá de forma automática la Recepción Definitiva. Si no se observan anomalías, no será necesario realizar comprobaciones. En caso contrario, el CLIENTE informará de cuáles son los defectos y se volverán a realizar las pruebas. Finalmente, se firmará el Acta de Recepción Definitiva, entregando total y definitivamente el software junto con su código y licencia al CLIENTE.

3.5 GARANTÍAS

El DESARROLLADOR garantiza el producto desarrollado en los siguientes aspectos:

Garantías de diseño, construcción y mantenibilidad

El DESARROLLADOR garantiza que el software y los equipos informáticos se han diseñado para atender las condiciones específicas del proyecto, en base a las normas de la Comunidad Europea. Asimismo, durante el periodo de garantía, el DESARROLLADOR asumirá los costes por sustitución y/o reparación de equipos y accesorios defectuosos. Los componentes sustituidos dispondrán del total del periodo de garantía desde la fecha de sustitución. El periodo de garantía se extenderá por 1 año desde la fecha de Recepción Provisional.

Garantías de disponibilidad

El sistema está previsto para un funcionamiento de manera continuada durante todo el año. El DESARROLLADOR deberá establecer un valor de garantía para la disponibilidad y fiabilidad. También indicará los requisitos de tiempo necesarios para el mantenimiento programado de la aplicación. La supervisión solo incluirá posibles fallos de los que sea responsable el DESARROLLADOR.

Garantías de soporte

La garantía sobre el suministro comporta el soporte del ESARROLLADOR al CLIENTE para asegurar la máxima operatividad del sistema, atendiendo las incidencias que ocurran durante su operación, vía telefónica o presencialmente, según lo que se requiera en cada caso. Una vez superado el período de garantía, el DESARROLLADOR facturará al CLIENTE lo que corresponda la reparación.

Garantías de ejecución

El DESARROLLADOR presentará al comienzo del trabajo un plan de ejecución de proyecto de desarrollo del software, en intervalos mensuales, donde figurarán los caminos críticos y los hitos de obligado cumplimiento.

Garantía financiera

El DESARROLLADOR presentará un aval bancario por valor del 25% del importe del contrato, a la recepción del pedido, y que será emitido por una entidad de solvencia reconocida. Estará en vigor hasta la fecha de inicio de la operación del software.

3.6 PENALIZACIONES

Penalizaciones sobre el plazo de entrega

En caso de que el DESARROLLADOR incumpliera el plazo final fijado para la Recepción Provisional del software, por causas imputables al mismo, tendrá una penalización del 2% del precio total del suministro por cada semana incumplida. Esta penalización queda limitada a un máximo del 10% del importe total del proyecto de desarrollo del software.

Penalizaciones por disponibilidad del suministro

Si el coeficiente de disponibilidad del software no alcanzara el mínimo establecido por el DESARROLLADOR se aplicaría una penalización del 2% sobre el precio total del mismo.

3.7 MODIFICACIONES

El DESARROLLADOR deberá traer una oferta completa y detallada en caso de modificación de cualquier parte del proyecto, en la que consten las consecuencias de dichas modificaciones respecto a sus suministros y entregas, y la validez de la oferta, que no deberá ser inferior a 60 días, si el plazo final lo admite. Sólo podrá iniciarse la ejecución de los trabajos de modificación si se dispone de la correspondiente autorización de la CLIENTE por escrito.

3.8 SEGUROS

El DESARROLLADOR será el único responsable de los accidentes que ocurran a su personal, al del CLIENTE o a terceros, así como de los daños que cause a las instalaciones o equipos debido a las operaciones que realizan. Para ello, el DESARROLLADOR dispondrá de una póliza de seguro que cubra

los riesgos, y acreditará al CLIENTE de su existencia. Por otro lado, cualquier daño, producido por causas ajenas al DESARROLLADOR o sus subcontratistas, quedará cubierto mediante los seguros suscritos a terceros.

3.9 OBLIGACIÓN

El DESARROLLADOR efectuará el trabajo según las leyes y reglamentos vigentes en el territorio español en la fecha del pedido. Si estas leyes varían, se habrán de modificar las características del proyecto, realizando la correspondiente oferta conforme al apartado 21 (modificaciones).

3.10 FUERZA MAYOR

Se define Fuerza Mayor como aquellas eventualidades no causadas por alguna de las partes y que son imprevisibles a la firma del pedido. Ninguna de las partes será juzgada por no cumplir sus obligaciones debido a estas causas, pero si están obligadas a comunicar, por escrito a la otra parte, el comienzo y final de la causa de Fuerza Mayor. Si este periodo excede los seis meses, el DESAROLLADOR y el CLIENTE tratarán de llegar a un acuerdo para no rescindir el pedido. Si este acuerdo no se consiguiera, se rescindirá el pedido sin perjuicio de los derechos y obligaciones asumidos hasta dicha fecha y ninguna de las partes pedirá indemnización.

3.11 RETRASOS

El DESARROLLADOR se compromete a notificar inmediatamente a el CLIENTE de cualquier retraso que pudiera producirse en el desarrollo del proyecto por causas de fuerza mayor o cualquier otra causa que implique el incumplimiento de plazos y costes estipulados. Además, deberá proponer medidas para recuperar lo antes posible los tiempos perdidos.

3.12 RESCISIÓN DEL PEDIDO

El CLIENTE podrá rescindir el pedido, después de enviar notificación por escrito, si el DESARROLLADOR no cumple con sus obligaciones contractuales esenciales dentro de los 3 meses siguientes a la fecha pactada o es declarado en suspensión de pagos, quiebra, embargo, etc. El DESARROLLADOR podrá rescindir el pedido, después de enviar notificación por escrito, si el CLIENTE no cumple con sus obligaciones contractuales esenciales referentes a pagos en los 3 meses siguientes a la fecha pactada. Ambas partes mantendrán las obligaciones asumidas hasta la fecha de entrada en vigor de la rescisión, comunicada a la parte inculpada mediante una carta enviada por notario. Excepción a esto son las causas de Fuerza Mayor.

3.13 ARBITRAJE Y LEY APLICABLE

Las cuestiones del pedido que surjan una vez firmado el contrato se someterán a un arbitraje de equidad mediante una autoridad aceptada por ambas partes. Si no se alcanzara un acuerdo, las partes se someterán a los juzgados y tribunales españoles.

3.14 INSTALACIONES PROVISIONALES Y UTILIZACIÓN DE SERVICIOS

El CLIENTE se hará cargo de los gastos de electricidad e internet y de los espacios requeridos por la empresa desarrolladora para la implantación y configuración del software en los equipos del cliente.

3.1 CONFIDENCIALIDAD

El DESARROLLADOR no revelará datos que pueda obtener de su relación con el CLIENTE a terceros. Asimismo, ningún documento, correspondiente al material suministrado ni la utilización que el CLIENTE hace del mismo, podrá ser citado o usado por el DESARROLLADOR con fines publicitarios, sin la correspondiente autorización del CLIENTE. El CLIENTE, por su parte, no podrá hacer uso de la información que dispone para otros fines diferentes al pedido realizado, salvo previa autorización por escrito por parte del DESARROLLADOR

3.2 ENTRADA EN VIGOR

Este pedido adquirirá el carácter de Contrato de compraventa en firme en el momento en que sea expresamente aceptado y firmado por el DESARROLLADOR. Las presentes condiciones comerciales prevalecerán sobre otras que puedan ser establecidas y no podrán ser modificadas ni dejadas sin efecto a no ser por pacto expreso entre el CLIENTE y el DESARROLLADOR, reflejado por escrito

PRESUPUESTO

MANO DE OBRA	Horas	Precio hora (euros)	Precio (euros)
ESTRUCTURACIÓN Y DISEÑO PREVIO DEL CÓDIGO	46	25	1150
DESARROLLO DEL CÓDIGO FUENTE	205	25	5125
DEPURACIÓN DE ERRORES	60	25	1500
IMPLANTACIÓN DEL SOFTWARE E INSTALACIÓN DE EQUIPOS	4	25	100
ELABORACIÓN DE MAUALES Y DOCUMENTACION	10	25	250
FORMACIÓN DE LOS USUARIOS	5	25	125
TOTAL MANO DE OBRA	330		8250

MATERIALES	cantidad	precio unidad (EUROS)	PRECIO (EUROS)
EQUIPOS INFORMÁTICOS	2	500	1000
LICENCIAS SOFTWARE	2	200	400
TOTAL MATERIALES			1400

13% Gastos generales	1254,5 €
6% Beneficio industrial	654,27 €
I.V.A 21%	2427,34 €
Total de la obra	13986,11 €

PRECIO TOTAL DEL PROYECTO: 13986,11 EUROS

AVISO:

Este documento es el resultado del Trabajo Fin de Grado de un alumno, siendo su autor responsable de su contenido.

Se trata por tanto de un trabajo académico que puede contener errores detectados por el tribunal y que pueden no haber sido corregidos por el autor en la presente edición.

Debido a dicha orientación académica no debe hacerse un uso profesional de su contenido.

Este tipo de trabajos, junto con su defensa, pueden haber obtenido una nota que oscila entre 5 y 10 puntos, por lo que la calidad y el número de errores que puedan contener difieren en gran medida entre unos trabajos y otros,

La Universidad de Cantabria, la Escuela Técnica Superior de Náutica, los miembros del Tribunal de Trabajos Fin de Grado así como el profesor tutor/director no son responsables del contenido último de este Trabajo.”