EXAMEN FINAL

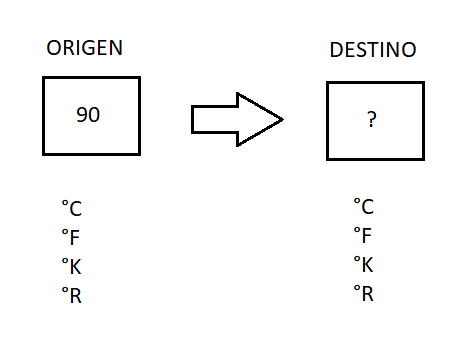
# ENUNCIADO

Desarrollar una aplicación que permita convertir una temperatura dada en cierta unidad de medida a otra unidad de medida.

La aplicación debe desarrollar una clase donde implemente toda la lógica de la solución y una ventana para la interface de usuario.

# SOLUCION

## Análisis



## Clase de la lógica

#pragma once

// Tipos de Conversion

enum TConv {

CTOF, CTOK, CTOR,

FTOC, FTOK, FTOR,

KTOC, KTOF, KTOR,

RTOC, RTOF, RTOK,

};

// Clase de la logica

class CLogica {

private:

// Constructor privado para que no se

// pueda instanciar la clase.

CLogica() {

}

public:

static float convertir(TConv tipo, float valor) {

float resultado = 0.0;

switch (tipo) {

case CTOF: resultado = valor \* 9 / 5 + 32; break;

case CTOK: resultado = 2; break;

case CTOR: resultado = 3; break;

case FTOC: resultado = 4; break;

case FTOK: resultado = 5; break;

case FTOR: resultado = 6; break;

case KTOC: resultado = 7; break;

case KTOF: resultado = 8; break;

case KTOR: resultado = 9; break;

case RTOC: resultado = 10; break;

case RTOF: resultado = 11; break;

case RTOK: resultado = 12; break;

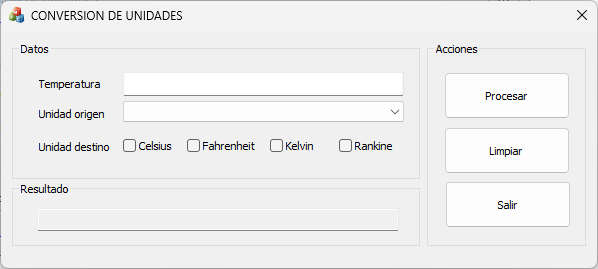
}

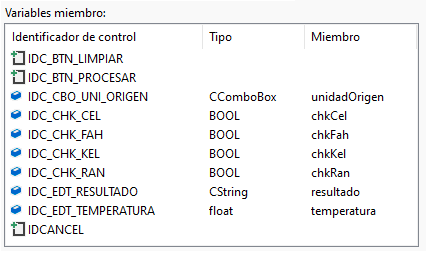
return resultado;

}

};

## Ventana





## Código IDC\_BTN\_PROCESAR

UpdateData(true);

resultado.Empty();

float valorConv = 0.0;

CString dato;

// Destino es Celsius

if (chkCel) {

switch (unidadOrigen.GetCurSel()) {

case 0: valorConv = temperatura; break;

case 1: valorConv = CLogica::convertir(FTOC, temperatura); break;

case 2: valorConv = CLogica::convertir(KTOC, temperatura); break;

case 3: valorConv = CLogica::convertir(RTOC, temperatura); break;

}

dato.Format(\_T("%.2f C"), valorConv);

resultado.Append(dato);

}

// Destino es Fahrenheit

if (chkFah) {

switch (unidadOrigen.GetCurSel()) {

case 0: valorConv = CLogica::convertir(CTOF, temperatura); break;

case 1: valorConv = temperatura; break;

case 2: valorConv = CLogica::convertir(KTOF, temperatura); break;

case 3: valorConv = CLogica::convertir(RTOF, temperatura); break;

}

dato.Format(\_T(" %.2f F"), valorConv);

resultado.Append(dato);

}

// Destino es Kelvin

if (chkKel) {

switch (unidadOrigen.GetCurSel()) {

case 0: valorConv = CLogica::convertir(CTOK, temperatura); break;

case 1: valorConv = CLogica::convertir(FTOK, temperatura); break;

case 2: valorConv = temperatura; break;

case 3: valorConv = CLogica::convertir(RTOK, temperatura); break;

}

dato.Format(\_T(" %.2f K"), valorConv);

resultado.Append(dato);

}

// Destino es Rankine

if (chkRan) {

switch (unidadOrigen.GetCurSel()) {

case 0: valorConv = CLogica::convertir(CTOR, temperatura); break;

case 1: valorConv = CLogica::convertir(FTOR, temperatura); break;

case 2: valorConv = CLogica::convertir(KTOR, temperatura); break;

case 3: valorConv = temperatura; break;

}

dato.Format(\_T(" %.2f R"), valorConv);

resultado.Append(dato);

}

UpdateData(false);