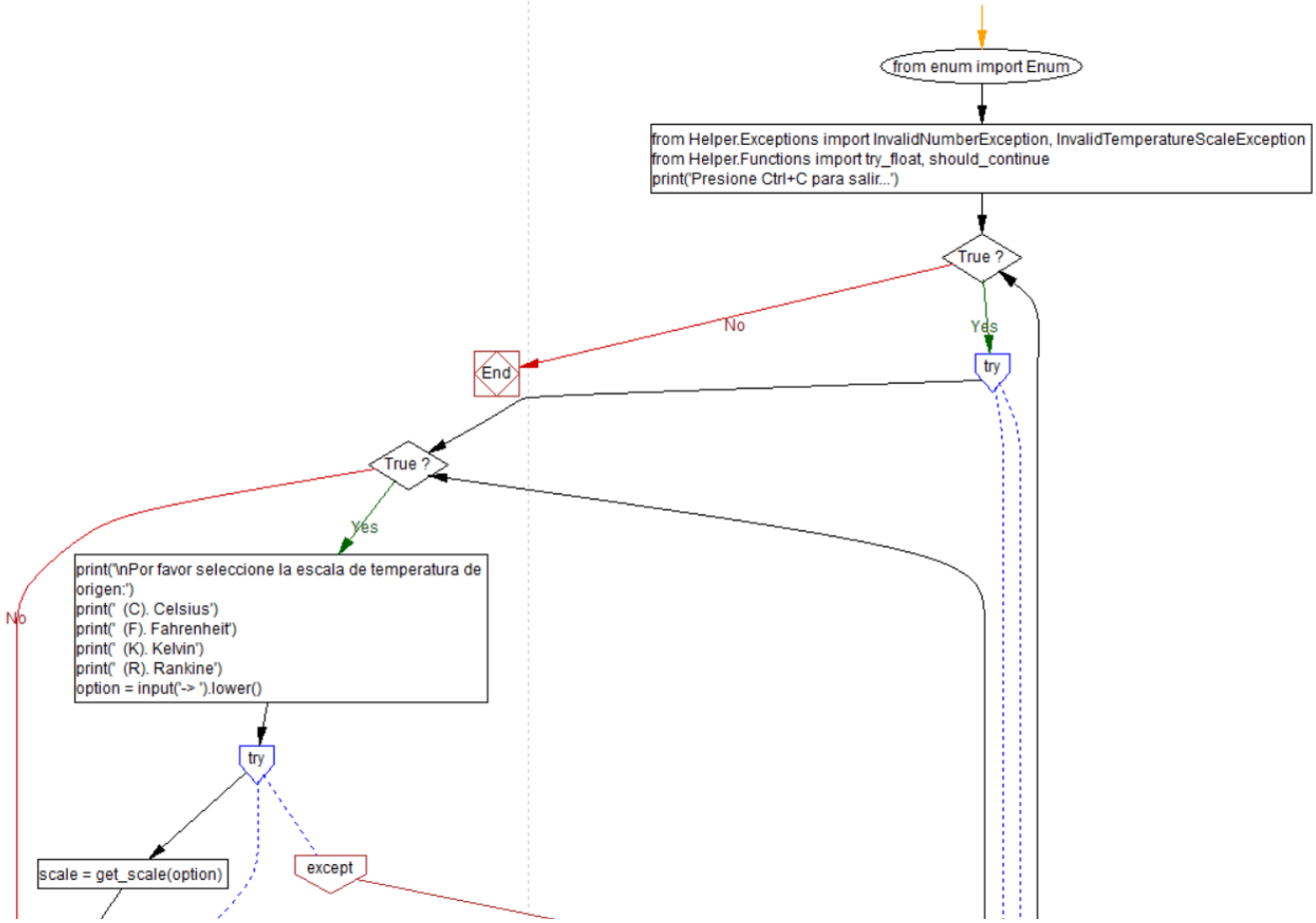
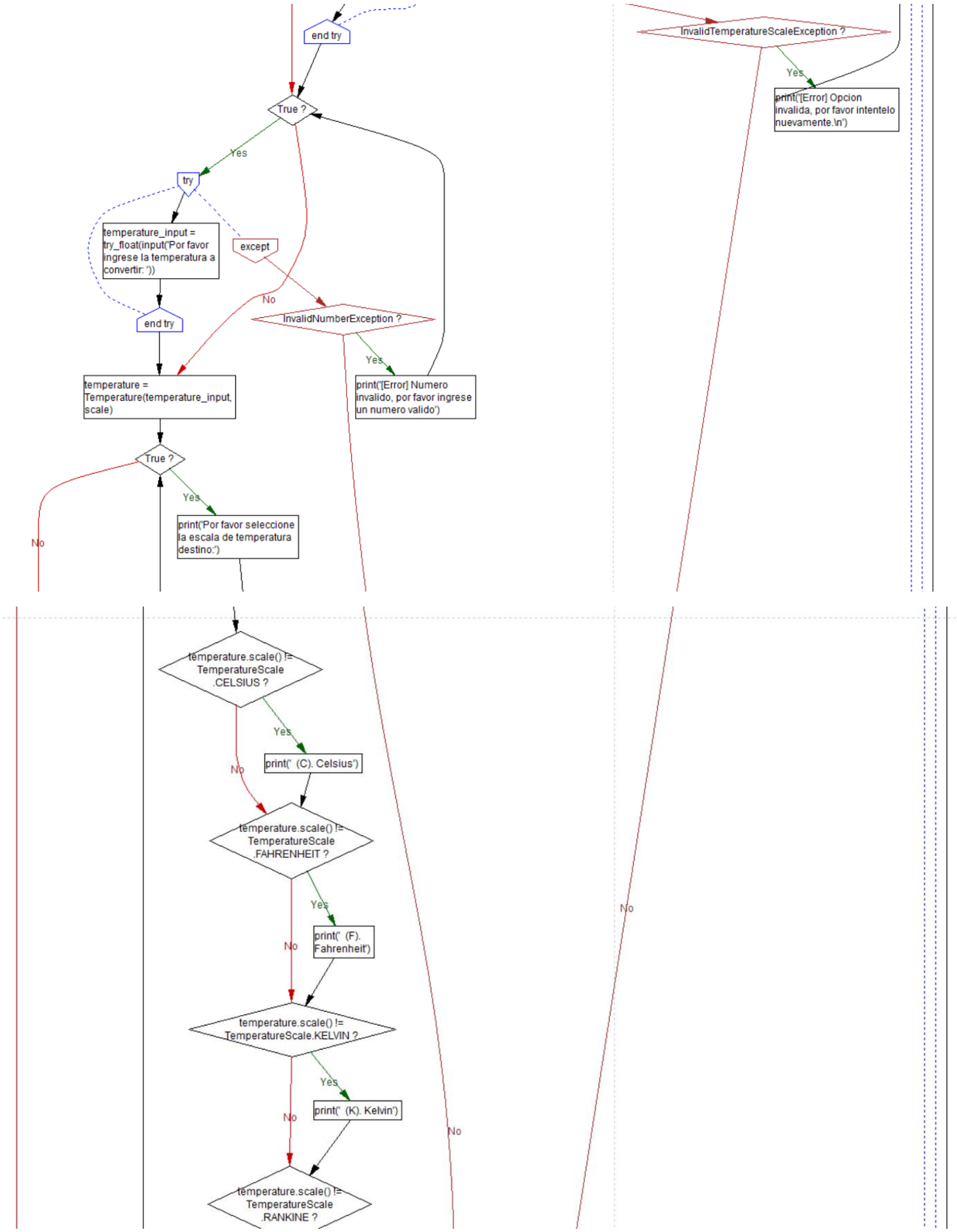
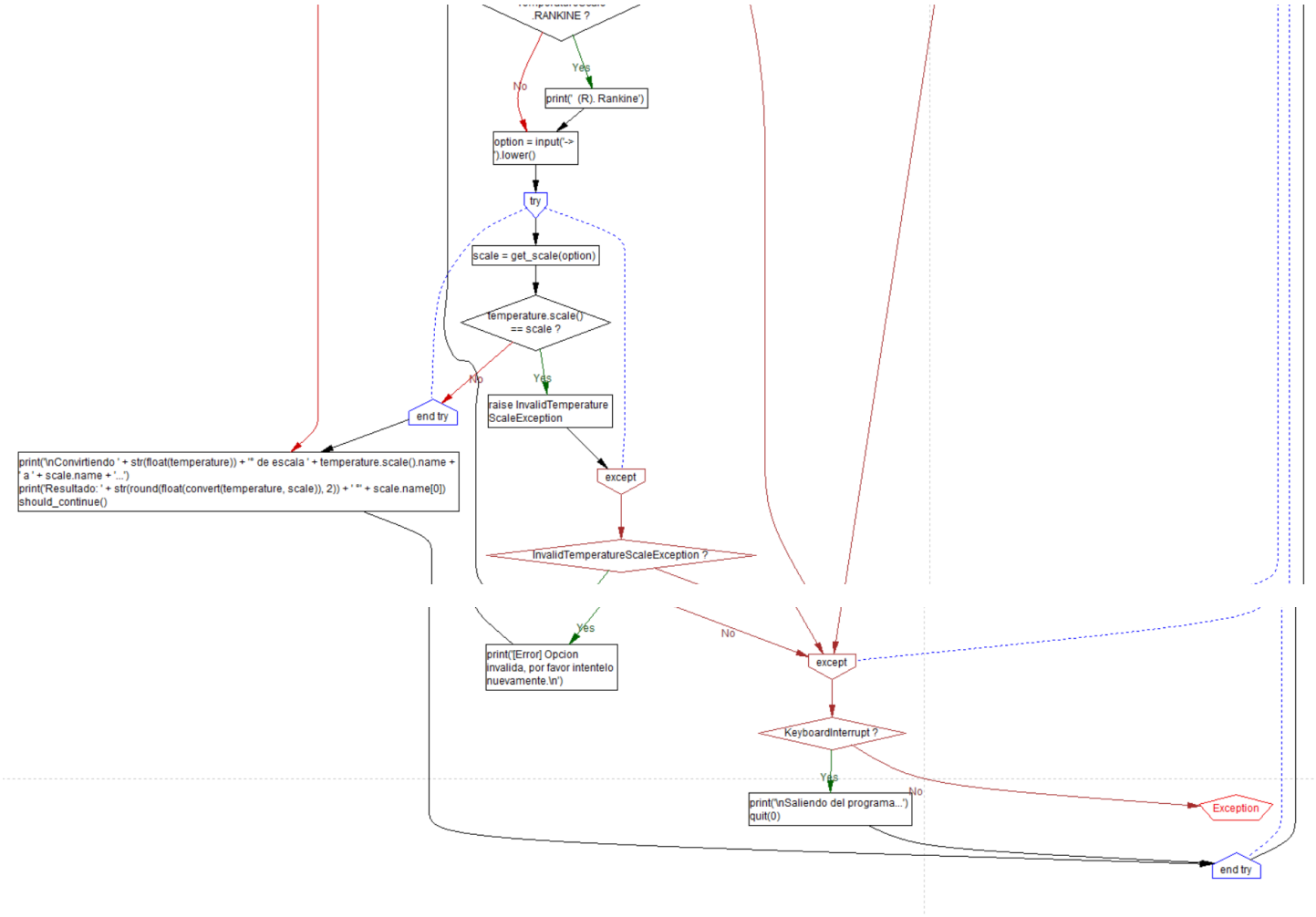
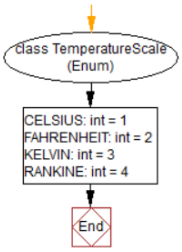
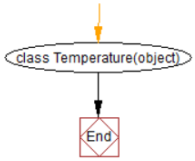
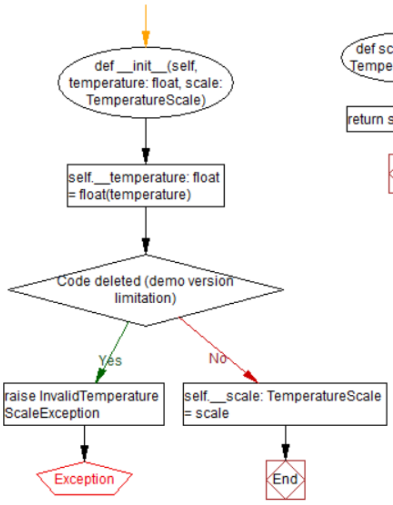

from enum import Enum
from Helper.Exceptions import InvalidNumberException, InvalidTemperatureScaleException
from Helper.Functions import try_float, should_continue
print('Presione Ctrl+C para salir...')
True ?
No
Yes
End
try
True ?
Yes
No
print('\nPor favor seleccione la escala de temperatura de origen:')
print(' (C). Celsius')
print(' (F). Fahrenheit')
print(' (K). Kelvin')
print(' (R). Rankine')
option = input('-> ').lower()
try
scale = get_scale(option)
except

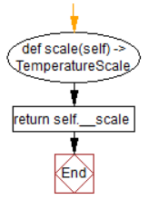

end try
True ?
Yes
try
temperature_input = try_float(input('Por favor ingrese la temperatura a convertir: '))
except
No
end try
InvalidNumberException ?
Yes
print('[Error] Numero invalido, por favor ingrese un numero valido')
temperature = Temperature(temperature_input, scale)
True ?
Yes
No
print('Por favor seleccione la escala de temperatura destino:')
InvalidTemperatureScaleException ?
Yes
print('[Error] Opcion invalida, por favor intentelo nuevamente.\n')
temperature.scale() != TemperatureScale .CELSIUS ?
Yes
No
print(' (C). Celsius')
temperature.scale() != TemperatureScale .FAHRENHEIT ?
Yes
No
print(' (F). Fahrenheit')
temperature.scale() != TemperatureScale.KELVIN ?
Yes
No
print(' (K). Kelvin')
temperature.scale() != TemperatureScale .RANKINE ?
No
No

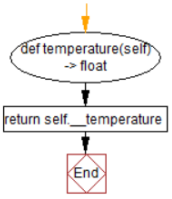

.RANKINE ?
Yes
No
print(' (R). Rankine')
option = input('->
').lower()
try
scale = get_scale(option)
temperature.scale()
== scale ?
No
Yes
raise InvalidTemperature
ScaleException
end try
except
print('\nConvirtiendo ' + str(float(temperature)) + '° de escala ' + temperature.scale().name +
' a ' + scale.name + '...')
print('Resultado: ' + str(round(float(convert(temperature, scale)), 2)) + ' °' + scale.name[0])
should_continue()
InvalidTemperatureScaleException ?
Yes
No
print('[Error] Opcion
invalida, por favor intentelo
nuevamente.\n')
except
KeyboardInterrupt ?
Yes
No
print('\nSaliendo del programa...')
quit(0)
Exception
end try

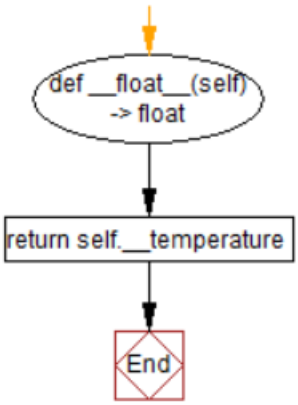

class TemperatureScale (Enum)
CELSIUS: int = 1
FAHRENHEIT: int = 2
KELVIN: int = 3
RANKINE: int = 4
End

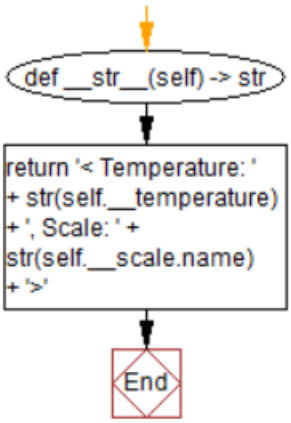

class Temperature(object)
End

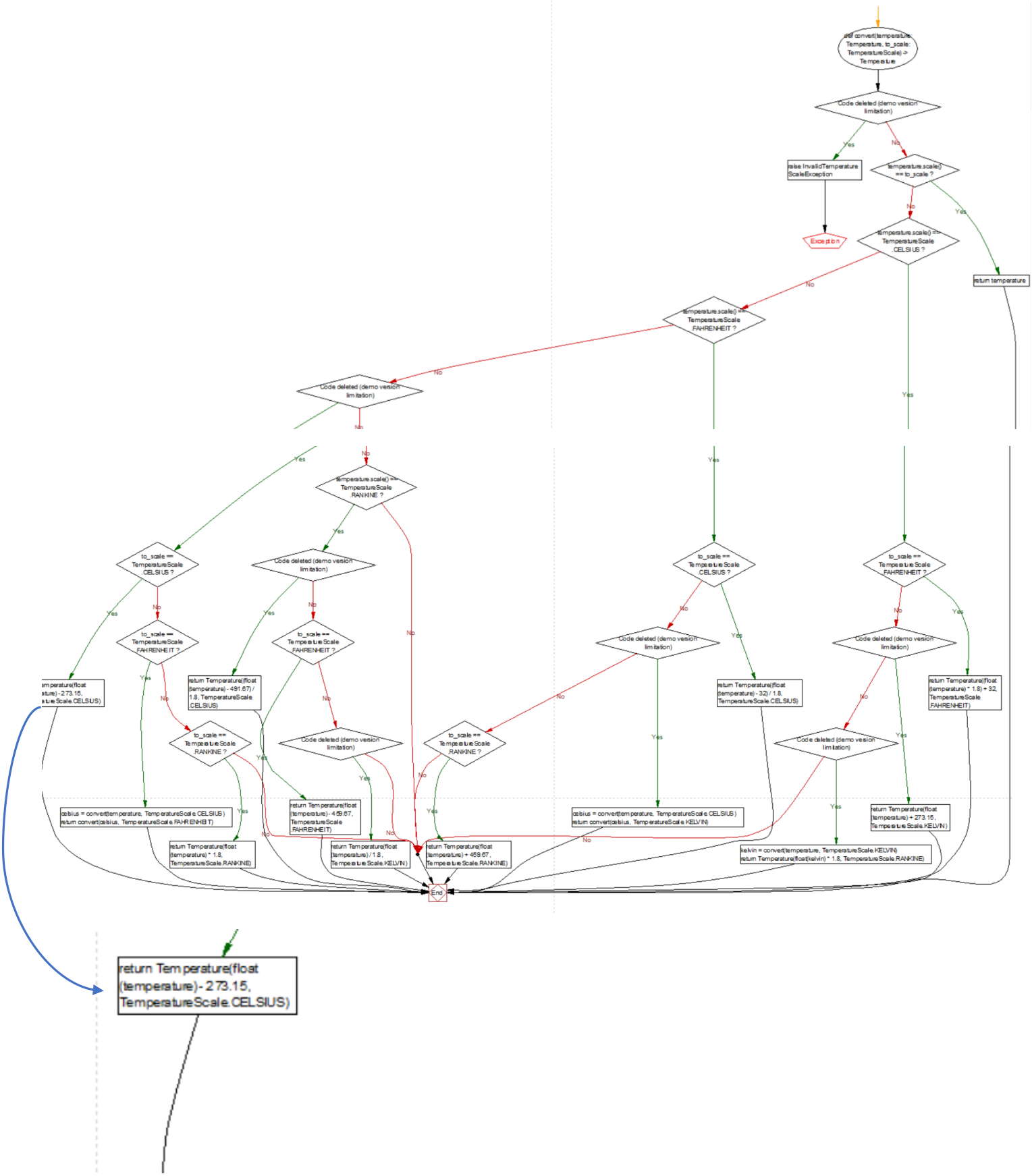

def __init__(self, temperature: float, scale: TemperatureScale)
self.__temperature: float = float(temperature)
Code deleted (demo version limitation)
Yes
No
raise InvalidTemperatureScaleException
Exception
self.__scale: TemperatureScale = scale
End

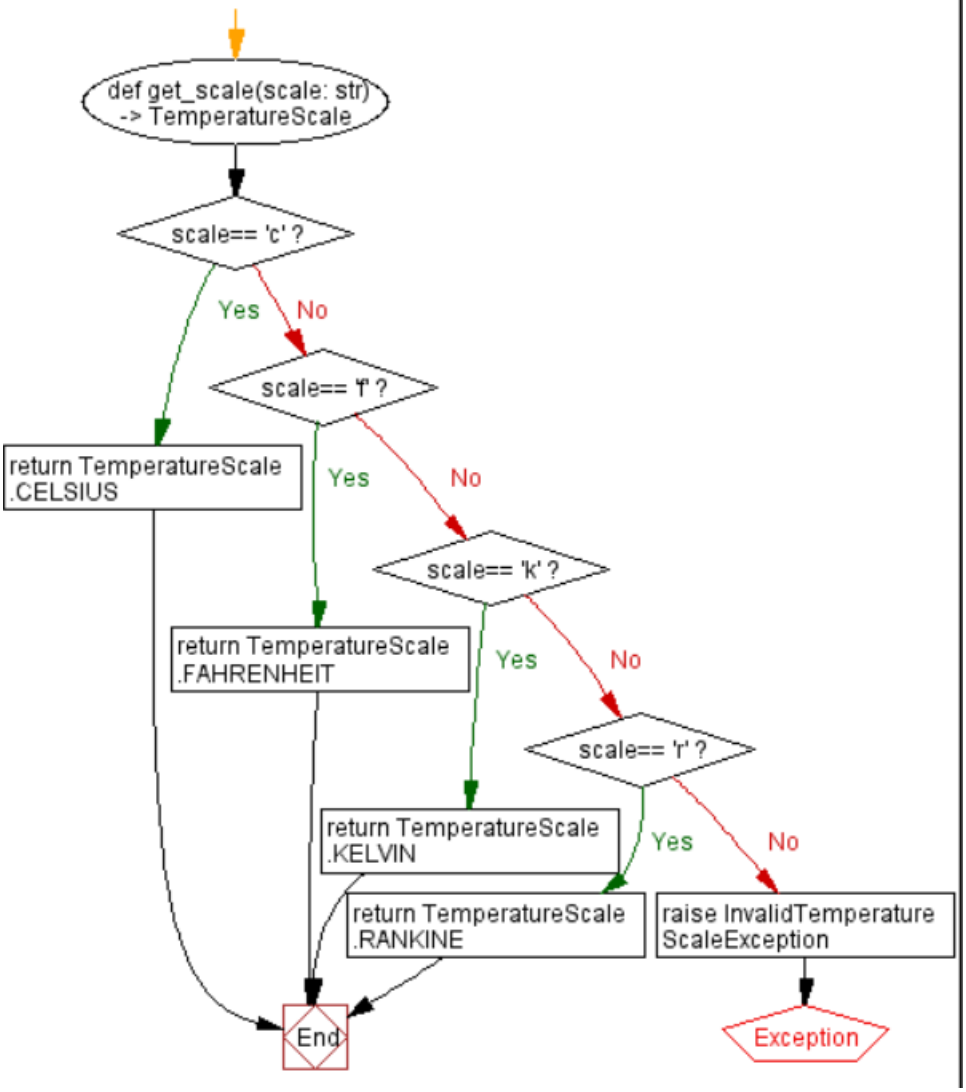

def scale(self) -> TemperatureScale
return self.__scale
End


def temperature(self) -> float
return self.__temperature
End


def __float__(self) -> float
return self.__temperature
End


def __str__(self) -> str
return '< Temperature: ' + str(self.__temperature) + ', Scale: ' + str(self.__scale.name) + '>'
End


def convert(temperature: Temperature, to_scale: TemperatureScale) -> Temperature
Code deleted (demo version limitation)
raise InvalidTemperatureScaleException
Exception
temperature.scale() == to_scale ?
temperature.scale() == TemperatureScale.CELSIUS ?
return temperature
temperature.scale() == TemperatureScale.FAHRENHEIT ?
Code deleted (demo version limitation)
temperature.scale() == TemperatureScale.RANKINE ?
to_scale == TemperatureScale.CELSIUS ?
Code deleted (demo version limitation)
to_scale == TemperatureScale.FAHRENHEIT ?
to_scale == TemperatureScale.RANKINE ?
return Temperature(float(temperature) - 491.67) / 1.8, TemperatureScale.CELSIUS)
return Temperature(float(temperature) - 32) / 1.8, TemperatureScale.CELSIUS)
return Temperature(float(temperature) * 1.8) + 32, TemperatureScale.FAHRENHEIT)
celsius = convert(temperature, TemperatureScale.CELSIUS) return convert(celsius, TemperatureScale.FAHRENHEIT)
celsius = convert(temperature, TemperatureScale.CELSIUS) return convert(celsius, TemperatureScale.KELVIN)
return Temperature(float(temperature) + 273.15, TemperatureScale.KELVIN)
return Temperature(float(temperature) * 1.8, TemperatureScale.RANKINE)
return Temperature(float(temperature) - 459.67, TemperatureScale.FAHRENHEIT)
return Temperature(float(temperature) / 1.8, TemperatureScale.KELVIN)
return Temperature(float(temperature) + 459.67, TemperatureScale.RANKINE)
kelvin = convert(temperature, TemperatureScale.KELVIN) return Temperature(float(kelvin) * 1.8, TemperatureScale.RANKINE)
End
Yes
No
return Temperature(float(temperature) - 273.15, TemperatureScale.CELSIUS)


def get_scale(scale: str)
-> TemperatureScale
scale== 'c' ?
Yes
No
return TemperatureScale
.CELSIUS
scale== 'f' ?
Yes
No
return TemperatureScale
.FAHRENHEIT
scale== 'k' ?
Yes
No
return TemperatureScale
.KELVIN
scale== 'r' ?
Yes
No
return TemperatureScale
.RANKINE
raise InvalidTemperature
ScaleException
End
Exception