"""

Parent class Template by JOR

"""

# In any complex application, create a base class which will never be instantiated.

class Device():

# Define a class object attribute, it will be the same for any instance of the class

pi = 3.142

# Constructor, called whenever an instance of the class is created.

def \_\_init\_\_(self) -> None:

print("Running constructor for base class")

# Create attributes and set initial values

self.debug = False

def run(self):

raise NotImplementedError("This is an abstract class, do not instantiate")

def calculate\_crc(self, frame:str)->int:

print("Checking CRC from base")

# Put real code in here, this is a dummy value for initial setup

crc = 123456789

return crc

"""

Child class Template by JOR

"""

# Create child class which can access the methods and properties of the base class

class Firewall(Device):

# Constructor, called whenever an instance of the class is created.

# Use parameter1 as a tag to identify the object

def \_\_init\_\_(self, parameter1) -> None:

# Call back to the parent class

Device.\_\_init\_\_(self)

print(f"Running constructor for {parameter1}")

# Create attributes and set initial values

self.parameter1 = parameter1

self.test\_message = ""

def configure\_firewall(self):

print("Configuring Firewall")

def calculate\_crc(self, frame:str)->int:

print("Checking CRC from child")

# Put real code in here, this is a dummy value for initial setup

crc = 123456789

return crc