```python
from abc import ABC, abstractmethod
class empleado_31(ABC):
    def __init__(self, RFC, apellido, nombres):
        self.__RFC = RFC
        self.__apellido =apellido
        self.__nombres=nombres
    @abstractmethod
    def calcular_ingresos(self):
        pass
    @abstractmethod
    def mostrar_datos(self):
        pass
class Emprenddor_31 (empleado_31):
    def __init__ (self, RFC,apellido,nombres, monto_vendido,tasa_comision):
        super().__init__(RFC,apellido, nombres)
        self.monto_vendido= monto_vendido
        self.tasa_comision= tasa_comision
    def calcular_ingresos(self):
        ingresos= self.monto_vendido*self.tasa_comision
        if self.monto_vendido< 1000:
            bonificacion = ingresos* 0.05
        elif 1000<=self.monto_vendido<=5000:
            bonificacion =ingresos*0.10
        else:
            bonificacion=ingresos*0.15
        if ingresos>1000:
            descuento=ingresos*0.15
        else:
            descuento= 0
        sueldo_neto=ingresos+bonificacion- descuento
        return sueldo_neto
    def mostrar_datps(self):
        return (f"Vendedor: {self.__nombres} {self.__apellido}, RFC: {self.__RFC}, Ingresos: {self.calcular_ingresos()}")
class EmpleadoPermanente(empleado_31, Emprenddor_31):
    def _init_(self, RFC, apellidos, nombres, sueldo_base):
        super()._init_(RFC, apellidos, nombres)
        self.sueldo_base = sueldo_base
```

```python
        else:
            descuento= 0
        sueldo_neto=ingresos+bonificacion- descuento
        return sueldo_neto
    def mostrar_datps(self):
        return (f"Vendedor: {self.__nombres} {self.__apellido}, RFC: {self.__RFC}, Ingresos: {self.calcular_ingresos()}")
class EmpleadoPermanente(empleado_31, Emprenddor_31):
    def _init_(self, RFC, apellidos, nombres, sueldo_base):
        super()._init_(RFC, apellidos, nombres)
        self.sueldo_base = sueldo_base

    def calcular_ingresos(self):
        bonificacion= self.sueldo_base* 0.11
        ingresos= self.sueldo_base+ bonificacion
        return ingresos
    def mostrar_informacion(self):
        return f"Empleado Permanente:{self.__nombres} {self.__apellido},
        RFC: {self.__RFC},
        Ingresos:{self.calcular_ingresos()}"

def validar_salario(empleado):
    ingresos=empleado.calcular_ingresos()
    if ingresos < 150:
        raise ValueError(f"salario neto {empleado.__nombres} {empleado.__apellido}")
    return ingresos
```