Université des sciences et de la téchnologie Houari Boumédiène

Tp4 : Héritage

LAOUAR Nadjoua
SEDDIK Hala Amina
SG : 04

Exercice 1 :

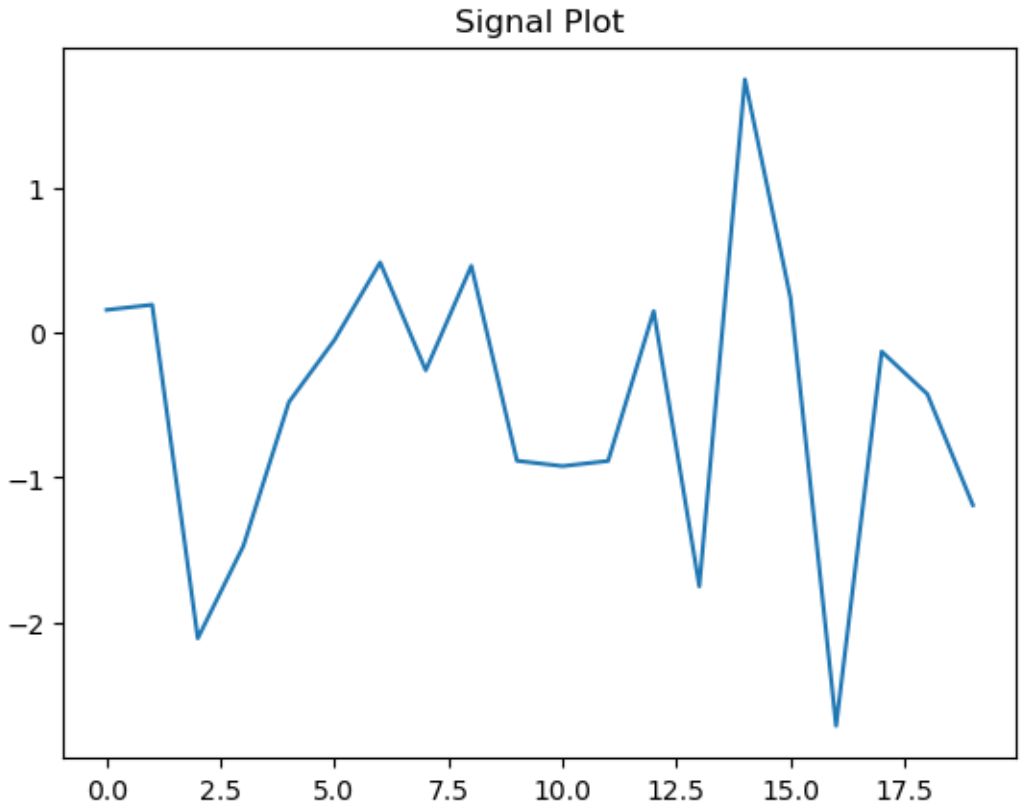
```
# Module "employees.py"

class Employe:
    def __init__(self, nom, identifiant, pin=500):
        self.Nom = nom
        self.ID = identifiant
        self.Pin = pin

    def calcul_prime(self, a):
        if a:
            return self.Pin * 2
        else:
            return self.Pin * 0.5

    def salaire(self, prime):
        return self.Pin * prime

    def affiche(self, a):
        prime = self.calcul_prime(a)
        salary = self.salaire(prime)
        print(f"Nom: {self.Nom}, ID: {self.ID}, Point_indiciaire:
{self.Pin}")
        print(f"Salaire: {salary}, Statut: Employé")


class Ingenieur(Employe):
    def __init__(self, nom, identifiant, etat, pin=500):
        super().__init__(nom, identifiant, pin)
        self.Etat = etat

    def salaire(self, prime):
        if self.Etat == "stagiaire":
            return self.Pin * 4 * prime
        else:
            return self.Pin * 6 * prime

    def affiche(self, a):
        prime = self.calcul_prime(a)
        salary = self.salaire(prime)
        print(f"Nom: {self.Nom}, ID: {self.ID}, Point_indiciaire:
{self.Pin}")
```

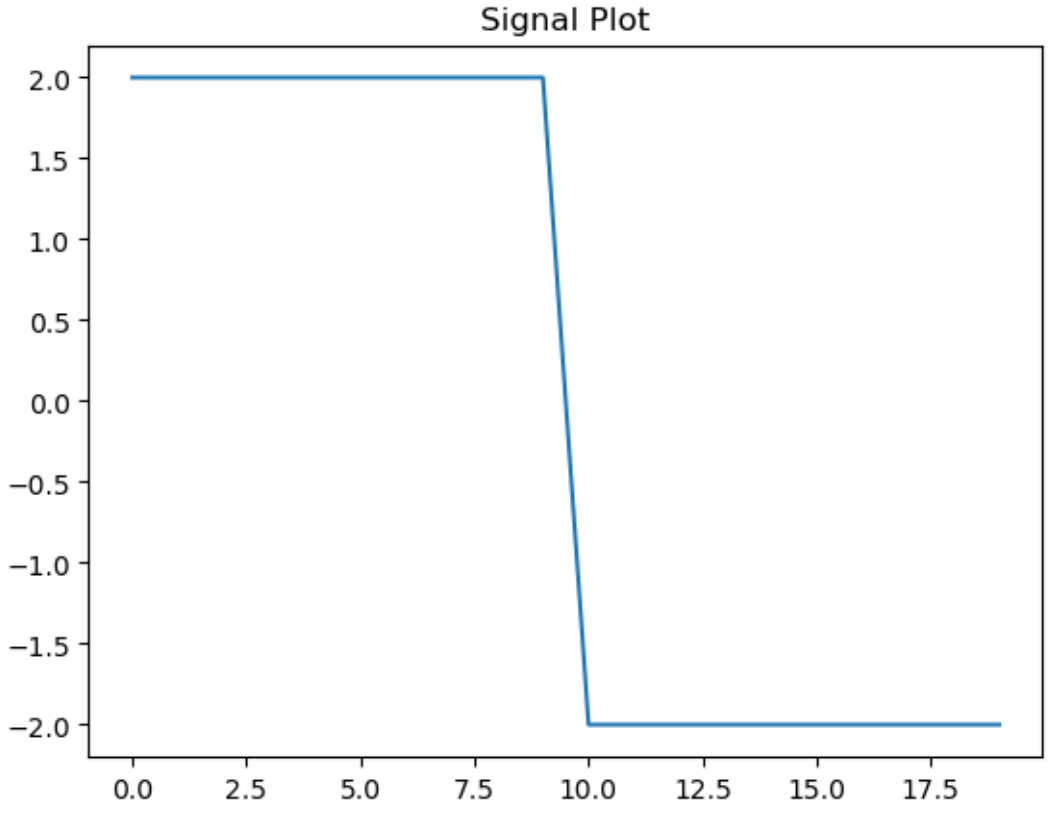
```
        print(f"Salaire: {salary}, Statut: Ingenieur {self.Etat}")


class Technicien(Employe):
    def salaire(self, prime):
        return self.Pin * 2 * prime

    def affiche(self, a):
        prime = self.calcul_prime(a)
        salary = self.salaire(prime)
        print(f"Nom: {self.Nom}, ID: {self.ID}, Point_indiciaire:
{self.Pin}")
        print(f"Salaire: {salary}, Statut: Technicien")

# Programme principal

ingenieur1 = Ingenieur("John Doe", 1, "stagiaire")
technicien1 = Technicien("Jane Doe", 2)

print("Ingénieur:")
ingenieur1.affiche(True)

print("\nTechnicien:")
technicien1.affiche(False)
```

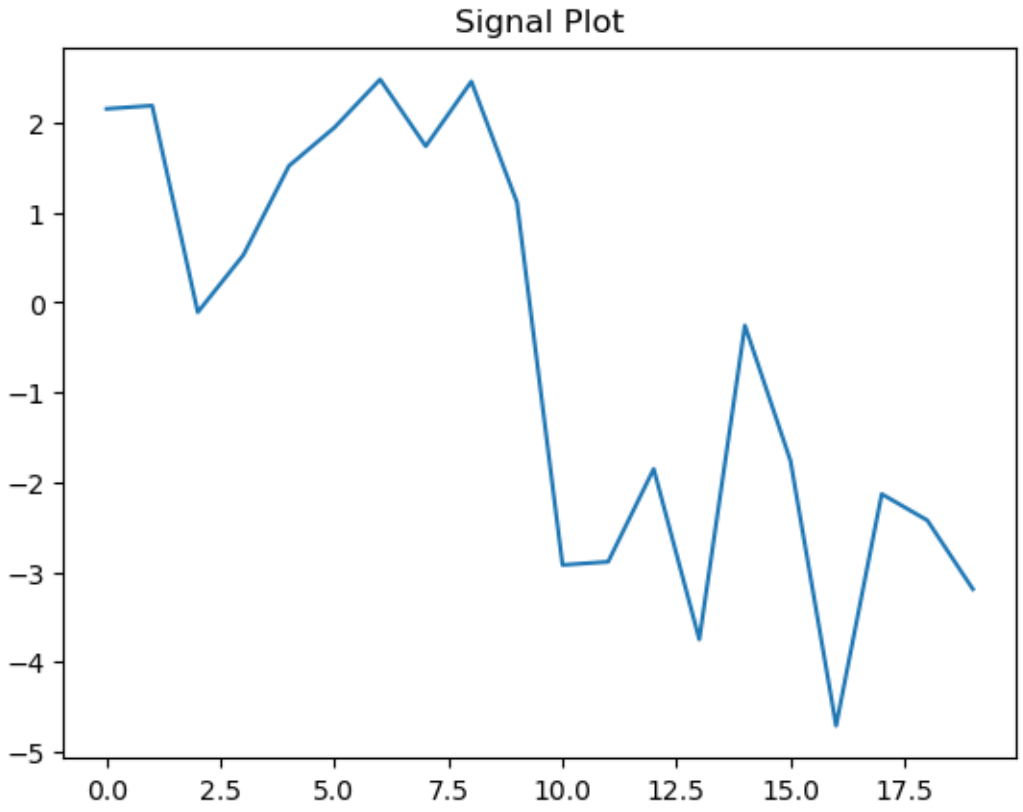
Exercice 2 :

```
# Module "signal_module.py"

import numpy as np
import matplotlib.pyplot as plt

class Signal:
    def __init__(self, nbr):
        self.nbr = nbr
        self.tab = np.zeros(nbr)

    def echantillons(self):
        return self.nbr

    def moyenne(self):
        return np.mean(self.tab)

    def correlation(self):
        return np.corrcoef(self.tab)

    def display(self):
        print(f"Nombre d'échantillons: {self.nbr}")
        print(f"Vecteur tab: {self.tab}")
        plt.plot(self.tab)
        plt.title("Signal Plot")
        plt.show()


class Aleatoire(Signal):
    def __init__(self, nbr, sigma, mean):
        super().__init__(nbr)
        self.sigma = sigma
```

```
        self.mean = mean
        self.init_alea()

    def init_alea(self):
        self.tab = np.random.normal(self.mean, self.sigma, self.nbr)

    def correlation(self):
        print("Classe Aleatoire")
        return np.power(self.tab, 2)


class Deterministe(Signal):
    def __init__(self, nbr, amplitude):
        super().__init__(nbr)
        self.amplitude = amplitude
        self.valeurs()

    def valeurs(self):
        self.tab[:self.nbr // 2] = self.amplitude
        self.tab[self.nbr // 2:] = -self.amplitude

    def correlation(self):
        return np.corrcoef(self.tab)

def addition(aleatoire, deterministe):
    s = Deterministe(aleatoire.nbr, 0)  # Crée un objet déterministe
localement
    s.tab = aleatoire.tab + deterministe.tab
    return s


# Programme principal
if __name__ == "__main__":
    aleatoire_obj = Aleatoire(20, 1, 0)
    deterministe_obj = Deterministe(20, 2)

    # Appeler toutes les méthodes
    print(f"Nombre d'échantillons Aleatoire:
{aleatoire_obj.echantillons()}")
    print(f"Moyenne Aleatoire: {aleatoire_obj.moyenne()}")
    aleatoire_obj.display()

    print(f"Nombre d'échantillons Deterministe:
{deterministe_obj.echantillons()}")
    print(f"Moyenne Deterministe: {deterministe_obj.moyenne()}")
    deterministe_obj.display()

    correlation_aleatoire = aleatoire_obj.correlation()
    correlation_deterministe = deterministe_obj.correlation()

    s = addition(aleatoire_obj, deterministe_obj)
    s.display()
```


Signal Plot
1
0
−1
−2
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Signal Plot
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Signal Plot
2
1
0
−1
−2
−3
−4
−5
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5