**Correction de la SAE 101 :**

**Question préliminaire :**

**"""créer un tableau d'amis qui modélise le réseau d'amis selon le principe """**

**p\_amis=["Yasmine","Muriel","Joël","Yasmine","Joël","Muriel","Yasmine","Thomas","Joël","Nassim","Andrea","Ali","Joël","Ali",**

**"Joël","Andrea","Nassim","Ali","Nassim","Andrea","Thomas","Daria","Thomas","Carole","Thierry","Axel","Léo","Thierry","Léo","Axel","Léo","Valentin","Andrea","Valentin"]**

**Définition du tableau :**

**amis = ["Alice", "Bob", "Alice", "Charlie", "Bob", "Denis"]**

**Création d'un tableau pour tester les fonctions**

**t\_amis =["Sarah","Aya","Amina","Georges","Sarah","Frédéric","Sarah","Chloé","Georges","Aya"]**

**Question 1 :**

**"""Renvoyer le nombre de répetition du prénom dans le tableau"""**

**def nb\_amis(amis,prenom):**

**i=0**

**#créer une variable ami qu'on initialise à zéro et qui prend à la fin le nombre de répétition du prenom**

**ami=0**

**while i<len(amis):**

**#si l'élement de l'indice i est égal à prenom**

**if amis[i]==prenom:**

**ami+=1**

**i+=1**

**return ami**

**Test de la question 1 :**

**def test\_nb\_amis():**

**assert nb\_amis(p\_amis,"Yasmine")==3**

**assert nb\_amis(amis,"Alice")==2**

**assert nb\_amis(amis,"charles")==0**

**assert nb\_amis(t\_amis,"Georges")==2**

**assert nb\_amis(t\_amis,"Sarah")==3**

**print("le test de la foncion nb\_amis est ok ")**

**test\_nb\_amis()**

**Question 2 :**

**"""consiste à créer un tableau vide et ajouter le prénom tant qu'il n'y est pas dans la tableau """**

**def tab\_taille\_personne(amis):**

**i=0**

**tab=[]**

**while i<len(amis):**

**#si le prénom n'est pas un élément du tableau on l'ajoute**

**if amis[i] not in tab:**

**tab.append(amis[i])**

**i+=1**

**return tab**

**"""Créer une fonction qui renvoie la longueur du tableau de la fonction tab\_taille\_personne"""**

**def taille\_personne(amis):**

**return len(tab\_taille\_personne(amis))**

**Test de la question 2 :**

**"""créer une fonction qui teste la fonction taille\_personne"""**

**def test\_taille\_personne():**

**assert taille\_personne(amis)==4**

**assert taille\_personne(p\_amis)==13**

**assert not taille\_personne(amis)==2**

**assert taille\_personne(t\_amis)==6**

**print("Le test de la fonction taille\_personne est ok ")**

**test\_taille\_personne()**

**Question 3 :**

**""" Une fonction qui prend en paramètre un chemin vers un fichier et retourner**

**un tableau modélisant les interactions entre les personnes du fichier """**

**def lecture\_path(path):**

**fichier=open(path,encoding='utf-8',mode='r')**

**tab=[]**

**li=fichier.readline()**

**#lire le fichier jusqu'à la dernière ligne ""**

**while li!="":**

**li=li.strip() #pour effacer les espaces au début et à la fin de la chaine de caractère**

**li=li.split(";") #séparer la chaine là où il y a ';'**

**tab.append(li[0])**

**tab.append(li[1])**

**li=fichier.readline()**

**return tab**

**Test de la question 3 :**

**"""tester la fonction lecture\_reseau en entrant en paramètre le chemin vers le fichier """def test\_lecture\_reseau():**

**assert lecture\_path("newfiles/testquestion3.csv")==['Yasmine','Muriel','Joël','Yasmine','Joël','Muriel','Yasmine','Thomas',**

**'Joël','Nassim','Andrea','Ali','Joël','Ali','Joël','Andrea','Nassim','Ali','Nassim','Andrea','Thomas','Daria','Thomas','Carole',**

**'Thierry','Axel','Léo','Thierry','Léo','Axel','Léo','Valentin','Andrea','Valentin']**

**assert not lecture\_path("newfiles/Communaute1.csv")==['Barbra','Cloe','Louis','Rufino','Idelle','Mady','Björn','Giedrius']**

**print("le test de la question est ok")**

**test\_lecture\_reseau()**

**Question 4 :**

**"""Renvoyer un tableau des amis d'une personne dont le nom entré en paramètre"""**

**def tableau(amis,prenom):**

**tab=[]**

**j=0**

**while j<len(amis):**

**#si l'élement de l'indice j est égal à prenom**

**if amis[j]==prenom:**

**if j%2==0: #si l'indice du tableau est pair**

**tab.append(amis[j+1]) #ajouter au tableau le prénom suivant**

**else:**

**tab.append(amis[j-1]) #ajouter au tableau le prénom précédant**

**j+=1**

**return tab**

**"""Créer un dictionnaire qui prend comme clés les prénoms des membres et comme valeur le tableau de leurs amis """**

**def dico\_reseau(amis):**

**dic={}**

**amis2=tab\_taille\_personne(amis) #créer un tableau amis2 qui contient les prénoms des personnes participant au réseau**

**i=0**

**while i<len(amis2):**

**dic[amis2[i]]=tableau(amis,amis2[i]) #ajouter le prénom au dictionnaire et prenant comme valeur le tableau de ses amis**

**i+=1**

**return dic**

**Test de la question 4 :**

**""" Tester la fonction dico\_reseau """**

**def test\_dico\_reseau():**

**assert dico\_reseau(amis)=={'Alice': ['Bob', 'Charlie'], 'Bob': ['Alice', 'Denis'], 'Charlie': ['Alice'], 'Denis': ['Bob']}**

**assert not dico\_reseau(amis)=={'Alice': ['Bob', 'Charlie']}**

**assert not dico\_reseau(p\_amis)=={'Muriel':['Yasmine','Daria']}**

**assert dico\_reseau(t\_amis)=={'Sarah': ['Aya', 'Frédéric', 'Chloé'],'Aya': ['Sarah', 'Georges'],**

**'Amina': ['Georges'],'Georges': ['Amina', 'Aya'],'Frédéric': ['Sarah'],'Chloé': ['Sarah']}**

**assert not dico\_reseau(t\_amis)=={'Aya': ['Sarah', 'Georges']}**

**print("le test de le fonction est ok")**

**test\_dico\_reseau()**

**Question 5 :**

**""" Renvoyer le nombre d'amis des personnes les plus populaires du réseau"""**

**def nb\_amis\_plus\_pop(dico\_reseau):**

**#Créer une tableau tab2nom contenant les prénoms des personnes participant au réseau**

**tab2nom=list(dico\_reseau)**

**max=0**

**i=0**

**"""si le nombre d'amis de la personne qui est interpreté par la longueur du tableau est supérieur à max alors**

**la variable prend la nouvelle valeur"""**

**while i<len(tab2nom):**

**tab=dico\_reseau[tab2nom[i]] #stocker le tableau des amis de la personne dans un tab**

**if len(tab) > max : #condition si la longueur du tableau est supérieur à la valeur de max**

**max=len(tab)**

**i+=1**

**return max**

**test de la question 5 :**

**"""définir une fonction test\_nb\_amis\_plus\_pop qui teste le nombre d'amis des personnes les plus populaires du réseau"""**

**def test\_nb\_amis\_plus\_pop():**

**assert nb\_amis\_plus\_pop(dico\_reseau(amis))==2**

**assert nb\_amis\_plus\_pop(dico\_reseau(p\_amis))==5**

**assert not nb\_amis\_plus\_pop(dico\_reseau(amis))==3**

**assert not nb\_amis\_plus\_pop(dico\_reseau(p\_amis))==1**

**assert nb\_amis\_plus\_pop(dico\_reseau(t\_amis))==3**

**assert not nb\_amis\_plus\_pop(dico\_reseau(t\_amis))==6**

**assert nb\_amis\_plus\_pop(dico\_reseau(lecture\_path("newfiles/Communaute1.csv")))==11**

**print("le test de la fonction nb\_amis\_plus\_pop est ok")**

**test\_nb\_amis\_plus\_pop()**

**Question 6 :**

**"""Renvoyer un tableau contenant les prénoms de toutes les personnes populaires du réseau"""**

**def les\_plus\_pop(dico\_reseau):**

**#Créer un tableau tab2nom contenant les prénoms des personnes participant au réseau**

**tab2nom=list(dico\_reseau)**

**tab2pop=[]**

**i=0**

**while i<len(tab2nom):**

**tab=dico\_reseau[tab2nom[i]]**

**#si le nombre d'amis est égal au nombre maximal des amis dans ce réseau ajouter son préonm au tableau des populaires**

**if len(tab) == nb\_amis\_plus\_pop(dico\_reseau) :**

**tab2pop.append(tab2nom[i])**

**i+=1**

**return tab2pop**

**Test de la question 6 :**

**"""Une fonction qui teste les trois tableau """**

**def test\_les\_plus\_pop():**

**assert les\_plus\_pop(dico\_reseau(amis))==['Alice', 'Bob']**

**assert not les\_plus\_pop(dico\_reseau(amis))==['Denis']**

**assert les\_plus\_pop(dico\_reseau(p\_amis))==['Joël']**

**assert not les\_plus\_pop(dico\_reseau(p\_amis))==["Andrea"]**

**assert les\_plus\_pop(dico\_reseau(t\_amis))==['Sarah']**

**assert les\_plus\_pop(dico\_reseau(lecture\_path("newfiles/Communaute1.csv")))==['Rufino', 'Mady', 'Vittorio']**

**print("le test de la fonction les\_plus\_pop est ok ")**

**test\_les\_plus\_pop()**