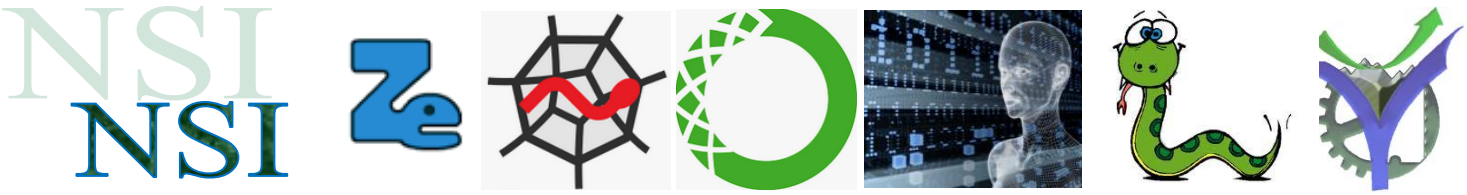

Nom :	Note : / 20
	Classe :

Algorithmes, propos liminaires

La fonction factorielle n notée n ! Pour rappel :

n! = n * (n - 1) * (n - 2) * ... * 1

(n+1) ! = (n + 1) n !

$$n! = \prod_{1 \leqslant i \leqslant n} i = 1 \times 2 \times 3 \times \ldots \times (n-1) \times n.$$

0! = 1

```
def factorielle(n):
    F = [1]
    for k in range(1,n+1):
        F.append( F[k-1] * k )
    return F[-1]
```

Voilà une possibilité de réaliser le calcul de cette opér

1) Décrire le principe utilisé pour ce script :

2) Supposons que nous cherchons à réaliser toutes les combinaisons obtenues à partir d'un ensemble de 3 éléments notés (a, b, c). Donner toutes les combinaisons possibles de trois éléments :

Quelle relation avec 3 ! ?

3) Même question pour un ensemble de quatre éléments (a, b, c, d) et 4 !

Problème du voyageur

4) En changeant la ville de départ, écrivez les distances totales obtenues :

Ville de départ	Nancy	Metz	Paris	Reims	Troyes
Distance (km)	810				

5) Vous pouvez afficher le bilan de tous les trajets avec voyageur_naif.py

Donner alors le coût minimum obtenu : Le maximum obtenu :

Déterminer l'erreur relative en % :

Si l'erreur absolue d'une mesure est ε = 0,2 m sur une mesure de 40 m, alors l'erreur relative est donnée par : $\frac{40,2-40}{40} = 0,005$. L'erreur relative est donc de 0,5 %.

6) Que dire des durées de calculs ? Remplir le tableau ci-dessous en utilisant le script

Nombre de villes	**Nombre d'opérations à calculer** calcul_duree_algo.py	**Durée estimée du calcul selon les conditions du script (3 Gop • s^{-1}) en années**
17		
20		
22		
24		

Conclure sur la possibilité de trouver une solution au problème du voyageur de commerce ?

7) Comparaison du calcul du nombre d'années solution 1 et 2, voilà une partie du script :

```
# Calcul du nombre d'années solution 1
# duree_calcul = operations_a_faire / frequence_processeur
print("Bilan années méthode 1")
print(operations_a_faire / nombre_op_par_annee," ans")
print()

# Calcul du nombre d'années solution 2
while operations_a_faire >= nombre_op_par_annee:
    operations_a_faire = operations_a_faire - nombre_op_par_annee
    nombre_annee += 1
print("Bilan années méthode 2")
print(nombre_annee," ans")
```

Vous pouvez essayer de faire tourner l'algorithme pour n = 500 villes, que ce passe t-il ? Pourquoi ?

Problème du rendu de monnaie

1) Testez votre code avec différentes sommes :

Somme	Coupures rendues	
147€	1 billet de 100€ 2 billets de 20€	1 billet de 5€ 1 pièce de 2€

Rendu de monnaie glouton dans d'autres systèmes monétaires

2) Modifiez le code précédent en changeant les coupures avec les valeurs 10€, 6€ et 1€. Essayez le programme pour rendre la somme 12€. Que vous affiche le programme ?

Somme	Coupures rendues
12€	

3) Est-ce bien la solution optimale ? Expliquez pourquoi.

4) Modifiez le code précédent en changeant les coupures avec les valeurs 3€ et 2€. Essayez le programme pour rendre la somme 4€. Que vous affiche le programme ?

Somme	Coupures rendues
4€	

5) Est-ce bien la solution optimale ? Expliquez pourquoi.

Le problème du sac à dos

1) Appliquez les trois stratégies pour l'énoncé donné.

	Contenu du sac	Valeur (€)
Stratégie 1		
Stratégie 2		
Stratégie 3		

2) Y a-t-il une stratégie meilleure qu'une autre ?

Problème du jardinier

1) D'après vous, en vous appuyant sur le graphique, en déduire l'espèce de l'iris de votre ami(e) :

2) D'après vous, en vous appuyant sur le graphique, en déduire l'espèce de l'iris de votre ami(e) :

3) D'après vous, en vous appuyant sur le graphique, en déduire l'espèce de l'iris de votre ami(e). Est-ce évident ?

L'algorithme « k-NN » des k plus proches voisins

1) Avec k = 1, de quelle espèce est le nouvel iris ?

2) Avec k = 2, de quelle espèce est le nouvel iris ?

3) Avec k = 3, de quelle espèce est le nouvel iris ?

4) Conclusion sur le choix du paramètre k :