Première Algorithmique	Algorithmes gloutons *Correction*	

A - Description générale des algorithmes gloutons

B - Problème : rendu de monnaie

Exercice 1

1.

Combinaison	*Nombre de valeurs*
9 x 1 €	9
7 x 1 € + 1 x 2 €	8
5 x 1 € + 2 x 2 €	7
3 x 1 € + 3 x 2 €	6
1 x 1 € + 4 x 2 €	5
4 x 1 € + 1 x 5 €	5
2 x 1 € + 1 x 2 € + 1 x 5 €	4
2 x 2 € + 1 x 5 €	3

2.

```
euros = [1, 2, 5, 10, 20, 50, 100, 200]

def rendu_monnaie(somme : int) -> list:
    """retourne une liste de tuples (v, n) où v est la valeur
    des pièces ou billets à rendre et n le nombre"""
    i = len(euros) - 1
    a_rendre = []
    # Pas de monnaie à rendre pour l'instant
    n = 0
    monnaie = None
    while somme > 0:
        if somme >= euros[i]:
            n = n + 1 # On ajoute la valeur euros[i]
            monnaie = (euros[i], n)  # On  met à jour la monnaie à rendre
            somme = somme - euros[i] # La somme à rendre diminue
        else:
            n = 0  # On ne peut plus rendre cette valeur
            i = i - 1 # On passe à la valeur inférieure
            if monnaie != None:
                a_rendre.append(monnaie)  # On ajoute monnaie aux valeurs à rendre
                monnaie = None
    a_rendre.append(monnaie)
    return a_rendre
```

C - Problème d'optimisation : le voyageur de commerce

C.1-Explosion combinatoire

C.2-Une autre situation

Voici le tableau des distances routière entre Nancy, Metz, Paris, Reims et Troyes.

	Nancy	Metz	Paris	Reims	Troyes
Nancy	0	55	303	188	183
Metz	55	0	306	176	206
Paris	303	306	0	142	153
Reims	188	176	142	0	123
Troyes	183	203	153	123	0

Si l'on cherche à énumérer, partant par exemple de Nancy, tous les ordres possible, nous en trouverons 12 différents (et 12 autre dans le sens inverse mais avec les mêmes distances) et le meilleur est :

Nancy – Metz – Reims – Paris – Troyes avec 709 km

Exercice 2

1. Partant de Nancy, nous irons en premier à Metz, distante de 55 km, ensuite à Reims en 176 km, puis Troyes en 123 km, et enfin à Paris en 153 km, avec un dernier retour à Nancy en 303 km : le circuit comptera alors 810 km.

   Remarque : L'itinéraire obtenu est plus long que le circuit minimal de 709 km (Nancy – Metz – Reims – Paris – Troyes – Nancy) mais parmi les trois meilleurs solutions.

2.
```
villes = ("Nancy", "Metz", "Paris", "Reims", "Troyes")
dist = ((0, 55, 303, 188, 183),
        (55, 0, 306, 176, 203),
        (303, 306, 0, 142, 153),
        (188, 176, 142, 0, 123),
        (183, 203, 153, 123, 0))
```

3.
```
def plus_proche(ville : int, dist : tuple, visitees : list) -> int:
    """Retourne le numéro de la ville non encore listée la plus
    proche, en supposant qu'il en existe au moins une."""
    pp = None
    for i in range(len(visitees)):
        if not visitees[i]:
            if (pp == None or dist[ville][i] < dist[ville][pp]):
                pp = i
    return pp
```

4.
```
def voyageur(villes : tuple, dist : tuple, depart : int) -> list:
    """Retourne l'ordre des villes à visité et
    en fin de tableau la distance de ce parcours"""
    n = len(villes)
    visitees = [False] * n
    courante = depart
    distance = 0
    trajet = [villes[depart]]
    for _ in range(n-1):
        visitees[courante] = True
        suivante = plus_proche(courante, dist, visitees)
        distance = distance + dist[courante][suivante]
        courante = suivante
        trajet.append(villes[courante])
    distance = distance + dist[courante][depart]
    trajet.append(distance)
    return trajet
```

D - Problème du sac à dos

D.1- Quelques situations

Exercice 3

a)

AD	5300
BCD	7500

b)

AC	7800
AD	5300
BCD	7000

c)

AE	8900
BD	11200
BE	8000
CDE	10300

d)

AD	11800
AEF	11900
BC	12000
BDF	10100
BEF	10000
CDEF	10300

D.2- Algorithmes gloutons

Exercice 4

1.

	Stratégie a)	Stratégie b)	Stratégie c)	Optimal
Situation 1	AD 5300	DCB 7500	BCD 7500	BCD 7500
Situation 2	AC 7800	DCB 7000	AC 7800	AC 7800
Situation 3	AE 8900	EDC 10300	BD 11200	BD 11200
Situation 4	AD 11800	FEDC 10300	AEF 11900	BC 12000

2. a) Un objet lourd remplissant le sac à dos à lui seul et une multitude d'objets légers, l'objet lourd ayant une valeur légèrement supérieure aux autres. Exemple : un objet de 10 kg de valeur 10 et dix objets de 1 kg de valeur 9.

b) Une multitude d'objets légers de peu de valeur, et des objets plus lourds de grande valeur. Exemple : dix objets de 1kg et de valeur 1, et un objet de 10 kg et de valeur 100.

c) Un objet de rapport valeur/poids maximal et occupant juste un peu plus de la moitié du sac, et deux objets remplissant chacun exactement la moitié du sac et de rapport valeur/poids légèrement inférieur. Exemple : un objet de 6 kg et de valeur 60 (rapport 10) et deux objets de 5 kg et de valeur 45 chacun (rapport 9).

Dans les deux premiers cas, le rapport entre la valeur optimale et la valeur donnée par la stratégie peut être arbitrairement grand. Dans le dernier cas en revanche il ne peut être supérieur à 2.