Mobilité électrique

Les véhicules électriques se développent rapidement dans les villes.
Dans un **immeuble collectif récent**, 5 bornes de recharge doivent être installées dans le parking souterrain pour permettre aux résidents de recharger leurs voitures la nuit.



Le gestionnaire de l'immeuble dispose cependant d'une **limite technique importante** :

- la puissance électrique disponible pour le parking est limitée à **22 kW maximum**.
- chaque borne de recharge peut fournir jusqu'à **7,4 kW** à un véhicule branché.

=> Si les 5 véhicules se branchent en même temps, la puissance demandée est de :

…

Ce cas correspond à une **situation réelle** rencontrée par de nombreuses copropriétés et parkings collectifs, qui doivent mettre en place des systèmes dits de *« pilotage intelligent de la charge »*.

1) Que se passe-t-il si plusieurs voitures sont branchées en même temps ?

…
…
…

2) Quelles différentes stratégies pourraient être envisagées pour recharger plusieurs véhicules ?

…
…
…

Informations parking

Élément	Valeur / Donnée
Nombre de bornes	5
Puissance max du site	22 kW (limite à ne pas dépasser)
Puissance max d'une borne	7,4 kW (monophasé)
Capacité moyenne batterie	40 kWh
Objectif de charge	Atteindre un SoC cible (ex. 80 %) avant le départ
Formule utile	Énergie à fournir (kWh) = Capacité × (SoC_cible – SoC_initial)/100
Définition de SoC	SoC (State of Charge) = État de charge de la batterie exprimé en %. Exemple : SoC = 50 % signifie que la batterie est remplie à moitié.

PARTIE A : Pseudo-code / code python

Gestion de 5 bornes / priorité aux 1er arrivé / Max de 3 véhicules en charges simultanément

```
dispo = 22 // puissance totale dispo
P[1..5] = 0 // puissance donnée aux 5 véhicules
        POUR i de 1 à 5 FAIRE (ordre d'arrivée)
                SI dispo > 0 ALORS
                        SI dispo >= 7,4 ALORS
                                P[i] = 7,4
                                dispo = dispo - 7,4
                        SINON
                                P[i] = dispo
                                dispo = 0
                        FIN SI
                FIN SI
        FIN POUR
```

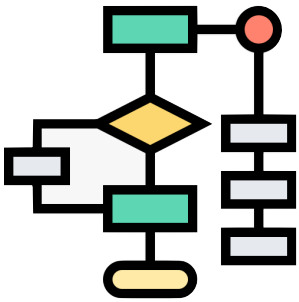


Tableau d'analyse à compléter :

Étape (i)	Puissance dispo avant	Test dispo >= 7,4 ?	Puissance donnée P[i]	Puissance dispo après
1 (V1)				
2 (V2)				
3 (V3)				
4 (V4)				
5 (V5)				

Questions de réflexion :

1. Quels véhicules ont pu être rechargés en priorité ? Pourquoi ?

2. Cette stratégie est-elle équitable pour tous les usagers ? Quels seraient ses avantages et ses limites ?

3. Pour une recharge équitable, qu'elle serait la puissance à fournir par la borne ?

4. Ecrire un Pseudo-code de gestion du la recharge avec une logique de puissance équitable entre les 5 véhicules

Code PYTHON

Gestion de 5 bornes / priorité aux 1er arrivé / Max de 3 véhicules en charges simultanément

5. Taper le code python et vérifiez son bon fonctionnement

```python
# Puissance totale disponible (kW)
dispo = 22.0

# Initialisation : 5 véhicules avec 0 kW au départ
P = [0, 0, 0, 0, 0]

# Attribution des puissances selon l'ordre d'arrivée
for i in range(5):
    if dispo > 0:
        if dispo >= 7.4:
            P[i] = 7.4
            dispo -= 7.4
        else:
            P[i] = dispo
            dispo = 0

# Affichage du résultat
for i in range(5):
    print(f"Véhicule {i+1} : {P[i]} kW")

print(f"Puissance restante : {dispo} kW")
```

6. Modifier le programme de gestion de la recharge avec une logique de puissance équitable entre les 5 véhicules

PARTIE B : Communication Voiture / Borne par trames binaires

Quand deux systèmes communiquent (ici, une **borne de recharge** et une **voiture électrique**), ils ne s'envoient pas des phrases en français mais des **messages codés en binaire** (0 et 1).

Ces messages sont organisés sous forme de **trames**. Une trame est comme une petite enveloppe contenant plusieurs parties :

- un **identifiant (ID)** → pour savoir qui envoie (borne ou voiture)
- un **type** → pour indiquer la nature du message (démarrage, arrêt, niveau de charge)
- une **donnée** → pour transmettre l'information utile (par exemple : le SoC = état de charge de la batterie)

On peut comparer une trame à une **lettre postale** : l'expéditeur (ID), l'objet de la lettre (Type) et le contenu (Donnée). Sans cette structure, la borne et la voiture ne pourraient pas se comprendre.

Dans notre activité, nous allons utiliser des trames simplifiées de **8 bits (1 octet)**, découpées ainsi :

Champ	Taille	Exemple	Rôle
ID	2 bits	01 = borne / 10 = voiture	Qui envoie le message ? (borne/voiture)
Type	2 bits	Message (00 = start, 01= stop, 10 = SoC	Quel type de message ? (START / STOP / SOC)
Donnée	4 bits	Niveau de SocC (0 à 100 %)	Information utile (ici : niveau de charge en %)

Activité 1 : Construire l'échelle de codage

1. Combien de codes peuvent être attribués avec une précision de codage à 4 bits ?

2. Complétez le tableau de l'échelle de codage du niveau de charge du SoC sur 4 bits

SoC (%)	Valeur décimale	Code binaire (4 bits)
0 %	0	0 0 0 0
25 %	...	...
50 %	...	...
75 %	...	...
100 %	...	...

Activité 2 : Décoder une trame => Trame reçue : 10101000

- Identifiez l'émetteur (ID)
- Le type de message
- Le niveau de SoC (en %)

Activité 3 : Décrire une séquence => La borne reçoit successivement

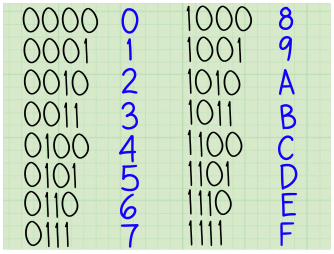
01000000 – 10100101 – 01010000
Expliquez ce qu'il s'est passé étape par étape.

Activité 4 : Encoder un message

Écrivez la trame complète (8 bits) que la voiture enverrait pour signaler SOC = 75 %.

PARTIE C :Trames et Hexadécimal

Les machines communiquent en binaire (0 et 1). Mais écrire et lire de longues suites de bits est difficile. Par exemple : 10101000 est une seule trame (8 bits), mais pas très lisible pour un humain.



=> Pour simplifier, on utilise l'hexadécimal (base 16) :
- Plus compact : 1 octet = 2 chiffres hexa
- Plus lisible, utilisé partout en informatique (réseaux, mémoire, couleurs HTML…)
- Conversion facile : 4 bits = 1 chiffre hexa

Conversion binaire ↔ hexadécimal

Binaire (4 bits)	Décimal	Hexa
0000	0	0
1000	8	8
1010	10	A
1100	12	C
1111	15	F

Exemples de trames
- START (borne) : 01000000→ 40
- SOC = 50 % (voiture) : 10101000 → A8
- STOP (borne) : 01010000 → 50

Activité 1 — Conversion binaire → hexadécimal

Complétez le tableau suivant :

Trame binaire	Trame hexadécimale	Signification
01000000	…	START (borne)
10101000	…	SOC 50 % (voiture)
01010000	…	STOP (borne)
10101111	…	SOC 100 % (voiture)

Activité 2 — Encoder une trame en hexadécimal

La voiture envoie son état de charge : SoC = 75 %.
1. Écrivez la trame en binaire (8 bits).
2. Convertissez-la en hexadécimal (2 chiffres).

Activité 3 — Réflexion

Expliquez en une phrase pourquoi l'hexadécimal est plus pratique que le binaire pour représenter les trames.

PARTIE D : Trames et codage ASCII

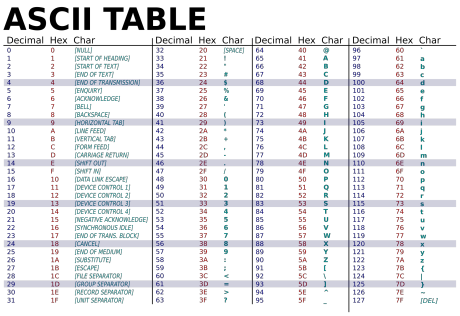
Rappel : qu'est-ce que l'ASCII ?

Les ordinateurs ne comprennent que des 0 et 1. Pour transmettre du texte (lettres, chiffres, symboles…), il faut donc traduire chaque caractère en un nombre entier. ▪ C'est ce que fait le **code ASCII** (American Standard Code for Information Interchange).

- Chaque caractère est associé à un nombre.
- Ce nombre est représenté en **binaire sur 8 bits** (un octet).
- Exemple :

• '0' → 48 en décimal → 00110000 en binaire
• '1' → 49 en décimal → 00110001 en binaire
• '%' → 37 en décimal → 00100101 en binaire

Ainsi, quand on envoie le message « 50% », la machine ne voit pas le texte mais : 00110101 00110000 00100101



Champ	Taille	Exemple	Rôle
ID	2 bits	10	Qui envoie ? (voiture)
Type	2 bits	10	Quel type de message ? (SOC)
Niveau SoC	4 bits	1000	Valeur numérique du SoC (ici 50 %)
Message	ASCII	"50%" → 00110101 00110000 00100101	Texte affiché sur la borne

Activité 1 — Complétez le tableau

Caractère	ASCII décimal	ASCII binaire (8 bits)
0	48	00110000
1	…	…
%	…	…

Activité 2 — Encoder une trame complète

La voiture envoie SOC = 100 % avec le message « 100% ».
1. Écrivez le champ Niveau SoC en binaire (4 bits).
2. Codez le message « 100% » en ASCII binaire (suite d'octets).
3. Construisez la trame finale en concaténant tous les champs.

Étape 3 — Décoder une séquence

La borne reçoit :
01000000 – 10101111 00110001 00110000 00110000 00100101 – 01010000

Expliquez ce qu'il s'est passé étape par étape.