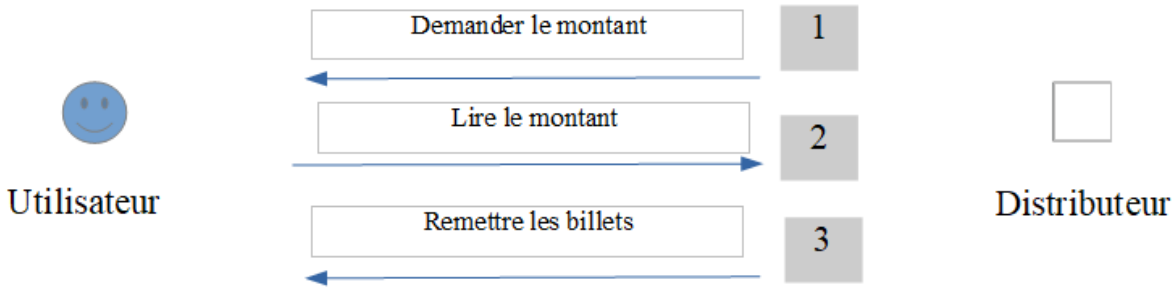
TP n°1

Rappel : entrées et sorties en JS

Pour rappel, pendant le S1, vous avez utilisé les instructions `entrerNB()` et `entrerCH()` pour retourner ce qui a été tapé au clavier, ainsi que `sortir(message)` pour afficher les résultats.
Ces instructions se trouvent dans le fichier **templateAlgo.txt** (disponible en début de ce cours su rmoodle). Vous pouvez aussi utiliser les instructions équivalentes :

Instruction utilisée en S1	Instruction équivalente
`let nb ;` `nb=entrerNB("taper une valeur : ") ;`	`let nb ;` `nb=Number(prompt ("taper une valeur : "));`
`let rep ;` `rep=entrerCH("taper une chaîne : ") ;`	`let rep ;` `rep=String(prompt("taper une chaîne : ")) ;`
`sortir("le résultat est " + nb) ;`	`console.log("le résultat est " + nb) ;`

Exercice 1 :



On souhaite écrire le script d'un distributeur de billets : l'utilisateur indique le montant qu'il souhaite retirer, le distributeur calcule le nombre de billets de chaque type qu'il doit remettre puis l'affiche.

On suppose qu'au départ, le distributeur dispose de 10 billets de 10 euros, 10 billets de 20 euros, 10 billets de 50 euros et 3 billets de 100 euros.

On suppose également que la somme demandée est toujours un multiple de 10 : par exemple, pour 220 euros, le distributeur affichera le message « 2 billet(s) de 100 euros, 0 billet(s) de 50 euros, 1 billet(s) de 20 euros et 0 billet(s) de 10 euros ».

Écrire le script et le tester avec les sommes 220, 1100 (3 billet(s) de 100 euros, 10 billet(s) de 50 euros, 10 billet(s) de 20 euros et 10 billet(s) de 10 euros) et 1120 euros (somme impossible à distribuer).

Dans cette première version, la caisse n'est pas mise à jour c'est-à-dire que le nombre de billets disponibles n'est pas modifié (10 billets de 10 euros, 10 billets de 20 euros, 10 billets de 50 euros et 3 billets de 100 euros).

Exercice 2 :

Avant de distribuer les billets, on demande à l'utilisateur de s'authentifier. Ajouter en début de script le code pour :

- demander le numéro de carte et le code secret de l'utilisateur,
- tester la validité des informations.
- Si le test échoue, l'utilisateur est invité à recommencer.

Dans cette première version, il n'y a que 2 numéros de carte autorisés : 0 avec le code 2020 et 1 avec le code 2021.

Exercice 3 :

Modifier le script pour que l'utilisateur ne puisse faire que 3 tentatives : après 3 tentatives infructueuses, le script se termine sans distribuer de billets.

Exercice 4 :

Les numéros et les codes de carte possibles sont maintenant enregistrés dans un tableau. Par exemple :

```
let tabCarte=[{num:0,code:2020},{num:1,code:2021}, {num:3,code:2023},
{num:10,code:2040}];
```

Chaque case du tableau contient un objet littéral ayant 2 attributs `num` et `code`. Par exemple,

`tabCarte[2].num` vaut `3` et `tabCarte[2].code` vaut `2023`

Modifier le script pour qu'il vérifie que le numéro de carte et le code à partir de ce tableau (indication : cela revient à chercher un élément dans le tableau).

Exercice 5 :

Le script comprend une partie de script dupliquée 4 fois : calcul du nombre de billets pour chaque valeur de billets (100, 50, 20 et 10). Utiliser la fonction `nbBillets()` ci-dessous pour éviter la duplication du code :

```
// role : indique le nombre de billets à rendre
 // en fonction de la somme demandée et du nombre de billets disponibles
 // paramètres : somme, la somme demandée
 //              type, la valeur du billet (100, 50, 20 ou 10)
 //              N, le nombre de billets disponibles pour cette valeur
 // retour : le nombre de billets à distribuer
function nbBillets(somme, type, N) {
    let nb= parseInt(somme / type);
    if (nb > N) {
        nb=N;
    }
    return nb;
}
```

Exercice 6 :

Modifier le script pour mettre à jour la caisse (c'est-à-dire enlever du stock les billets distribués) et permettre à l'utilisateur de faire plusieurs retraits (à condition que la caisse dispose encore de billets).