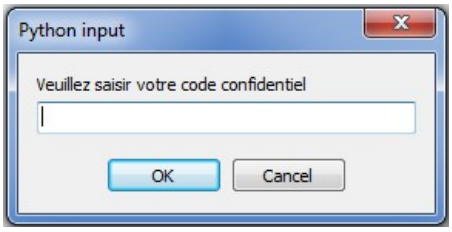
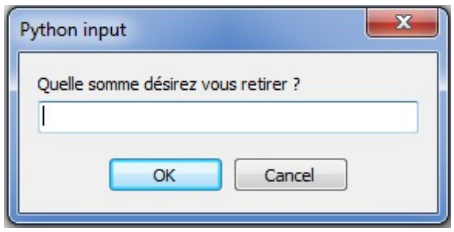
Miniprojet – Distributeur de billets

- Créer un programme qui simule un distributeur de billets
- Savoir le faire évoluer.

1ÈRE ÉTAPE :





Voici ce que doit pouvoir faire votre programme :

Au lancement du programme , le message suivant s'affiche :

Si l'utilisateur saisit un autre code que "1234" ,
le programme affiche dans l'interpréteur : **« Opération annulée ! »**

Sinon , l'utilisateur est invité à saisir le montant désiré.
le programme affiche alors dans l'interpréteur : (*par exemple*)
« Vous avez demandé 50 euros , veuillez retirer vos billets. »

Remarque :

Vous devriez avoir besoin de 3 variables : `essai , code , montant`

2ÈME ÉTAPE :

Améliorez le programme précédent afin qu'il demande 3 fois (*pas plus*) le code confidentiel lorsque l'utilisateur se trompe .
- En cas d'erreur il affichera dans l'interpréteur : **« Erreur , il vous reste 2 essais... »**
- Il faudra également que le programme contrôle que la somme demandée est entre 0 et 700 euros , là encore l'utilisateur ne doit disposer que de 3 essais sinon le programme affiche **« Opération annulée »**

Remarque :

Pour ceux qui connaissent (*pensez aux **fonctions***)

3ÈME ÉTAPE :

On définit en début de programme les variables :

```
utilisateur_1='Trucmuche' ; code_1='1234';compte_1=500 ;
utilisateur_2='Bidule';code_2='4321';compte_2=300
```

- Modifier le programme afin qu'il identifie quel est l'utilisateur qui a saisi son code pour afficher par exemple :
« Bonjour Bidule , vous disposez de 300 euros . Quelle montant désirez vous retirer ? »

- Après la saisie d'un montant correct , il affichera : **« Opération réussie . Il vous reste …. euros . »**

Remarque :

Pour pouvoir utiliser le montant saisi par l'utilisateur dans des calculs il faut qu'il soit identifié comme un nombre (entier par exemple) , pour cela on utilisera l'instruction : `int(input('....')` pour obtenir une variable enregistrée en tant que nombre et non plus comme une chaîne de caractères.

4ÈME ÉTAPE :

- Modifier le programme afin qu'il reste actif après une opération réussie ou échouée, c'est à dire qu'un autre utilisateur (*ou le même*) pourra refaire une autre opération .

- Le programme devra être capable d'indiquer à l'utilisateur qu'il ne lui reste plus d'argent ou qu'il ne peut pas retirer plus qu'un certain montant...

5ÈME ÉTAPE :

On veut maintenant gérer un nombre plus important d'utilisateurs en définissant 10 utilisateurs en début de programme .On utilise pour cela 3 **tableaux** (appelés "**listes"** en Python) l'un contenant les noms , un autre les codes confidentiels et le dernier, les comptes, voir ci-dessous :

```
# Déclaration des deux tableaux/listes : contrairement aux autres variables sous Python , il faut déclarer
# les variables tableaux pour les utiliser
# Ils sont donc d'abord construits avec la réservation de 10 éléments vides   ''  pour chacun :
nom=['']*10  ;  code=['']*10   ;   compte=['']*10

# Puis on les "remplit" élément par élément.
#(on peut utiliser le guillemet double _"_   ou le simple _'_ pour encadrer une chaine de caracteres  )

nom[0] = "Alice" ;            code[0] = "0606" ; compte[0]=1000
nom[1] = "Bob" ;              code[1] = "0607" ; compte[1]=1000
nom[2] = "Charles" ;          code[2] = "0608" ; compte[2]=1000
nom[3] = "Djamel" ;           code[3] = "0609" ; compte[3]=1000
nom[4] = "Etienne" ;          code[4] = "0610" ; compte[4]=1000
nom[5] = "Frederique" ; code[5] = "0611" ; compte[5]=1000
nom[6] = "Guillaume" ; code[6] = "0612" ; compte[6]=1000
nom[7] = "Hector" ; code[7] = "0613" ; compte[7]=1000
nom[8] = "Isabelle" ; code[8] = "0614" ; compte[8]=1000
nom[9] = "Jerome" ; code[9] = "0615" ; compte[9]=1000
```

Le code **code[i]** est le code confidentiel de la personne dont le nom est **nom[i]** ; **compte[i]** donne la somme de cette même personne . Ainsi le **[i]** (appelé **index**) fait correspondre les éléments de chaque tableau concernant une même personne.

Modifiez votre programme afin qu'il gère ces 10 utilisateurs.

Remarque :

Pour inclure cette liste au début de votre programme faire un copier-coller de l'extrait ci-dessus.

6ÈME ÉTAPE :

On décide d' « externaliser » la table d'utilisateur en l'écrivant dans un fichier à part , de façon à ne pas perdre les enregistrements de montants qui ont été modifiés . Ce fichier s'appellera « **livre_compte.txt** »

Comme tous les fichiers lus à partir d'un programme dialoguant avec le Système de Gestion des Fichiers de l'ordinateur , "**`livre_compte.txt`**" sera lu du premier octet jusqu'au dernier **sans possibilité de retour en arrière avant la fermeture** du fichier.

Le script suivant vous montre comment stocker dans une liste « liste_lignes » chacune des lignes du fichier s'il est enregistrée à la racine du lecteur « C:\ » .
Puis comment réécrire le fichier de manière à enregistrer un nouvel utilisateur en plus :
Noemie de code confidentiel '0616' qui a un montant de 1000 euros sur son compte.

```
# ouverture du fichier :
fichier_lu=open('C:\livre_compte.txt','r')
# récupération du contenu dans la liste "liste_lignes" :
liste_lignes=fichier_lu.readlines()
# on referme le fichier  : un autre programme pourra donc intervenir sur ce fichier
fichier_lu.close()

# ajout de trois lignes pour Noemie en fin de liste_lignes ('\n' permet de coder le retour à la ligne ! )  :
liste_lignes=liste_lignes+['Noemie\n0616\n1000\n']
# Ecriture ligne par ligne dans le fichier
fichier_ecrit=open('C:\livre_compte.txt','w'))
for ligne in liste_lignes:
    fichier_ecrit.write(ligne)
fichier_ecrit.close()
```

Modifier le programme de manière à lire les données dans le fichier « **livre_compte.txt** » et à pouvoir commander l'enregistrement d'un nouvel utilisateur lorsque , à la place d'un code confidentiel , on saisit 'new'.