Nom Prénom : ..... Heure de passage : .....

BTS SIO
Sous-épreuve E22
Algorithmique appliquée
Contrôle en Cours de Formation

Déroulement de l'épreuve

Cette épreuve de Contrôle en cours de Formation (CCF) se déroule en trois étapes :

- **Écrit (30 minutes)**

  Vous devez traiter l'étape 1 du sujet. Pour cette partie, l'ordinateur est interdit mais la calculatrice est autorisée.

  Vous inscrirez vos réponses dans le document réponse à la fin du sujet.

  Les algorithmes à écrire peuvent être rédigés en **langage naturel** ou en Python mais ni en C# ni en VB.Net.

  À la fin de l'étape 1, votre document réponse doit être remis à la personne surveillant l'épreuve. Vous garderez le sujet.

- **Machine (30 minutes)**

  Vous devez traiter l'étape 2 du sujet à l'aide d'un ordinateur. Le langage utilisé est celui travaillé dans l'année, à savoir Python. Vous sauvegarderez votre travail sur la clé USB fournie.

  La durée totale pour effectuer les deux premières étapes est exactement d'une heure.

- **Oral (20 minutes au maximum)**
  Cette partie se déroule en deux temps. Tout d'abord, vous disposez de 10 minutes pour présenter votre travail de l'étape 2 puis, au cours des 10 minutes suivantes, un entretien permet de préciser votre démarche.

À la fin de l'épreuve le sujet devra être rendu à l'examinateur.

Rendu de monnaie

Étape 1

On dispose d'une liste de pièces et billets de monnaie rangée dans l'ordre décroissant. Pour faire simple on appellera « pièce » une pièce ou un billet sans faire la distinction.
Voici la variable `piece` ainsi que sa valeur :

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

Dans cette situation, pour rendre une certaine somme en le moins de pièces possible, c'est simple : il faut commencer par rendre la plus grosse pièce possible et recommencer tant qu'il reste de l'argent à rendre.

Par exemple $13 = 10 + 2 + 1$ et cette égalité indique qu'on rend 13€ en 3 pièces.

Question 1

> Tu dois rendre 26€. Écris l'égalité correspondant au rendu de pièces.
> Combien de pièces faut-il pour rendre cette somme ?

Question 2

> Même question avec 267€.

On a commencé à écrire une fonction `trouve_piece` qui

- en entrée prend un entier `somme` qui est la somme d'argent à rendre ;
- renvoie le plus grand élément de la liste `piece` que l'on peut rendre

Par exemple `trouve_piece(49)` vaut 20 car 20 est la plus grosse pièce qu'on peut rendre quand on veut rendre 49€.

Question 3

> Sur ta copie, complète l'algorithme de la fonction `trouve_piece` :
>
> ```
> fonction trouve_piece(somme)
>     variables
>         resultat, i, n : entiers
>         trouve : booleen
>
>     resultat ← 0
> ```

```
    trouve ← faux
    i ← 0
    n ← longueur(...)
    tant que i < n et trouve = faux repeter
        si piece[i] ≤ ... alors
            resultat ← ...
            trouve ← vrai
        fin si
        i ← ...
    fin tant que
    renvoyer resultat
```

On a ensuite écrit la fonction `nb_pieces_rendues` qui

- en entrée prend un entier `somme` qui est la somme d'argent à rendre ;
- renvoie le nombre de pièces nécessaires pour rendre la somme à l'aide des pièces de la liste `piece`.

Par exemple `nb_pieces_rendues(13)` vaut 3 puisqu'on rend 13€ en 3 pièces.

Question 4

Sur ta copie, complète l'algorithme de la fonction `nb_pieces_rendues` :

```
fonction trouve_piece(somme)
    variables
        resultat: entier

        resultat ← ...
    tant que somme > 0 repeter
        somme ← ...
        resultat ← resultat + 1
    fin tant que
    renvoyer resultat
```

On suppose maintenant que `piece = [4,3,1]`

Question 5

a. Que renvoie alors `nb_pieces_rendues(6)` ?

b. Peut-on rendre la valeur 6 avec moins de pièces ?

c. L'algorithme utilisé est-il optimal dans ce cas ?

Étape 2

Question 6

> Ouvrir le fichier `rendu_monnaie.py` et compléter les fonctions.

Question 7

> Écrire la fonction `liste_pieces_rendues qui`
>
> – en entrée prend un entier `somme` qui est la somme d'argent à rendre ;
>
> – renvoie la liste des pièces nécessaires pour rendre la somme à l'aide des pièces de la liste `piece`.
>
> Par exemple `liste_pieces_rendues(13)` vaut `[10, 2, 1]`.
> Autre exemple : `liste_pieces_rendues(43)` vaut `[20, 20, 2, 1]`.