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 :

- **Étape 1 : Écrit (30 minutes)**

  Vous devez traiter la partie A 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.

- **Étape 2 : sur machine (30 minutes)**

  Vous devez traiter la partie B 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.

- **Étape 3 : 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.

Nombre de diviseurs d'un entier

Étape 1

On s'intéresse au nombre de diviseurs qu'un entier peut avoir : on veut fabriquer une fonction `nb_div` qui

- en entrée prend un `int` non nul `n` ;
- renvoie le nombre de diviseurs de `n`.

Question 1

> Que devra renvoyer `nb_div(12)` ? `nb_div(49)` ?

On a implémenté la fonction en PYTHON :

Python

```
def nb_div( n : int) -> int:
    r = ......
    for i in range(......):
        if n % i == ......:
            r = ......
    return r
```

Question 2

> Compléter sur votre copie les pointillés pour que la fonction remplisse son rôle.

On souhaite construire une fonction `nb_div_tous_entiers` qui :

- en entrée prend un `int` non nul `n` ;
- renvoie la liste de tous les diviseurs de 1, 2, 3 jusqu'à `n`.

Par exemple `nb_div_tous_entiers(6)` renverra
`[1, 2, 2, 3, 2, 4]` car

- 1 n'a qu'un diviseur ;
- 2 en a 2 ;
- 3 en a 2 ;
- *et cætera*.

Question 3

> Écrire la fonction `nb_div_tous_entiers`.

On dispose de la fonction `mystere` suivante :

Python

```python
def mystere(n : int) -> int:
    m = 0
    r = 0
    for i in range(1, n+1):
        if nb_div(i) > m:
            m = nb_div(i)
            r = i
    return r
```

Question 4

> Que fait la fonction `mystere` ?
> Que renvoie `mystere(10)` ?

Étape 2

Les fonctions de l'étape précédente ont été implémentées en PYTHON dans le fichier `nombre_diviseurs`

Question 5

> Écrire la fonction `nb_div_connus` qui
>
> - en entrée prend un entier `n` non nul ;
> - renvoie le plus petit entier qui a `n` diviseurs.
>
> Par exemple `nb_div_connus(4)` devra renvoyer 6 car 6 est le plus petit entier qui a 4 diviseurs.

Question 6

> Écrire la fonction `liste_div_connus` qui
>
> - en entrée prend un entier 3 entiers `a`, `b` et `n` avec `0 < a < b` et `n` non nul ;
> - renvoie la liste des entiers compris entre `a` et `b` **inclus** qui ont exactement `n` diviseurs