NOM :

On importe les fonctions suivantes :
from math import floor, sqrt

1. **Test :**
   Un vendeur travaille dans une boutique avec le contrat suivant. S'il effectue moins de 30 heures de travail, il est payé 10 euros de l'heure. Entre 31 heures et jusqu'à 40 heures, on lui verse 5 euros de plus par heure. Enfin, s'il dépasse ce quota, ses heures supplémentaires sont payées doubles. Ecrire une fonction qui détermine son salaire hebdomadaire en fonction du nombre d'heures h effectuées par notre vendeur.

```
1 def salaire(h) :
2     """ Cette fonction renvoie le salaire hebdomadaire du vendeur qui a effectué h heures dans la boutique """
3
4
5
6
7
8
9
```

2. **Boucles :**

•On considère la suite $H_n = \sum_{k=1}^{n} \frac{1}{k}$. On peut montrer que $H_n \underset{n\to+\infty}{\sim} \ln(n)$ et $\lim_{n\to+\infty} H_n = +\infty$.
Par suite $\forall p \in \mathbb{N},\ \exists n_0 \in \mathbb{N},\ \forall n \geq n_0,\ H_n > p$.
Ecrire une fonction Python, qui étant donné un entier $p \in \mathbb{N}$ renvoie le premier entier $n \in \mathbb{N}$ tel que $H_n \geq p$.

```
1 def harmonique(p) :
2     """ Cette fonction renvoie le premier entier n tel que H_n > p """
3
4
5
6
7
```

•Ecrire la fonction factorielle qui étant donné un entier n renvoie n !

```
1 def factorielle(n) :
2     """ Cette fonction renvoie n ! """
3
4
5
6
```

•Ecrire la fonction hyperfactorielle qui étant donné un entier n renvoie l'hyperfactorielle de n, à savoir :
$hf(n) = \prod_{k=1}^{n} k! = 1! \times 2! \times 3! \times \cdots \times n!$

```
1 def hyperfactorielle(n) :
2     """ Cette fonction renvoie hf(n) """
3
4
5
6
```

3. **Autour des listes de nombres :**

•Aïe, la fonction sum semble avoir un bug. A vous de compléter la fonction sommeliste qui la remplace :

```
1 def sommeliste(L) :
2     """ Cette fonction calcule et renvoie la somme des éléments d'une liste L donnée """
3     s=0 # initialisation
4
5
6
```

•On définit une distance d entre deux listes L_1 et L_2 de nombres de même taille n en posant :
$d = Max\{|L_1[i] - L_2[i]|\ ,\ 0 \leq i \leq n-1\}$.
Compléter la fonction suivante qui renvoie ce nombre d pour deux listes L_1 et L_2 données. On pourra utiliser la fonction *abs* de PYTHON qui renvoie la valeur absolue d'un réel, mais pas la fonction *max*.

```
1 def distanceliste(L1, L2) :
2     """ Cette fonction détermine et renvoie le maximum des réels |L1(i) - L2(i)| """
3     n=len(L) # taille commune des listes L1 et L2
4     maxi=abs(L1[0] - L2[0]) # maximum provisoire
5
6
7
8     return maxi
```

Que donne distanceliste($[5, 4, -2], [3, 8, 3]$) ?

4. **Diviseurs et nombres premiers :**

•Expliquer ce que fait la fonction suivante : pour ce faire, on remplira la ligne de commentaire de la fonction :

```
1 def listediviseur(n) :
2     """
3                                                                    """
4     L=[1]
5     for k in range(2,n) :
6         if n%d==0 :
7             L.append(d)
8     return L
```

Que renvoie listediviseur(12) ?

•Un nombre est parfait s'il est égal à la somme de ces diviseurs stricts, par exemple 6 est parfait car $6 = 1 + 2 + 3$.
Compléter la fonction suivante qui renvoie la liste des nombres parfaits entre 2 et un entier n donné.

```
1 def nombresparfaits(n) :
2     """ Cette fonction renvoie la liste des nombres parfaits inférieurs à n. """
3     L=[ ] # initialisation de la liste vide
4
5
6
7     return L
```

La liste renvoyée par nombresparfaits(100) contiendra-elle 15 ? 18 ? 28 ? 36 ? 63 ? 100 ? 496 ?