TD

Savoirs et compétences :

- ❒ *AA.C4 : Comprendre un algorithme et expliquer ce qu'il fait*
- ❒ *AA.C5 : Modifier un algorithme existant pour obtenir un résultat différent*
- ❒ *AA.C6 : Concevoir un algorithme répondant à un problème précisément posé*
- ❒ *AA.C7 : Expliquer le fonctionnement d'un algorithme*
- ❒ *AA.C8 : Écrire des instructions conditionnelles avec alternatives, éventuellement imbriquées*
- ❒ *AA.S8 : Instructions conditionnelles*
- ❒ *AA.S9 : Instructions itératives*

1 Applications

1.1 Structures de boucles

Q 1 : Calculer 2^9 à l'aide d'une boucle itérative.
Q 2 : Écrire un algorithme affichant la table de multiplication de 9.
Q 3 : Calculer $16!$ à l'aide d'une boucle itérative.
Q 4 : Calculer

$$\sum_{k=0}^{15} \frac{1}{k!}$$

Q 5 : Écrire une fonction calculant le nombre de chiffres d'un entier écrit en base 10.
Q 6 : On considère la suite u définie par

$$\forall n \in \mathbb{N}^* \quad u_n = \sum_{k=1}^{n} \frac{1}{\sqrt{k}}$$

Quel est la plus petite valeur n pour laquelle $u_n \geq 1000$?
Q 7 : Écrire une fonction trouvant le plus petit nombre premier supérieur ou égal à un entier donné.
Q 8 : Écrire une fonction calculant le nombre de diviseurs d'un entier n donné.
Q 9 : Calculer p_5/q_5 où p et q sont définies par :

$$\begin{aligned} p_0 &= 1 \\ q_0 &= 1 \\ \forall n \in \mathbb{N} \quad p_{n+1} &= p_n^2 + 2q_n^2 \\ \forall n \in \mathbb{N} \quad q_{n+1} &= 2p_n q_n \end{aligned}$$

1.2 Instructions conditionnelles

Q 1 : Définir la fonction f qui à x associe $\begin{cases} 2 & \text{si} \quad x \in [-4,-2] \\ -x & \text{si} \quad x \in [-2,0] \\ 0 & \text{si} \quad x \in [0,4] \end{cases}$

Q 2 : Ecrire une fonction calculant le produit des entiers impairs de 1 à $2n+1$.
Q 3 :

```
if condition :
    bloc d instructions 1
else :
    bloc d instructions 2
```

a) **Écrire une fonction** `bloc d instructions 2` **qui renvoie la négation du booléen sans utiliser** `.`

```
if condition 1 :
    bloc d instructions 1
elif condition 2 :
    bloc d instructions 2
elif condition 3 :
    bloc d instructions 3
.
.
.
else :
```

b) **Écrire une fonction** `bloc final` **qui renvoie le ou logique des booléen et sans utiliser** `for variable in liste :` `instructions` **ni**

```
print('Bonjour Baptiste')
print('Bonjour Lisa')
print('Bonjour Pierrick')
print('Bonjour Louise-Eugénie')
print('Bonjour Qasim')
print('Bonjour Lorenzo')
print('Bonjour Arthur')
print('Bonjour Ylies')
```

sans utiliser

```
for x in prenoms:
    print('Bonjour ' + x).
```

```
while condition :
```

c) **Écrire une fonction** `instructions` **qui renvoie le et logique des booléen et** `neg(b)` **sans utiliser** `b`, `not` **ni** `ou(a,b)`.

```
def est_proche(x):
    """x est proche de 3
    distance = abs(x-3)
    if distance <= 10**(-
        return True
    else :
        return False
```

sans utiliser

```
def est_proche(x):
    """x est proche de 3 à 10**-4 près ?"""
    distance = abs(x-3)
    return distance <= 10**(-4)
```

.

```
def max3 (a, b, c) :
    """ renvoie le maxi
    précondition : a, ,
    if a <= c and b <=
        return c
    elif a <= b and c <
        return b
    else :
        return a
```

et

```
prenoms = [ 'Baptiste', 'Lisa', 'Pierrick',\
'Louise-Eugénie', 'Qâsim', 'Lorenzo', 'Arthur', 'Y
```

ni

```
def f(n):
    """Fonction de Syracuse.
    Précondition : n est un enti
    if n % 2 == 0:
        return n // 2
    else:
        return 3 * n + 1

def syracuse(n):
    """Renvoie le premier entier
    Précondition : n est un enti
    x = n
    k = 0
    while x != 1:
        x = f(x)
        k = k + 1
    return k
```

1.3 Boucles "while"

Q 1 : Que fait la fonction suivante ? La corriger pour qu'elle coïncide avec le but annoncé.

```
a
```