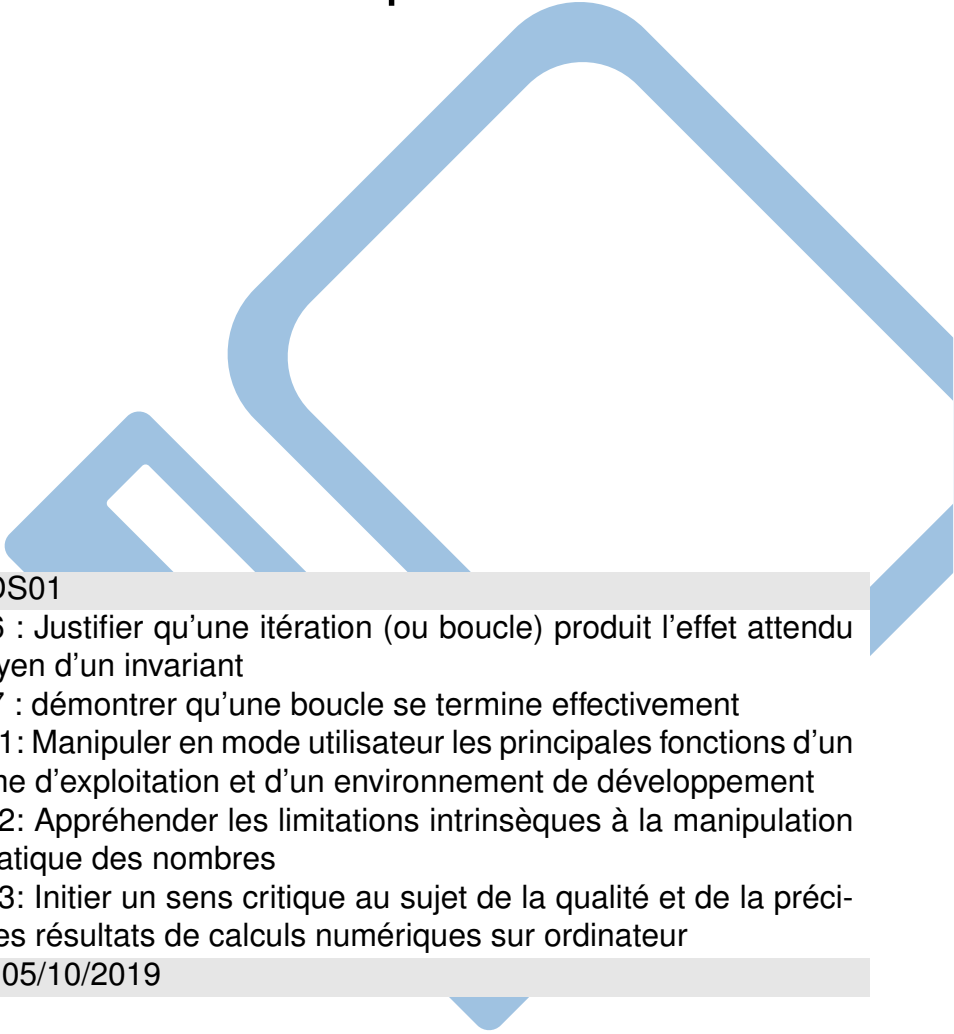


Avec Correction

DS01 Informatique



Référence	S01- DS01
Compétences	Alg-C6 : Justifier qu'une itération (ou boucle) produit l'effet attendu au moyen d'un invariant Alg-C7 : démontrer qu'une boucle se termine effectivement Déc-C1: Manipuler en mode utilisateur les principales fonctions d'un système d'exploitation et d'un environnement de développement Déc-C2: Appréhender les limitations intrinsèques à la manipulation informatique des nombres Déc-C3: Initier un sens critique au sujet de la qualité et de la précision des résultats de calculs numériques sur ordinateur
Description	Fait le 05/10/2019

1 Introduction

Question 1 Ecrire sur le diagramme de Contexte donné en document réponse le nom des composants de l'unité centrale.

2 Analyse d'une réponse temporelle

Le tracé de la figure 1 correspond à la réponse $s(t)$ à un échelon $u(t) = 1$ de la fonction de transfert $H(p)$, avec :

$$H(p) = \frac{K}{1 + \frac{2\xi}{\omega_0} p + \frac{p^2}{\omega_n^2}}$$

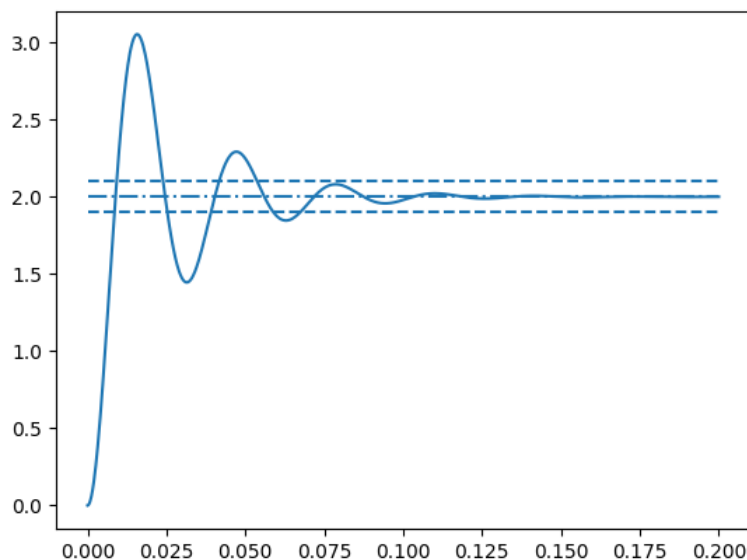


FIGURE 1 – Tracé de la réponse temporelle $s(t)$

Sont indiquées sur la figure l'asymptote à la courbe quand t tend vers l'infini (en trait mixte) et les droites limites à $\pm 5\%$ de cette asymptote (en pointillés).

L'objectif de cette étude est d'écrire un script python permettant d'identifier les paramètres ξ , ω_0 et K de cette fonction de transfert.

Afin d'effectuer ce tracé, deux listes ont été créées :

- t contenant l'ensemble des valeurs de t ,
- s contenant l'ensemble des valeurs de $s(t)$.

Rappel : `L[-1]` donne la dernière valeur de la liste `L`.

Question 2 Ecrire sous python la commande permettant de déterminer K à partir de s .

On donne la fonction python `recherche(s)` ci-dessous qui recherche la valeur `val` dans la liste `s`.

```
def recherche(s):
    val=s[0]
    for i in range(len(s)):
        if s[i] > val:
            val=s[i]
    return val
```

Question 3 Préciser quelle est la particularité de cette valeur `val` pour la liste `s`.

Question 4 Justifier que le code suivant permet de calculer T_p la valeur de la pseudo période de $s(t)$.

```
i=0
while s[i]!=recherche(s):
    i+=1
Tp=2*t[i]
```

On rappelle que le dépassement en pourcent défini comme suit : $D\% = 100 \cdot \frac{s(max)-s(+\infty)}{s(+\infty)}$ peut être calculé en fonction de ξ comme suit $D\% = 100 \cdot e^{-\frac{\xi \cdot \pi}{\sqrt{1-\xi^2}}}$, on a donc défini la fonction $D(xi)$ suivante :

```
def D(xi):
    return 100*np.exp(-xi*np.pi/np.sqrt(1-xi**2))
```

Afin de rechercher la valeur de ξ , on propose de suivre l'algorithme suivant :

1. On sait que $\xi < 0.7$ car il y a un dépassement au dessus de la droite à $+5\%$, donc on démarre avec $xi=0.7$,
2. On regarde si $D(xi)$ est plus petit que $D\% = 100 \cdot \frac{s(max)-s(+\infty)}{s(+\infty)}$,
3. Tant que c'est vrai, cela signifie que le ξ est inférieur à xi , alors on retire 0.01 à xi ,
4. Quand ce n'est plus vrai, cela signifie qu'on vient de trouver ξ .

Question 5 Compléter le code python suivant afin de déterminer ξ à partir des résultats précédents.

```
xi=0.7
while .....:
    xi+=-0.01
```

On rappelle que $T_p = \frac{2 \cdot \pi}{\omega_0 \cdot \sqrt{1-\xi^2}}$

Question 6 Ecrire le code python permettant de déterminer ω_0 à partir des résultats précédents.

Afin de vérifier nos résultats, nous souhaitons calculer le temps de réponse par deux moyens.

Dans un premier temps, nous allons chercher à le calculer à partir de s en cherchant, en partant de la fin, le moment à partir duquel la valeur de $s(t)$ sort de la bande de $\pm 5\%$. On donne pour cela l'extrait de code suivant :

```
i=-1
while .....:
    i+=-1
t5=t[i+1]
```

Question 7 Compléter le code afin d'obtenir le résultat souhaité.

On rappelle que $t_{R,5\%} = \frac{1}{\xi \cdot \omega_0} \cdot \ln(20)$, on donne donc la fonction suivante (\log correspond bien au logarithme népérien en python) :

```
def tr5(xi,w0):
    return np.log(20)/(xi*w0)
```

Question 8 Ecrire un code python qui affiche "OK" si la différence entre les deux valeurs calculées pour le temps de réponse est inférieure à 0,01s et "KO" si ce n'est pas le cas. La fonction valeur absolue s'écrit `abs()` en python.

Question 9 Expliquer l'intérêt du code suivant.

```
while t5>0.025:
    xi+=10**(-3)
    s=K*(1-np.exp(-w0*xi*t)/np.sqrt(1-xi**2)*np.cos(w0*np.sqrt(1-xi**2)*t-np.arctan(xi/np.sqrt(1-xi**2))))
    i=-1
    while s[i]<1.05*s[-1] and s[i]>0.95*s[-1]:
        i+=-1
    t5=t[i+1]
```

3 Valeur approchée de ξ

Il se trouve que le cahier des charges du système demande un temps de réponse à 5% inférieur à 0,025s. Pour cela, il faut que ξ soit environ égal à 0,65. Ce nombre doit alors stocké.

Question 10 Ecrire sous la forme d'un mot de 32 bits respectant la norme IEEE 754 (signe, exposant, mantisse) le float 0, 65.

Question 11 Montrer que $00110011001100110011 = \frac{2^{20}-1}{5}$.

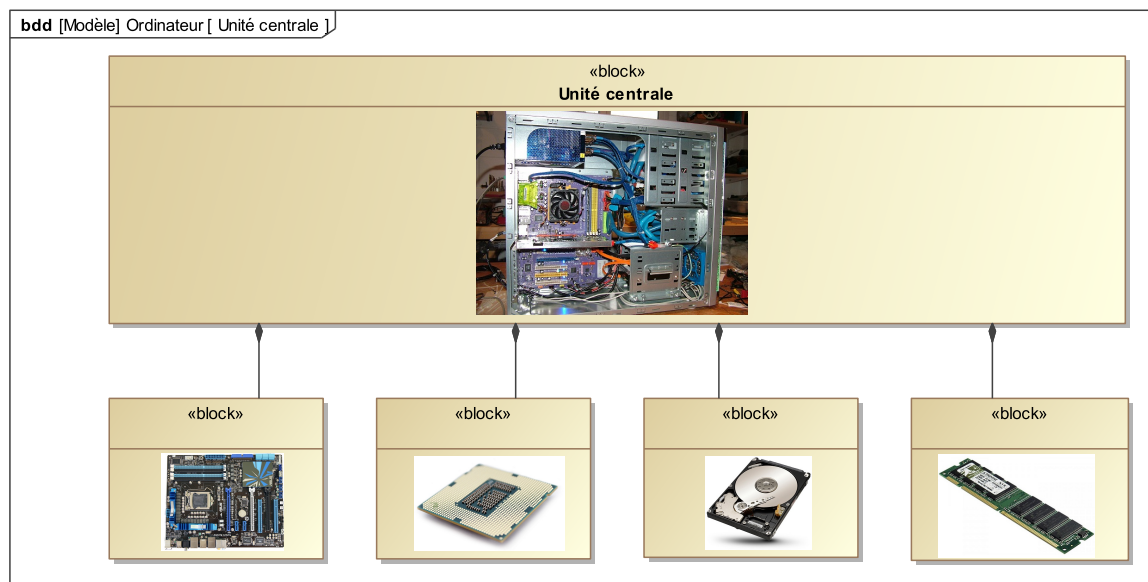
On donne : $2^{-3} = 0.125$, $2^{-1} = 0.5$, $2^{-23} \approx 1, 2 \cdot 10^{-7}$.

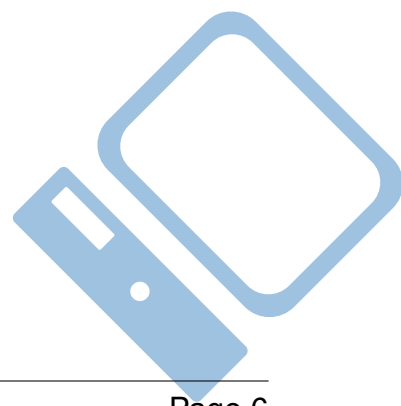
Question 12 Déterminer l'erreur due au stockage de 0,65 à l'aide de la norme IEE74.

4 Document réponse

Nom :

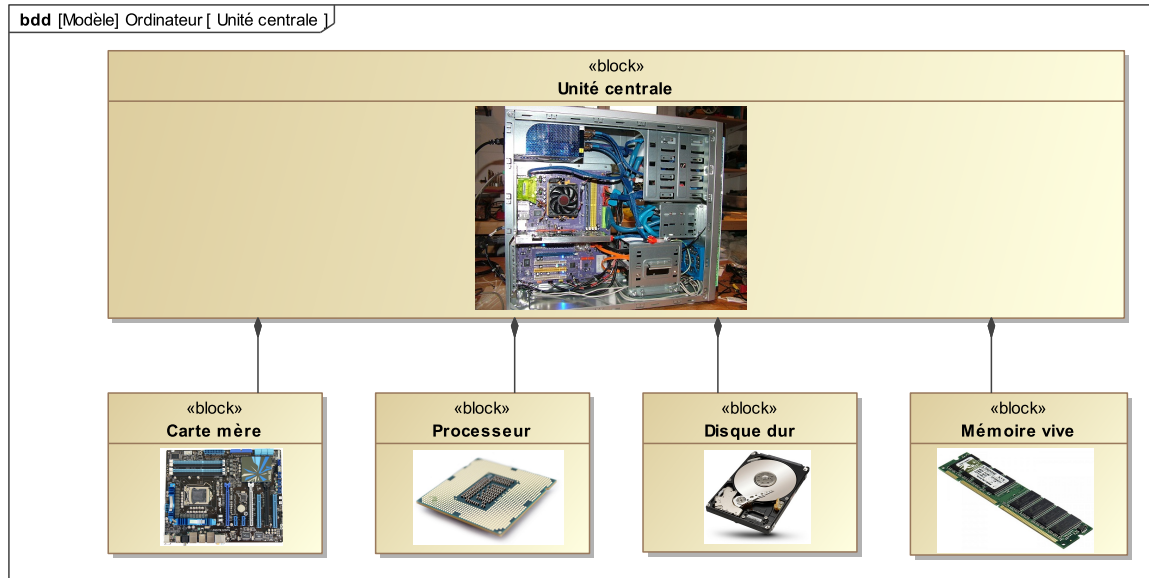
Prénom :





1 Correction

Question 1 :



Question 2 : $K=s[-1]$.

Question 3 : La valeur `val` est la plus grande valeur de la liste `s`.

Question 4 : Cette fonction cherche le temps qui correspond au double du temps correspondant au point le plus haut de la courbe. C'est bien T_p .

Question 5 :

```
xi=0.7
while D(xi)<100*(recherche(s)-s[-1])/s[-1]:
    xi+=-0.01
```

Question 6 :

```
w0=2*np.pi/(Tp*np.sqrt(1-xi**2))
```

Question 7 :

```
i=-1
while s[i]<1.05*s[-1] and s[i]>0.95*s[-1]:
    i+=-1
t5=t[i+1]
```

Correction

Question 8 :

```

if abs(t5-tr5(xi,w0))<0.01:
    print("OK")
else:
    print("KO")

```

Question 9 : Le code suivant cherche la valeur de ξ telle que le temps de réponse soit inférieur à 0,025s.

Question 10 : Le nombre à traduire est $0,65_2$.

$$\begin{array}{rclclcl}
 0,65 & \times & 2 & = & 1,3 & = & 1 & + & 0,3 \\
 0,3 & \times & 2 & = & 0,6 & = & 0 & + & 0,6 \\
 0,6 & \times & 2 & = & 1,2 & = & 1 & + & 0,2 \\
 0,2 & \times & 2 & = & 0,4 & = & 0 & + & 0,4 \\
 0,4 & \times & 2 & = & 0,8 & = & 0 & + & 0,8 \\
 0,8 & \times & 2 & = & 1,6 & = & 1 & + & 0,6 \\
 0,6 & \times & 2 & = & 1,2 & = & 1 & + & 0,2
 \end{array}$$

...
 On remarque une récurrence dans l'écriture du $0,65_{10}$ en binaire : $0,65_{10} = 0,10100110011..._2$
 Le nombre stocké est alors : $1, \underbrace{01001100110011001100110_2}_{23bits} * 2^{-1}$

- Signe = 0,
- Mantisse : $\underbrace{01001100110011001100110_2}_{23bits}$,
- Exposant : $127 - 1 = 126_{10} = 01111110_2$

Question 11 : $a = 00110011001100110011 = 11111111111111111111 - 11001100110011001100 = (2^{20} - 1) - 4 * a$, donc $a = \frac{2^{20}-1}{5}$.

Question 12 : Le nombre stocké est donc : $10100110011001100110011.2^{-23} = (2^{20} + 2^{22} + \frac{2^{20}-1}{5}).2^{-23} = 2^{-3} + 2^{-1} + \frac{2^{-3}}{5} - \frac{2^{-23}}{5} = 0,125 + 0,5 + 0,025 - \frac{1.2.10^{-7}}{5} = 0,65 - 24.10^{-9}$.
 L'erreur est donc de $-\frac{1.2.10^{-7}}{5} = -24.10^{-9}$

