teaching sciences for innovation

Proposition de corrigé

Concours : Banque PT

Année : 2015

Filière : PT

Épreuve : Informatique et Modélisation

Ceci est une proposition de corrigé des concours de CPGE, réalisée bénévolement par des enseignants de Sciences Industrielles de l'Ingénieur et d'Informatique, membres de l'UPSTI (Union des Professeurs de Sciences et Techniques Industrielles), et publiée sur le site de l'association :

https://www.upsti.fr/espace-etudiants/annales-de-concours

A l'attention des étudiants

Ce document vous apportera des éléments de corrections pour le sujet traité, mais n'est ni un corrigé officiel du concours, ni un corrigé détaillé ou exhaustif de l'épreuve en question.

L'UPSTI ne répondra pas directement aux questions que peuvent soulever ces corrigés : nous vous invitons à vous rapprocher de vos enseignants si vous souhaitez des compléments d'information, et à vous adresser à eux pour nous faire remonter vos éventuelles remarques.

Licence et Copyright

Informez-vous !

Retrouvez plus d'information sur les Sciences de l'Ingénieur, l'orientation, les Grandes Ecoles ainsi que sur les Olympiades de Sciences de l'Ingénieur et sur les Sciences de l'Ingénieur au Féminin sur notre site : www.upsti.fr

L'équipe UPSTI

Epreuve d'informatique et modélisation

version du : 2 juillet 2015

Resumé :
Ce corrigé correspond à la partie informatique du sujet Informatique et Modélisation 2015 de la Banque PT. Il est important de noter que les solutions pouvant être multiples pour une problématique donnée, ce corrigé n'envisage pas toutes les possibilités.

UPSTI

Question 1 *Ecrire une fonction $init\_T(Tmax, dt)$ prenant pour arguments la duree $Tmax$ de la simulation et le pas de temps dt et retournant la liste T. Si $Tmax$ n'est pas multiple de dt, on arrondira la durée de la simulation au multiple entier de dt immédiatement supérieur.*

Corrigé

```python
def init_T(Tmax,dt):
    T=[]
    if Tmax%dt==0:
        N=int(Tmax//dt)+1
    else:
        N=int(Tmax//dt)+2
    for i in range(N):
        T.append(i*dt)
    return T
```

Question 2 *Ecrire une fonction $init\_E(T, f)$ prenant pour arguments la liste des instants de la simulation et la fréquence f de la porteuse et retournant la liste des valeurs $e(t_i)$ de la tension e aux instants $T[i]$, d'après la définition de $e(t)$ et pour le message (0,1,0).*

Corrigé

```python
def init_E(T,f):
    E=[]
    message=[0,1,0]
    Emin=1.3
    Emax=1.5
    for i in range(len(T)):
        j=i//16
        val=message[j]
        if val==0:
            e0=Emin
        else:
            e0=Emax
        E.append(e0*sin(2*pi*f*T[i]))
    return E
```

Question 3 *Donner une approximation de $\frac{ds}{dt}(t_i)$ en fonction de $s(t_i)$, $s(t_{i+1})$ et $\Delta t = t_{i+1} - t_i$ en utilisant la formule d'Euler explicite. En déduire, dans le cas où la diode est bloquée à l'instant t_i, la relation de récurrence donnant $s(t_{i+1})$ en fonction de $s(t_i)$, τ, et Δt.*

Corrigé

Formule d'Euler explicite : $\frac{ds}{dt}(t_i) \approx \frac{s(t_{i+1}) - s(t_i)}{\Delta t}$ avec Δt petit.

Lorsque la diode est bloquée à l'instant t_i, alors : $\frac{ds}{dt}(t_i) + \frac{1}{\tau} \cdot s(t_i) = 0$

Corrigé

On a alors : $\frac{s(t_{i+1})-s(t_i)}{\Delta t}+\frac{1}{\tau}\cdot s(t_i)=0$
Donc on obtient la relation de récurrence suivante donnant $s(t_{i+1})$:
$s(t_{i+1})=(1-\frac{\Delta t}{\tau})\cdot s(t_i)$

Question 4 *Pourquoi ne peut-on pas utiliser la formule d'Euler explicite pour effectuer le test ici ? Donner, en utilisant la démarche proposée, une condition portant sur $s(t_{i+1})$, $s(t_i)$, τ et Δt.*

Corrigé

Ici, la formule d'Euler explicite n'est pas utilisable car elle demanderait de connaître la valeur future de $s(t_{i+2})$ (encore inconnue) pour pouvoir calculer $\frac{ds}{dt}(t_{i+1})$. Ceci est nécessaire afin d'évaluer l'expression $\frac{ds}{dt}(t_{i+1})+\frac{1}{\tau}\cdot s(t_i)$, et d'en déduire si la diode sera toujours passante au pas de temps suivant.

Formule d'Euler implicite : $\frac{ds}{dt}(t_{i+1})\approx\frac{s(t_{i+1})-s(t_i)}{\Delta t}$ avec Δt petit.

D'où :

$\frac{ds}{dt}(t_{i+1})+\frac{1}{\tau}\cdot s(t_{i+1})>0$

$\frac{s(t_{i+1})-s(t_i)}{\Delta t}+\frac{1}{\tau}\cdot s(t_{i+1})>0$

La diode reste donc passante à l'instant t_{i+1} si $s(t_{i+1})>(1-\frac{\Delta t}{\tau})\cdot s(t_i)$

Question 5 *Ecrire alors une fonction $solve(T,E,tau)$ prenant pour arguments la liste T des instants de la simulation, la liste E des tensions d'entrée et la constante de temps tau et retournant la liste S des tensions de sortie. Les conditions initiales seront prises nulles et, si nécessaire, l'état initial de la diode sera supposé passant.*

Corrigé

python

```
def solve(T,E,tau):
    etat_diode="passante" #Diode supposee etre passante au depart
    S=[0] #Condition initiale nulle s(0)=0
    for i in range(1,len(T)):
        delta_t=T[i]-T[i-1]
        if etat_diode=="passante": #cas diode passante
            S[i]=E[i]
            if S[i]>E[i]:
                etat_diode="bloquee"
        else:                      #cas diode bloquee
            S[i]=(1-delta_t/tau)*S[i-1]
            if S[i]>(1-delta_t/tau)*S[i-1]:
                etat_diode="passante"
    return S
```

Question 6 *Indiquer la valeur du pas de temps (1, 10 ou 100s) correspondant à chacune des trois simulations, en justifiants vos réponses.*

Corrigé

Figure 1 : pas de temps de 100s
Figure 2 : pas de temps de 10s
Figure 3 : pas de temps de 1ns
Plus le pas de temps est petit devant la constante de temps du circuit, plus l'algorithme est capable de suivre les variations de e(t) avec précision (discrétisation plus fine). C'est pour cela que sur la figure 5, le tracé de e(t) ressemble le plus à celui de la figure 1.

Question 7 *Expliquer en quelques phrases les causes des différences obtenues entre les trois résultats s(t). Repérer en particulier les instants auxquels la diode change d'état. Que constatez-vous ?*

Corrigé

Dans le cas de la figure 3, le pas de temps est tellement grossier que l'algorithme est incapable de suivre l'évolution temporelle de e(t) qui se fait à haute fréquence. Par conséquent, c'est comme si la diode ne changeait pas d'état depuis le départ. Il est donc impossible de détecter la moindre enveloppe.

Dans le cas de la figure 4, le pas de temps est plus fin que dans le cas précédent mais pas assez pour suivre fidèlement les variations de e(t). Du coup, c'est comme si la diode changeait d'état mais pas toujours aux bons instants. La précision du résultat n'est donc pas satisfaisante pour représenter correctement le phénomène modélisé.

Enfin, dans le cas de la figure 5, le pas de temps est suffisamment petit pour suivre correctement les variations de e(t) et est donc capable d'effectuer les changements d'état de la diode au bons instants.

Question 8 *Indiquer, pour chacun des trois résultats, s'il est possible d'identifier un tel seuil, et donc si la récupération du message binaire semble réalisable si l'on se base uniquement sur ce résultat. Conclure sur le critère que doit respecter le pas de temps d'une simulation temporelle pour que les résultats de celle-ci aient une chance d'être pertinents.*

Corrigé

Seule la simulation pour le pas de temps de 1ns permet de d'identifier clairement les seuils séparant les deux niveaux haut et bas de la tension s(t).

Autrement dit, le pas de temps à retenir pour qu'une simulation temporelle ait des résultats pertinents doit être très petit devant la constante de temps du système modélisé.

Dans le cas présent, il y a un facteur 1000 entre le pas de temps d'1ns et la constante de temps du système modélisé qui est de 1µs.

Question 9 *Donner les bits de parité associés aux représentations binaires des entiers 5, 16 et 37.*

Corrigé

$(5)_{10} = (0101)_2$. Le bit de parité associé à la représentation binaire de 5 est de 0.
$(16)_{10} = (1000)_2$. Le bit de parité associé à la représentation binaire de 16 est de 1.
$(37)_{10} = (00010101)_2$. Le bit de parité associé à la représentation binaire de 37 est de 1.

Question 10 *Ecrire une fonction parite(bits) prenant pour argument une liste bits constituée d'entiers valant 0 ou 1 et retournant l'entier 0 ou 1 correspondant à son bit de parité.*

Corrigé

python

```
def parite(bits):
    """
    Fonction qui renvoie la valeur du bit de parite d'un un mot binaire donne sous forme
    d'une liste de 0 et de 1 nommee bits
    """
    somme_bits=0
    for i in range(len(bits)): #calcul de la somme des bits
        somme_bits+=bits[i]
    if somme_bits%2==0: #test de la parite de la somme des bits
        return 0
    else:
        return 1
```

Question 11 *Donner un exemple d'erreur n'étant pas détectable par cette technique. Si une erreur a été détectée, est-il possible de la corriger sans retransmettre la donnée ?*

Corrigé

Un exemple d'erreur de transmission non détectable par la technique de découpage en blocs et association d'un bit de parité serait la permutation entre un 0 et un 1 dans le mot binaire transmis.

Exemple :

- mot binaire de 7 bits envoyé= $(1000010)_2 = (66)_{10}$. Bit de parité associé=0.
- mot binaire de 7 bits reçu= $(0100010)_2 = (34)_{10}$. Bit de parité associé=0.

Donc le bit de parité est le même et pourtant le mot binaire reçu est différent de celui qui est reçu.

Si une erreur est détectée par cette technique, on sait simplement qu'il n'y a pas le bon nombre de "1" dans le mot binaire reçu. On ne sait rien quant à la position des "1" manquants ou en trop dans le mot binaire reçu. Il n'est donc pas possible de corriger ce type d'erreur sans retransmettre la donnée.

Question 12 *Ecrire une fonction encode_hamming(donnee) prenant pour argument une liste donnee de quatre bits (représentés par des entiers valant 0 ou 1) et retournant une liste de bits contenant le message encodé. On pourra appeler la fonction parite(bits) précédemment définie.*

Corrigé

python

```
def encode_hamming(donnee):
    """
    Fonction qui renvoie un message encode par le code de hamming de 7 bits
    a partir d'un argument, nomme donnee, qui est un message de 4 bits sous forme d'une liste
    de 1 et de 0
    """
    d1,d2,d3,d4=[donnee[i] for i in range(len(donnee))]
    p1=parite([d1,d2,d4]) #calcul de parite du triplet 1
    p2=parite([d1,d3,d4]) #calcul de parite du triplet 2
    p3=parite([d2,d3,d4]) #calcul de parite du triplet 3
    return [p1,p2,d1,p3,d2,d3,d4]
```

Question 13 *Ecrire une fonction $decode\_hamming(message)$ prenant pour argument une liste de 7 bits et retournant quatre bits contenant la donnée décodée. En cas d'erreur, on affichera à l'écran un avertissement indiquant la position du bit affecté et on effectuera la correction. On supposera dans cette question que s'il y a une erreur, elle est unique.*

Corrigé

```python
def decode_hamming(message):
    """
    Fonction qui renvoie le message decode, et corrige en cas d'erreur detectee par la
    technique de hamming
    """
    m1,m2,m3,m4,m5,m6,m7=[message[i] for i in range(len(message))]
    c1=parite([m4,m5,m6,m7])
    c2=parite([m2,m3,m6,m7])
    c3=parite([m1,m3,m5,m7])
    message_decode=[message[2],message[4],message[5],message[6]]
    if c1==0 and c2==0 and c3==0:
        return message_decode #message decode retourne s'il n'y pas d'erreur
    else:
        position_erreur=c1*4+c2*2+c3 #calcul de la position de l'erreur dans le message decode
        print("une erreur de transmission a ete detectee pour le bit en position ", position_erreur)
        if message_decode[position_erreur-1]==1: #permutation de la valeur du bit errone
            message_decode[position_erreur-1]=0
        else:
            message_decode[position_erreur-1]=1
        return message_decode #message decode retourne corrige en cas d'erreur
```

Question 14 *Déterminer le codage de Hamming de la donnée 1011, puis la donnée décodée par l'algorithme dans l'hypothèse où les deux premiers bit du message codé ont été incorrectement transmis. Quel a été l'effet de la "correction" sur a donnée dans ce cas ?*

Corrigé

Codage de Hamming de la donnée 1011 : [0,1,1,0,0,1,1]
Code erroné envoyé (2 premiers bits erronés) : [1,0,1,0,0,1,1]
Valeur des bits de contrôle du message reçu : (0,1,1)
Position du bit erroné détecté par l'algorithme (qui est basé sur la détection d'une seule erreur dans un message transmis) : bit erroné en position 3
Donnée décodée et corrigée par l'algorithme : 1001
Evidemment, la correction d'erreur mise en place dans l'hypothèse d'une seule erreur sur l'un des bits du message transmis ne fonctionne pas correctement et renvoie un message corrige qui est erroné.

Question 15 *Sans coder, proposer un moyen simple de différencier une double erreur d'une erreur unique au moyen d'un bit de parité supplémentaire et expliquer comment cela permet d'éviter le problème mis en évidence à la question précédente. On s'appuiera sur les techniques introduites dans cette partie. On ne demande pas d'essayer de corriger la double erreur.*

Corrigé

On pourrait rajouter un contrôle de la parité des bits de parité. Cela permettrait de connaître le nombre d'erreurs de transmission et donc d'éviter de fausses correction comme dans l'exemple.

Question 16 *Donner les types et les valeurs des variables id_titre, $zones$ et $date\_fin$ à l'issue de ces instructions pour le fichier 0001.txt donné ci-dessus.*

Corrigé

$id\_titre = 49987654$: c'est un entier.
$zones = [1,3]$: c'est une liste (d'entiers).
$date\_fin = [2015,08,31]$: c'est une liste (d'entiers). Remarque : erreur du sujet dans le script python. Il s'agit bien de la variable $date\_fin$ et non ch_date_fin.

Question 17 *Ecrire le bloc d'instructions à exécuter à la suite des opérations précédentes pour construire le tableau passages à partir des lignes 3, 4 et 5 contenues dans la liste lignes.*

Corrigé

```python
passages=[]
for i in range(2,5):
    data = lignes[i]
    data =data.rstrip('\n').split(',')
    liste =[]
    [liste.append(data[0].split('-')[j]) for j in range(3)]
    [liste.append(data[1].split(':')[k]) for k in range(3)]
    liste.append(data[2])
    passages.append(liste)
```

Question 18 *Ecrire une fonction estAvant(date1, date2) prenant pour arguments deux dates au format [année, mois, jour] (donc sous forme de listes de trois entiers chacune) et retournant True si date1 est antérieure ou égale date2, et False sinon.*

Corrigé

```python
def estAvant(date1,date2):
    if date1[2]<=date[2]:
        if date1[1]<=date2[1]:
            if date1[0]<=date2[0]:
                return True
    else:
        return False
```

Question 19 *Ecrire une fonction nbSecondesEntre(heure1, heure2) prenant pour arguments deux horaires au format [heures, minutes, secondes] (donc sous forme de listes de trois entiers chacune) et retournant le nombre de secondes séparant les deux instants. Le résultat devra être positif si heure1 est postérieure à heure2.*

Corrigé

```python
def nbSecondesEntre(heure1, heure2):
    return (heure1[0]-heure2[0])*3600 +(heure1[1]-heure2[1])*60 +(heure1[2]-heure2[2])
```

Question 20 *Ecrire une fonction testPassage() dont les arguments sont à préciser, retournant la valeur True si le passage est autorisé et la valeur False sinon et, dans ce dernier cas, affichant à l'écran le message correspondant aux règles de priorité ci-dessus. On pourra utiliser toutes les variables définies dans cette partie et appeler les fonctions définies dans les deux questions précédentes.*

Corrigé

```python
def testPassage(id_titre, zones, date_fin,passages) :
    if id_titre in Liste_noire : #test liste noire
        print ("Titre refuse")
        return False
    if Zone not in zones : #test zone
        print("Non valide dans cette zone")
        return False
    date =Maintenant[:,3]
    if estAvant( date_fin, date) : #test date
        print ("Titre expire")
        return False
    i=2
    while (i>=0) and (passage[i][6]==Id_point):
        heureactuelle=Maintenant[3, :]
        heuredernierpassage=passages[i][3, :]
        if nbSecondesEntre(heureactuelle, heuredernierpassage)<450:
            print ('Titre deja valide')
            return False
        i=i-1
    return True
```

Question 21 *Donner la requête SQL permettant de récupérer les dates et les heures de tous les passages ayant eu lieu sur la ligne numérotée 1 entre le 1er juillet et le 31 août 2014 (inclus).*

Corrigé

```mysql
SELECT passages.date,passages.heure FROM passages JOIN points ON passages.id_point =
points.id WHERE points.ligne=1 and (passages.date >="01/07/2014") AND (passages.date <=
"31/08/2014" );
```

Autre solution possible (plus courte à écrire) :

Corrigé

```mysql
SELECT p.date,p.heure FROM passages as p JOIN points as pt ON p.id_point =
pt.id WHERE pt.ligne=1 and (P.date >="01/07/2014") AND (p.date <="31/08/2014" );
```

Question 22 *Donner la requête SQL permettant de compter le nombre de passages dézonés, c'est-à-dire ayant eu lieu hors de l'intervalle de validité du titre utilisé, effectués le 31 décembre 2014.*

Corrigé

Pour la deuxième il y a une erreur dans le sujet : il n'y a pas de clé primaire (id) dans la table. passages.

```sql
SELECT COUNT passages.id FROM passages, points, titres WHERE passages.id_point =
points.id AND passages.id_titre =titres.id AND passages.date="2014-12-31" AND (points.zone <
titres.zone_max OR points.zone >titres.zone_max );
```