

Examen d'introduction à la microinformatique 1er Bachelor Solvay

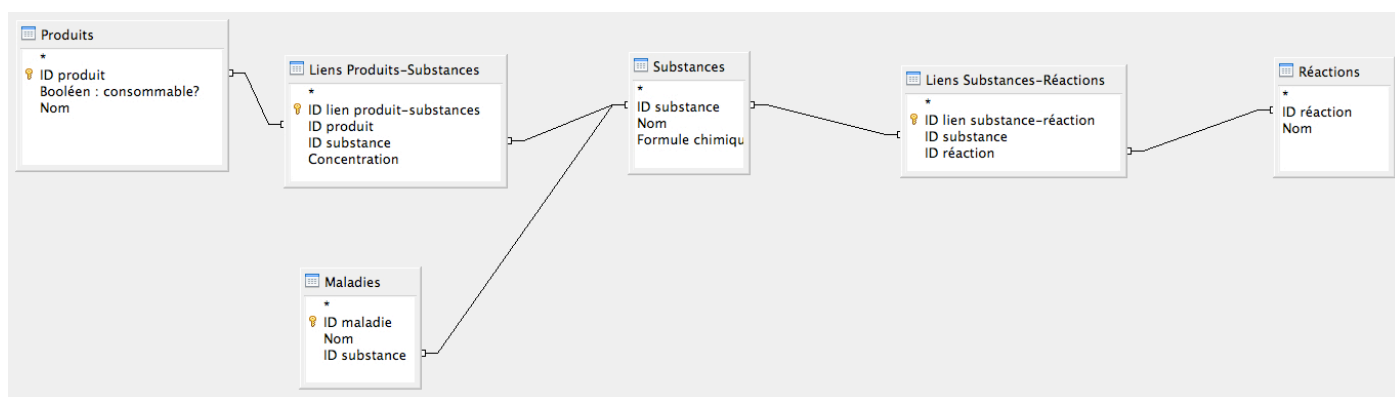
Lundi 14/1/2008

Nom :

Prénom :

Question 1 – Base de données relationnelles (/5) :

La commission européenne a décidé récemment de répertorier dans une base de données relationnelle les substances chimiques susceptibles de poser, à la consommation, des problèmes de santé. Elle désire facilement avoir accès aux différentes substances chimiques (répertorié par leur nom, formule chimique, ...) ainsi que les produits, séparés en consommables et non-consommables, dans lesquels apparaissent ces substances chimiques. Pour chacun de ces produits, on indiquera à quelle concentration (en pourcentage) on trouve les substances chimiques incriminées. En plus, la commission souhaite retrouver aisément les maladies que l'on pense ces substances susceptibles de provoquer (on supposera qu'une maladie n'est causée que par une seule substance, bien qu'une même substance puisse en causer plusieurs), ainsi que les réactions chimiques les plus importantes dans lesquels ces substances sont impliquées (une réaction chimique nécessite au moins deux substances chimiques). Réalisez la base de données relationnelle permettant d'enregistrer toutes ces informations, en y intégrant tables, relations et attributs indispensables.



Question 2 – Python (/5)

Expliquez le rôle du petit code Python qui suit (indiquez à côté de certaines lignes le rôle qu'elles jouent) et ce qui sera affiché à l'écran à l'issue de son exécution. L'instruction « `l.append(a)` » ajoute « `a` » à la liste « `l` ».

```
*****
```

```
>>> def func(x):
```

```
    a = 0
```

```
    for i in x:
```

```
        a=a+1
```

```
    return a
```

```
>>> def func2(x):
```

```
    s = 0
```

```
    it = 0
```

```
    a=func(x)
```

```
    r=0
```

```
    while it<a-1:
```

```
        s=s+x[it]
```

```
        it=it+1
```

```
    if (s%2 == 0) and (x[a-1] == 1):
```

```
        r=1
```

```
    elif (s%2 == 1) and (x[a-1] == 0):
```

```
        r=1
```

```
    else:
```

```
        r=0
```

```
    return r
```

```
>>> list=[]
```

```
>>> list2=[]
```

```
>>> list.append((1,0))
```

```
>>> list.append((1,1,0))
```

```
>>> list.append((1,0,0,1,1))
```

```
>>> list.append((1,0,0,0,1))
```

```
>>> list.append((0,0,1))
```

```
>>> list.append((0,0,1,0))
```

```
>>> for x in list:
```

```
    list2.append(func2(x))
```

```
>>>list2
```

```
*****
```

Question 3 – Théorie (/4)

Des biologistes demandent à un informaticien de leur permettre de stocker informatiquement des protéines qui sont des séquences d'acides aminés (chaque acide aminé pouvant être choisi parmi 20 candidats possibles), et cela de la manière la plus économe qui soit. Ces protéines sont représentées comme des séquences de lettres (prise en effet dans un alphabet de 20 lettres) dans lesquelles de nombreuses répétitions apparaissent, comme par exemple :

AAABBBBBBUUEEEEGGHHHHHHHKJJJJLLLLLLLLLLLLBB

Les biologistes savent d'expérience que le nombre maximum de ces répétitions pour une même lettre est de 50. Quelle traduction en binaire de ces protéines notre ingénieux informaticien pourrait leur proposer.

il y a	- A : 3	
	- B : 5 + 2	
	- U : 2	
	- E : 4	
	- G : 2	ça fait 9 lettres différentes
	- H : 7	42 lettres
	- K : 2	
	- J : 6	
	- L : 10	

MAIS ATTENTION 20 acides aminés possibles (20 lettres possibles)
 $2^4\text{bits} = 16$ possibilités (trop petit) et $2^5\text{bits} = 32$ donc 5 bits pour les lettres

On a que les lettres peuvent se répéter max 50 fois
 $2^5\text{bits} = 32$ (trop petit) donc 6 bits ($2^6 = 64$) pour le nombre de fois que chaque lettre apparaît

Question 4 – Théorie (/3)

Soit un système de mémoire cache organisé de telle sorte que les blocs circulant entre la mémoire centrale et la mémoire cache soient de 32 byte, et une mémoire cache de dimension classique de 4kByte (4096 byte). Dans un ordinateur, dont les adresses mémoires sont de longueur 4 byte, expliquez la manière dont ces adresses seront découpées de manière à faire fonctionner le système de cache.

mémoire cache = 4kByte = 4096 byte

-> 2^{12} bytes

blocs = 32 bytes

-> 2^5 bytes => 5 bits d'OCTET

4096 : 32 bytes = 128 lignes

-> 128 lignes de 32 bytes

-> 2^7 lignes => 7 bits de LIGNE

(remarque : $2^7 \cdot 2^5 = 2^{12}$)

longueur adresse mémoire = 4 byte = $4 \times 8 \text{ bits} = 32 \text{ bits}$

$32 - 7 - 5 = 20$

-> 7 bits (LIGNE)

-> 5 bits (OCTET)

-> 20 bits restants... (TAG)

Question 5 - Théorie (/3)

La couche 3 du protocole Internet hiérarchisé OSI, à la réception d'un paquet, va :

- 1) soit remplacer l'adresse IP du paquet actuel
- 2) soit l'envoyer sur la couche 2 en redécoupant le paquet
- 3) et dans ce cas y adjoindre une adresse MAC.

Expliquez brièvement pourquoi ces trois possibilités.

- 1) Lorsque le paquet n'est pas destiné à un ordinateur du réseau local et le serveur le renvoie sur Internet
- 2) Si le paquet est destiné au réseau local, c'est le protocole Ethernet qui prend le relais, il faut donc redécouper le paquet en trames et assigner à chaque trame une adresse MAC (le système d'adressage sur réseau local).

Examen d'introduction à la microinformatique 1er Bachelor Solvay

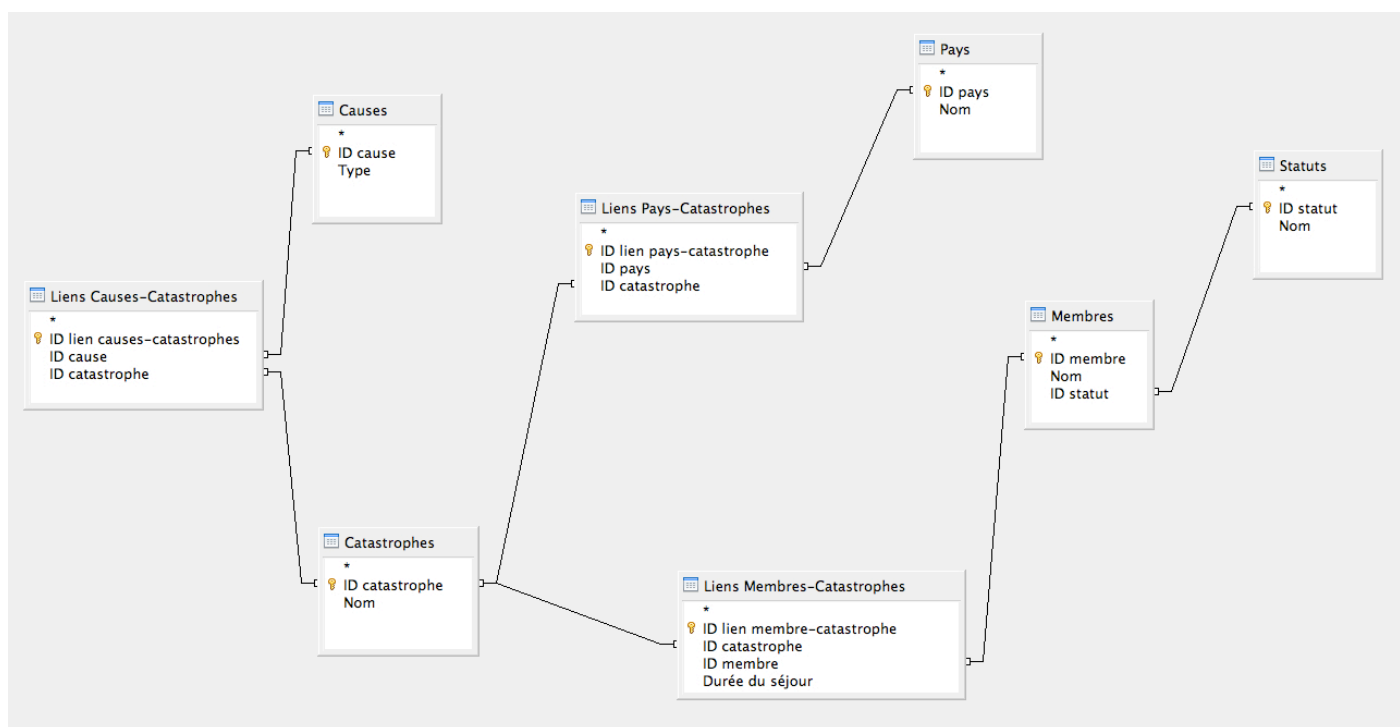
Samedi 17/05/2008

Nom :

Prénom :

Question 1 – Base de données relationnelles (/5) :

L'ONG « MSF » aimerait constituer une base de données dans laquelle elle répertorierait les catastrophes humanitaires survenues et pour lesquelles l'ONG a envoyé du personnel soignant sur place. Cette base devrait contenir les différentes catastrophes survenues associées aux causes qui les ont provoquées (tremblement de terre, tsunami, guerre civile, famine). Plusieurs causes pourraient être à l'origine d'une même catastrophe humanitaire. La base de données devrait permettre de retrouver le ou les pays dans lesquels chaque catastrophe a eu lieu. Elle devrait permettre aussi de retrouver facilement les membres de MSF qui se sont rendus sur les lieux et la durée de leur séjour (précisez l'emplacement dans la table où vous indiqueriez l'information sur cette durée). Chaque membre sera également caractérisé par son statut unique tel que : aide soignant, gestionnaire, ingénieur, chirurgien ou comptable. Réalisez la base de données relationnelle permettant d'enregistrer toutes ces informations, en y intégrant tables, relations et attributs indispensables.



Question 2 – Python (/5)

Expliquez brièvement le rôle du petit code Python qui suit et indiquez très précisément ce qui sera affiché à l'écran en exécutant les quatre dernières lignes du code. L'instruction « list.append(a) » ajoute « a » à la liste « list ».

```
*****
```

```
>>> def find(li1, li2):
    lx = []
    lx2 = []
    y = z1 = z2 = xx = yy = zz2 = 0
    for a1 in li2:
        y = y+1
    z1 = 1
    for a2 in li1:
        z2 = 1
        for a3 in li2:
            if a2 == a3 and z2 == zz2 + 1:
                lx.append(z1)
                xx = xx + 1
                zz2 = z2
                z2 = z2 + 1
                if (xx == y):
                    lx2.append(lx[0])
                    lx = []
                    xx = 0
                    zz2 = 0
            z2=z2+1
        z1 = z1 + 1
    return lx2
```

```
>>> find('gta','gt')
>>> find('aagtaaaa','aaa')
>>> find('gtcgtcgtc','gtc')
>>> find('gtaaacgtcgaataac','gtc')
```

```
*****
```

Question 3 – Théorie (/3)

Expliquez brièvement en quoi consiste le mécanisme de cryptage dit « asymétrique », et en quoi s'est-il avéré un important progrès sur son prédécesseur « symétrique » ?

Le cryptage asymétrique nécessite 2 clés dite privée et publique (on crypte avec publique et on décrypte avec privée) et celle la publique circule, ce qui permet un protocole beaucoup plus sécurisé car même si la publique est captée, le secret reste bien gardé en l'absence de la privée.

Question 4 – Théorie (/3)

Imaginez un système d'adressage relatif disposant pour ce faire de 8 registres. De quelle taille doivent être ces registres afin de pouvoir diviser par deux la longueur des adresses mémoires RAM qui seraient de 16 bits en adressage absolu. Justifiez votre réponse.

16 bits en adressage absolu
-> 8 bits en adressage relatif

il y a 8 registres

$2^3 = 8 \Rightarrow 3$ bits pour l'adressage des registres

$\Rightarrow 5$ bits pour l'adressage dans la RAM (8bits - 3bits = 5bits)

il est donc 11 bits registre et 5 bits RAM pour retrouver les 16 bits d'adressage absolu. (16bits - 5bits de RAM = 11bits de registres)

Question 5 - Théorie (/4)

Justifiez pourquoi le premier octet des adresses IP de classe A code pour un nombre compris entre 0 et 127, le premier octet des adresses IP de classe B un nombre compris entre 128 et 191 et finalement le premier octet des adresses IP de classe C un nombre compris entre 192 et 223.

Classe A : 00000000 -> 01111111 de 0->127

Classe B : 10000000 -> 10111111 de 128->191 (dont l'ULB)

Classe C : 11000000 -> 11011111 de 192 -> 223

Examen d'introduction à la microinformatique 1er Bachelor Solvay

Jeudi 05/1/2009

Nom :

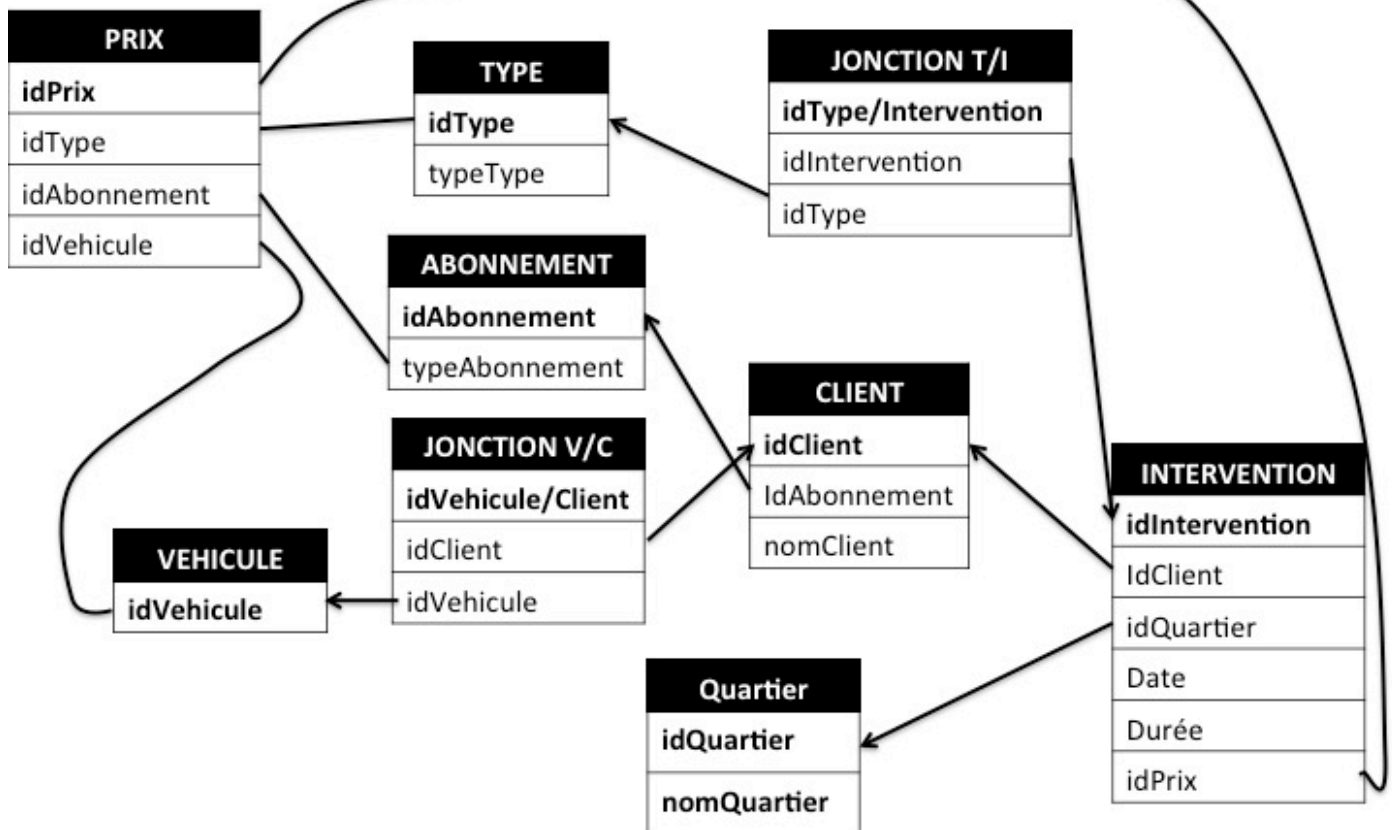
Prénom :

Question 1 – Base de données relationnelles (/5) :

Le service de dépannage « Touring Secours » voudrait répertorier dans une base de données relationnelles les différents clients qui y sont affiliés ainsi que les différentes interventions qu'il a effectuées à Bruxelles durant l'année 2008 pour ses clients. Pour chacune des interventions, il est important de pouvoir retrouver la date, la durée et le prix. Le quartier dans lequel cette intervention a eu lieu sera également repris à part. Le prix dépend d'abord du ou des types d'intervention pratiquée, par exemple : « aide au démarrage », « réparation crevaison », « vidange », « recharge batterie » (une seule intervention pouvant être de plusieurs types), mais également du type d'abonnement que le client a contracté chez Touring Secours (prioritaire, secondaire, etc.. étant bien entendu qu'un client ne contracte qu'un seul type d'abonnement) ainsi que du véhicule concerné. Cet abonnement autorise toutefois le client à être propriétaire de plusieurs véhicules et un même véhicule à se trouver associé à plusieurs clients. On considérera qu'une intervention ne concerne qu'un client et qu'un véhicule. Réalisez le schéma de la base de données relationnelles qui permettra de facilement retrouver ces différentes informations.

n → 1

Janvier 2009



Question 2 – Python (/5)

Expliquez dans les grandes lignes le rôle du petit code Python qui suit et retrouvez ce qui sera affiché à l'écran à l'exécution des trois dernières instructions du code.

```
*****
```

```
def func(l1,l2):  
    a=0  
    for i in l1:  
        for j in l2:  
            if i==j:  
                a=a+1  
    return a
```

```
>>> def func2(l3):  
    a=0  
    for i in l3:  
        a=a+1  
    b=0  
    res=l3[0]  
    a=a-1  
    i=0  
    while i<a:  
        c = func(l3[i],l3[a])  
        if (c > b):  
            b=c  
            res=l3[i]  
        i=i+1  
    return res
```

```
>>> func2(["AGV073","AGH987","HIJ073","HGV073"])  
>>> func2(["AVH900","GVH988","GVH965","GGG988","GRH966"])  
>>> func2(["DDA011","VDD012","CDA010"])
```

```
*****
```

Question 3 – Théorie (/3)

Une instruction Python « $c = a + b$ » pourrait se traduire dans une instruction élémentaire de type CISC sous la forme « add c a b ». Remplacez cette seule instruction par une séquence de 3 instructions élémentaires de type RISC cette fois (beaucoup plus petites car n'acceptant qu'une et une seule opérante) et qui agirait à partir du registre « accumulateur ». Décrivez ce que ferait chacune de ces instructions.

Load a : copie dans l'accumulateur le contenu de la mémoire dont l'adresse est donnée dans l'opérante.

Add b : Ajoute à l'accumulateur le contenu de la mémoire dont l'adresse est donnée dans l'opérante .

Store c : Copie le contenu de l'accumulateur dans la mémoire à l'adresse donnée dans l'opérante

Question 4 – Théorie (/3)

Si l'on estime le temps d'accès à la mémoire centrale à environ 10 ns (nanoseconde) et celui d'accès au disque dur à environ 1 ms (milliseconde), déterminez le taux acceptable de ratés d'accès à la mémoire centrale afin que la performance soit dégradée au plus d'un facteur 2 par rapport à la performance que l'on obtiendrait sans aucun raté.

mémoire centrale = 10 ns = 10^{-8} s
disque dur = 1 ms = 10^{-3} s

raté en mémoire centrale 0,001%

si on sait que le nombre de raté est noté "a", alors la performance s'il y a "a" ratés = $(1-a) \times 10^{-8} + a \times 10^{-3}$. C'est à dire ceux qui ont pas ratés + ceux qui ont ratés.

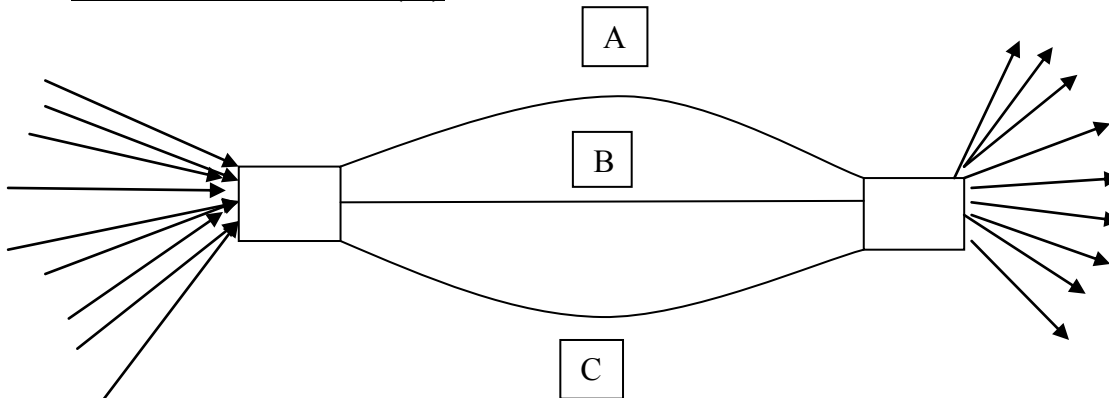
Dans l'énoncé, on nous dit que ça peut être max 2x la performance que s'il y a que des réussis et des réussis ont la performance de 10^{-8}

$$\Rightarrow (1-a) \times 10^{-8} + a \times 10^{-3} = 2 \times 10^{-8}$$

$$\Rightarrow a = 10^{-8} / (10^{-3} - 10^{-8})$$

$$\Leftrightarrow a = 10^{-5}$$

Question 5 - Théorie (/4)



Les neuf internautes expéditeurs à gauche désirent faire parvenir à leurs neuf destinataires à droite un fichier de 10 KBytes chacun. Pour ce faire, l'infrastructure connecte les deux « commutateurs » (du côté des expéditeurs et des destinataires) par 3 lignes A, B et C, permettant chacune un débit de 1 KByte/ms. Le temps de transfert entre les commutateurs et les internautes sera négligé. Calculez les durées moyennes pour que chaque destinataire à droite commence à recevoir les fichiers qui leur sont destinés, d'abord dans le cas d'une commutation par circuit ensuite dans le cas d'une commutation de paquets en mode connecté. Dans ce dernier cas, les fichiers seront découpés en paquets de 1KByte.

Circuits:

on a 3 lignes donc 3 fichiers sont envoyés simultanément sur chacune des lignes

les 3 premières personnes COMMENCE à recevoir le fichier : 0 ms

les 3 suivantes doivent attendre que le fichier soit totalement envoyé avant de recevoir le leur donc 10 ms

les 3 dernières il faut que les 6 autres fichiers soient envoyés donc 20 ms

$$(3 \cdot 0 + 3 \cdot 10 + 3 \cdot 20) / 9 = 10 \text{ ms}$$

Paquet en mode connecté:

les paquets peuvent être mélangés et ont tous 1KByte de taille

les 3 premières personnes COMMENCE à recevoir le fichier : 0 ms

les 3 suivants doivent donc attendre 1 ms

et les dernières : 2 ms

$$(3 \cdot 0 + 3 \cdot 1 + 3 \cdot 2) / 9 = 1 \text{ ms}$$

Examen d'introduction à la microinformatique 1er Bachelor Solvay

Mardi 12/5/09

Nom :

Prénom :

Question 1 – Base de données relationnelles (/5) :

L'organisation mondiale de la santé souhaite constituer une base de données relationnelles dans laquelle elle répertorierait les différentes épidémies survenues ces dernières années, la nature de la maladie responsable de chacune des épidémies et les différents pays touchés par chacune d'entre elle également (une seule maladie sera responsable d'une épidémie). Elle voudrait pouvoir encoder le nombre de victimes recensées dans chaque pays pour une épidémie donnée et également les traitements et vaccins qui ont été appliqués dans chaque pays et pour chacune des épidémies. Alors que la vaccin est appliqué identiquement pour chaque pays et dépend uniquement de l'épidémie en question, les traitements, quant à eux, peuvent être nombreux pour une épidémie donnée et différents pour chaque pays, car dépendant du niveau de développement du pays en question. Réalisez le schéma de la base de données relationnelles qui permettra de facilement encoder et retrouver ces différentes informations.

n → 1

Juin2009

MALADIE
idMaladie
nomMaladie

EPIDEMIE
idEpidemie
idMaladie
nomEpidemie

JONCTION E/P
idEpidemie/Pays
idPays
idEpidemie
Nb de victimes

PAYS
idPays
IdVaccin
IdDev
IdTraitement

DEV
idDev
niveauDev

VACCIN
idVaccin
idEpidemie
nomVaccin

TRAITEMENT
idTraitement
idEpidemie
idDev
nomTraitement

Est-ce qu'on utilise la classe DEV
ou est-ce qu'on fait un lien entre E/P et Traitement ?????

Question 2 – Python (/5)

Expliquez dans les grandes lignes ce que fait le petit code Python qui suit et retrouvez ce qui sera affiché à l'écran à l'exécution de la dernière ligne du code. Le test de la fonction « f1 » vérifie simplement qu'il s'agit bien d'un entier. « insert » rajoute un élément dans la liste à la position indiquée et « append » rajoute un élément en fin de liste.

```
>>> def f1(k1):
    s=0
    m=0
    for i in k1:
        if type(i) is int:
            s=s+i
            m=m+1
    return s/m

>>> def f2(k2):
    a2=100
    li=[]
    for i in k2:
        ij = f1(i)
        if ij <=5:
            if ij < a2:
                li.insert(0,i[0])
                a2=ij
            else:
                li.append(i[0])
    return li
```

```
>>> liste=[("Jean",1,2,3),("Marc",8,10),("Eric",3,6,10,5),("Pierre",1,4,5),("Fred",1,1,1)]
>>> f2(liste)
```

Question 3 – Théorie (/4)

On souhaite comparer les performances de deux processeurs lors de l'exécution d'un même programme. Le premier processeur fonctionne à la cadence de 2GHz et le deuxième 4GHz.

La répartition des instructions du programme dans les 4 catégories d'instruction suivantes (le nombre d'instructions élémentaires appartenant à chaque catégorie) : « Math », « Load », « Store » et « Branch », se fait de la manière suivante pour les deux processeurs :

	Math	Load	Store	Branch
Processeur 1	1000	400	100	100
Processeur 2	2000	500	600	200

Sachant que l'exécution d'une instruction « Math » requiert 1 cycle d'horloge, 2 pour les instructions « Load » et « Store » et 3 pour les instructions « Branch ». Quel processeur exécutera le plus vite le programme ? Détaillez votre réponse.

$$1 + 2 + 2 + 3 = 8 \text{ cycles}$$

on proportionne les cycles
mais le processeur "2" a une cadence 2x supérieur
-> 2x moins de cycle requis !!!!

$$\text{1er processeur : } 2\text{GHz} : 0,5 + 1 + 1 + 1,5 = 4$$

$$\rightarrow 1000 \times 0,5 + 400 \times 1 + 100 \times 0,5 + 100 \times 1,5 = 1150 \text{ (2300)}$$

$$\text{2ème processeur : } 4\text{GHz} : 0,25 + 0,5 + 0,5 + 0,75 = 2$$

$$\rightarrow 2000 \times 0,25 + 500 \times 0,5 + 600 \times 0,5 + 200 \times 0,75 = 1200 \text{ (2400)}$$

=> le premier processeur est plus rapide

on peut aussi faire x1, x2, x2, x3 pour le 1er et x0.5, x1, x1, x1.5 pour le 2ème

Question 4 – Théorie (/3)

Comparez les systèmes de stockage de fichiers sous forme de « liste liée » et de « table indexée » selon les trois critères qui suivent. Lequel est le plus fiable ? Lequel est le plus performant en matière d'accès directs et lequel est le plus exigeant en terme d'espace mémoire sollicité ?

Liste Liée :

Moins fiable, accès direct plus compliqué, pas d'espace mémoire

Table indexée :

Plus fiable, accès direct simplifié, coûteux en mémoire pour placer les tables

Question 5 - Théorie (/3)

Le protocole Internet IP4 est en passe d'être remplacé par IP6 pour lequel les adresses IP sont codées sur 128 bit. Expliquez le principal problème que cela pose pour le système de routage et évoquez brièvement une solution à ce problème.

P6 => 2^{128} adresses IP

=> Table de routage gigantesque

=> Solution = hiérarchisation des systèmes d'adressage ou disparition du système de routage pour par exemple opter pour la commutation des paquets en mode connecté

Examen d'introduction à la microinformatique 1er Bachelor Solvay

Mardi 25/08/09

Nom :

Prénom :

Question 1 – Base de données relationnelles (/5) :

Une agence de location de voitures veut par l'entremise d'une base de données relationnelles facilement établir des factures concernant la location d'un véhicule pendant une période donnée. Ces factures concernent des clients et des voitures pouvant être tout deux de différentes catégories, le montant de la facture variant tant en fonction de la catégorie de la voiture que de celle du client (une catégorie seulement par client et par véhicule). Les véhicules se trouvent entreposés dans un lieu donné, un véhicule loué étant retiré dans un certain lieu et pouvant être restitué dans un autre. Le prix de la location dépend également de ces deux lieux (par exemple, plus cher s'ils sont différents). Finalement, la location sera sujette à la prise de différentes assurances (une location peut faire l'objet de plusieurs assurances (vol, dégât, ...)) et ces assurances à leur tour influenceront sur le prix. Réalisez le schéma de la base de données relationnelles, en ce compris les attributs principaux, qui permettra d'établir ces factures.

Question 2 – Python (/5)

Expliquez dans les grandes lignes ce que fait le petit code Python qui suit et retrouvez ce que l'exécution de chacune des 4 dernières lignes du code affichera à l'écran. La fonction « append(x) » rajoute 'x' en fin de liste.

```
>>> def func1(li1, li2, x, y):  
    li3 = []  
    for a in li1:  
        if (a != x):  
            if (a != y):  
                li3.append(a)  
    for a in li2:  
        if ((a != x) or (a != y)):  
            li3.append(a)  
    return li3
```

```
>>> def func2(li1, li2, x, y, n):  
    li4 = func1(li1, li2, x, y)  
    xx = 0  
    li5 = []  
    while (xx < n):  
        li5.append(li4[xx])  
        xx = xx + 1  
    return li5
```

```
>>> func2('titi', 'toto', 't', 'o', 3)  
>>> func2('solvay', 'orange', 'o', 'a', 5)  
>>> func2('ULB', 'Universite', 'U', 'L', 6)  
>>> func2('informatique', 'Python', 'P', 'y', 12)
```

Question 3 – Théorie (/3)

L'instruction suivante en Python : $A[12] = h + A[8]$ se traduit dans les instructions machines du processeur de la manière suivante :

```
load R1, 32(R3)
add R1,R2,R1
store R1,48(R3)
```

Expliquez ce qui sera contenu dans les registres R1, R2 et R3 pendant l'exécution de ces trois instructions élémentaires.

load R1, 32(R3)

cela veut dire que on load l'adresse 32 du registre 3 DANS le registre 1

=> le registre 1 est donc l'accumulateur

=> le registre 3 est le tableau A

add R1, R2,R1

cela veut dire que : $R1 \leftarrow R2 + R1$

=> le registre 2 <- H

store R1, 48(R3)

cela veut dire que on stocke la valeur de R1 à l'adresse 48 du R3

La seule corrélation possible puisqu'on ajoute la valeur du registre R3 à R1, puis qu'on modifie cette valeur initiale, puis qu'on remet au même endroit cette valeur qui a été modifié :

R3 représente A, H est R2 et R1 est l'accumulateur

On a au début en R3 la valeur 8 ($A[8]$), puis on ajoute H, et puis obtient 12 qu'on remet en A

Donc l'évolution du contenu des registres :

R1 : 0,8,12

R2 : 4

R3 : 8,12

REMARQUE

ici 2 ou 3 champs d'adresse => diff de 1 champs d'adresse
et on travaille avec des registres

http://www.fil.univ-lille1.fr/~wegrzyno/portail/Info/Doc/HTML/seq8_architecture.html

Question 4 – Théorie (/3)

Comparez les mécanismes d'accès à l'information désirée pour ces trois systèmes de stockage d'information : bande magnétique (comme les anciennes cassettes audio ou vidéo), CD Rom et mémoire RAM. Comparez surtout les temps d'accès en fonction de la position de l'information dans le système de stockage.

Bande Magnétique :

Accès séquentiel et temps d'accès dépendant de la position

CD Rom :

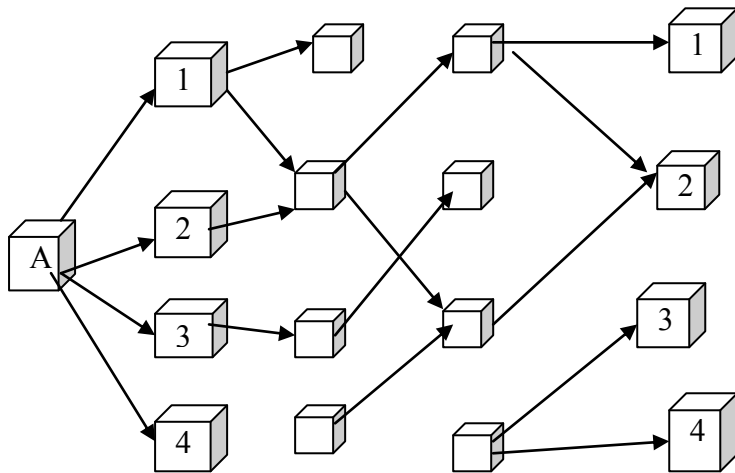
Accès en fonction de l'adresse mais temps d'accès dépendant de la position

RAM :

Accès en fonction de l'adresse mais temps d'accès indépendant de la position

on calcule chaque durée de transfert pour chaque dest finale.
 !!! le premier niveau contient un table de routage

Question 5 - Théorie (/4)



1: A -> 4 -> 1 -> 1 -> 1
 1 1 1 1 = 4 sec

2: A -> 3 -> 2 -> 2 -> 2
 1,5 1,5 1 1 = 5 sec

3: A -> 2 -> 3 -> 3 -> 3
 1 0,5 1 1 = 3,5 sec

4: A -> 1 -> 4 -> 4 -> 4
 1 1 0,5 0,5 = 3 sec

Un réseau Internet envoie des paquets au départ du nœud A vers les 4 nœuds finaux 1, 2, 3, 4. Pour cela les paquets passent par trois niveaux intermédiaires, chacun composés de 4 nœuds également, systématiquement libellés 1, 2, 3 et 4. Tous les nœuds d'un premier niveau sont connectés à tous les nœuds du suivant (la figure ne représente pas toutes les connexions). Les tables de routage de chacun des nœuds 1, 2, 3 et 4 sont telles que chaque paquet sera envoyé selon une logique spatiale. Ainsi, un paquet présent dans le nœud 1 devant arriver au nœud final 1 sera routé vers le nœud 1 suivant, s'il doit arriver à 2, il sera routé vers le nœud 2 suivant. De même, un paquet présent dans le nœud 3, devant arriver au nœud final 1, sera routé vers 1, et vers 2 s'il doit arriver au 2. Seule la table de routage du nœud A se particularise et est donnée comme suit :

Nœud suivant	Nœud final
4	1
3	2
2	3
1	4

Finalement, les temps de transfert entre deux nœuds sont donnés comme suit pour le nœud de départ :

A->1 = 1 sec
 A->2 = 1 sec
 A->3 = 1,5 sec
 A->4 = 1 sec

Entre chaque niveau, y compris jusqu'au dernier, les temps de transfert sont identiques et donnés comme suit:

1->1 = 1 sec, 1->2 = 1 sec, 1->3 = 0,5 sec, 1->4 = 1 sec
 2->1 = 1 sec, 2->2 = 1 sec, 2->3 = 0,5 sec, 2->4 = 1 sec
 3->1 = 2 sec, 3->2 = 1,5 sec, 3->3 = 1 sec, 3->4 = 1 sec
 4->1 = 1 sec, 4->2 = 0,5 sec, 4->3 = 1 sec, 4->4 = 0,5 sec

Au départ du nœud A, ordonnez par durée du transfert les destinations finales d'arrivée 1, 2, 3 et 4 ? Justifiez votre réponse.

Examen d'introduction à la microinformatique 1er Bachelor Solvay

Samedi 16/1/2010

Nom :

Prénom :

Question 1 – Base de données relationnelles (/5) :

Les organisateurs d'une conférence au sujet de l'informatique en général souhaitent encoder toute l'organisation de cette conférence dans une base de données relationnelle. La conférence comprendra un ensemble de thèmes (OS, sécurité, programmation, réseaux...) et les présentations qui s'y dérouleront seront organisées sous forme de sessions. Chaque session portera sur un thème donné (plusieurs sessions seront consacrées à un même thème) et comprendra un certain nombre de présentations. Plusieurs auteurs peuvent être responsables d'une même présentation et un même auteur peut être impliqué dans plusieurs présentations. Le rôle de chaque auteur dans une présentation sera précisé en différenciant les premiers auteurs, des seconds, troisièmes, etc... Chaque session sera sous la responsabilité d'un « chairman » qui pourra prendre en charge plusieurs sessions. Finalement, toute session se conclura par un débat auquel prendront part plusieurs auteurs (un même auteur pouvant participer à plusieurs débats) et qui sera dirigé par un modérateur (forcément un des auteurs participant au débat et qui ne sera modérateur qu'une seule fois durant toute la conférence). Réalisez le schéma de la base de données relationnelles qui permettra de facilement encoder et retrouver ces différentes informations.

Question 2 – Python (/5)

Expliquez dans les grandes lignes le rôle du petit code Python qui suit et retrouvez ce qui sera affiché à l'écran à l'exécution des quatre dernières instructions du code. Dans l'utilisation d'un dictionnaire (dans lequel tous les éléments « y » sont indexés par un « x »), items() itère sur tous les éléments x,y du dictionnaire et dic[x] renvoie le « y » correspondant au « x ». L'instruction « str=str.replace('a','b') » remplace dans une chaîne de caractère « str » toutes les occurrences de la lettre 'a' par la lettre 'b'.

```
>>> def func(a):
    dic = {}
    dedans = False
    c=1
    for x in a:
        for y,z in dic.items():
            if (x == y):
                dic[y]=dic[y]+1
                dedans = True
        if (not dedans):
            dic[x]=c
        dedans = False
    return dic
```

```
>>> def change(m):
    dic = func(m)
    n=m
    for x in m:
        for y,z in dic.items():
            if (x==y and z == 2):
                n=n.replace(y,'z')
    return n
```

```
>>> change("solvay")
>>> change("pipo")
>>> change("maman")
>>> change("java")
```

Question 3 – Théorie (/3)

Qu'est-ce-que l'appel dans un programme (Python par exemple) d'une fonction à l'intérieur d'une autre fonction partage avec le mécanisme de « l'interruption » ? Développez en quelques lignes ?

Sauvegarde du compteur d'instruction et de tous les registres mémoires utilisés par la fonction appelante dans une pile mémoire. Récupération de tous ces registres à la fin de l'exécution de la fonction appelée.

Question 4 – Théorie (/3)

On s'intéresse au fonctionnement de la mémoire cache d'un processeur. On suppose que ce dernier présente un taux de raté de 2% pour les instructions et de 4% pour les données à aller chercher dans la mémoire RAM. On suppose également que la fréquence d'accès des données est de 36% (36% des instructions requièrent d'aller chercher des données en mémoire). On suppose finalement que chaque exécution d'instruction nécessite 2 cycles d'horloge lorsqu'elle se déroule sans raté et que, lors d'un raté, la recherche et l'exécution d'une instruction ou d'une donnée exige 100 cycles d'horloge supplémentaire. Retrouvez en moyenne le gain en temps d'exécution que permettrait une cache parfaite, sans raté.

le processeur effectue chaque instruction l'une à la suite de l'autre et prend 2 cycles d'horloge par instruction et a 2% d'erreur

On a 4% de ratés qui impliquent 100 cycles d'horloge supplémentaires.

Donc pour calculer le temps moyen avec ces 4% de ratés il faut faire :

$$\begin{aligned}(0.96 \times 2 + 0.04 \times 102) &= 6 && \text{pour les données} \\ (0.02 \times 102 + 0.98 \times 2) &= 4 && \text{pour les instructions}\end{aligned}$$

donc vu qu'il y a 36% d'accès aux données :

$$\Rightarrow 0.36 \times 6 + 0.64 \times 4 = 4.72 \text{ cycles/instruction}$$

le temps moyen SANS échec étant de 2 cycles/instruction

$$\Rightarrow \text{Donc le gain d'efficacité est de } 2.72 \text{ cycles/instruction}$$

Question 5 - Théorie (/4)

SUCCESION DES PAGES PAR PROCESS


<u>P1</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>	<u>5</u>	<u>5</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>
<u>P2</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>2</u>	<u>3</u>	<u>2</u>	<u>3</u>	<u>2</u>	<u>3</u>	<u>2</u>	<u>3</u>	<u>2</u>	<u>3</u>
<u>P3</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>1</u>	<u>2</u>	<u>2</u>	<u>2</u>	<u>1</u>	<u>6</u>	<u>6</u>	<u>1</u>
<u>P4</u>	<u>5</u>	<u>6</u>	<u>5</u>	<u>6</u>	<u>1</u>	<u>1</u>	<u>6</u>	<u>6</u>	<u>1</u>	<u>8</u>	<u>8</u>	<u>8</u>	<u>5</u>	<u>1</u>	<u>5</u>	<u>1</u>

Quatre process (P1,P2,P3 et P4) doivent s'exécuter simultanément dans un même processeur. Dans un mécanisme de mémoire virtuelle, la succession des pages (chaque instruction successive sera dans une des pages renseignées) dont aura besoin chaque process est indiquée dans le tableau du haut. Ainsi, les premières pages dont aura besoin le process P1 pour s'exécuter seront successivement 1, 2 et 3. Le temps d'occupation du processeur par un process (que l'on appelle un « quantum de temps ») ne lui permet que d'exécuter quatre instructions consécutives et donc charger si nécessaire 4 pages au plus dans la mémoire RAM. Chaque page occupe 1 MByte. La mémoire RAM ayant une capacité maximale de 16MByte, après combien de « quantums de temps » la mémoire RAM dépassera sa capacité et exigera des « swap » avec le disque dur ?

chaque quantum exécute 4 instructions (la première de chaque process puis la seconde etc)

Il faut mémoriser les pages en mémoire au fil du temps et noter le temps lorsque ce nombre de pages différentes dépasse 16 (pages différentes par process dont la somme fait 16).

REPONSE : après 11 quantum de temps

quantum de temps 
1 2 3 4 5 6 7 8 9 10 11

P1 : 1 2 3 4 5 6 5 5 1 2 3 => 6 pages diff (1.2.3.4.5.6)

P2 : 2 3 4 2 3 4 2 3 2 3 2 => 3 pages diff (2.3.4)

P3 : 1 2 3 4 1 2 3 4 1 2 2 => 4 pages diff (1.2.3.4)

P4 : 5 6 5 6 1 1 6 6 1 8 8 => 4 pages diff (1.5.6.8)

au 11ème quantum de temps on a dépassé 16 pages
en effet $6+3+4+4 = 17$

Examen d'introduction à la microinformatique 1er Bachelor Solvay

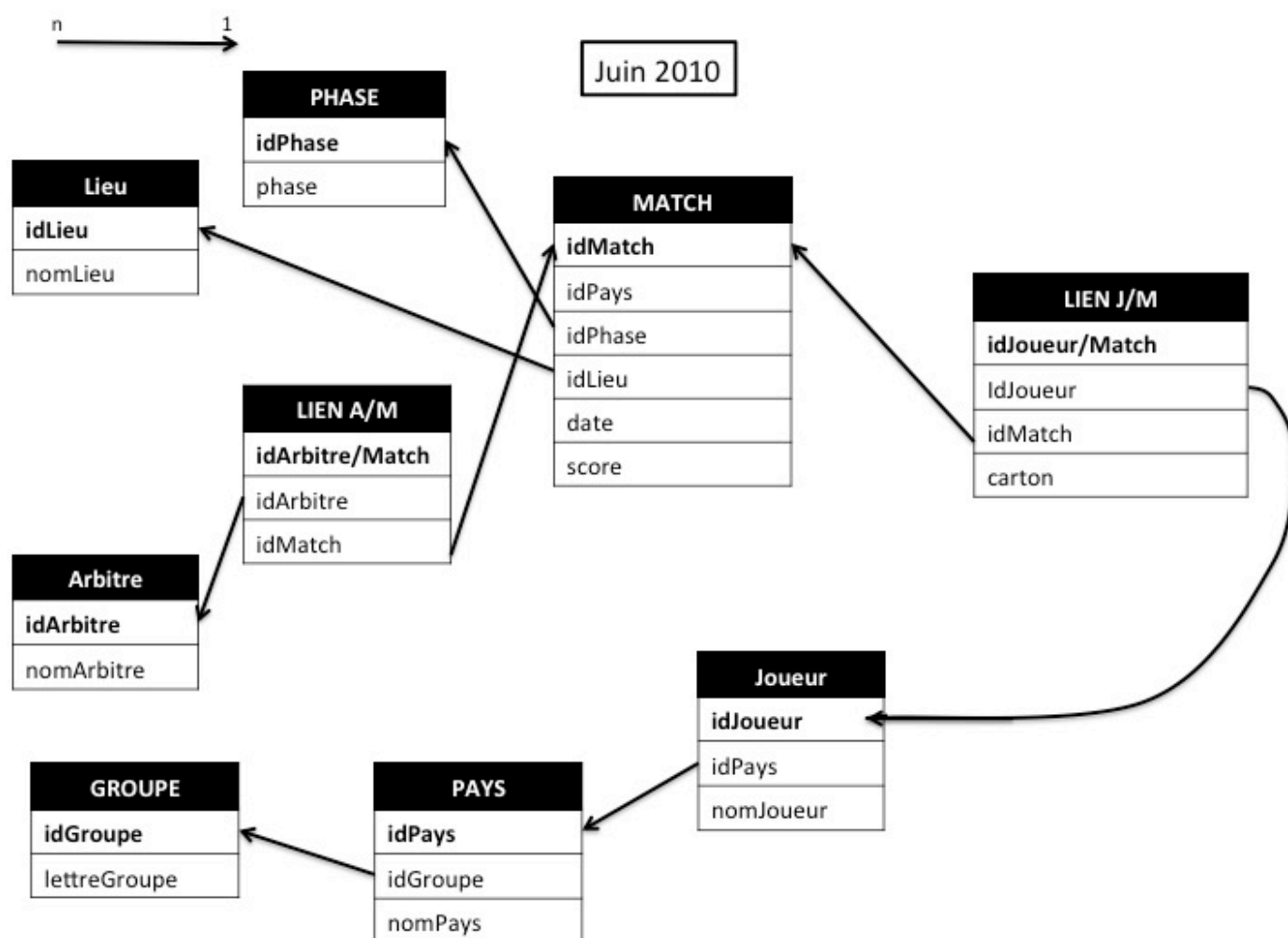
Mardi 8/6/2010

Nom :

Prénom :

Question 1 – Base de données relationnelles (/5) :

Les organisateurs de la prochaine coupe du monde de football en Afrique du Sud souhaitent stocker dans une base de données relationnelles les informations les plus importantes concernant le déroulement de la compétition. Celle-ci s'organise en une succession de phases (éliminatoires, huitièmes de final, quart de final...) qui chacune occasionne un certain nombre de matchs. Les équipes de football sont regroupées par groupe (groupe A, groupe B,...) et comprennent un certain nombre de joueurs affiliés à l'équipe. Les organisateurs souhaitent encoder pour chacun des match : la date et le lieu où il se déroule (une des villes d'Afrique du Sud qui accueillera plusieurs matchs, reprise dans la base de données avec ses caractéristiques propres), tous les joueurs étant montés sur le terrain durant le match, le score du match, les différents arbitres responsables du match (il peut y avoir plusieurs arbitres par match qui peuvent bien entendu arbitrer plusieurs matchs), et si des joueurs ont reçu des cartons (quel joueur). Réalisez le schéma de la base de données relationnelles qui permettra de facilement encoder et retrouver ces différentes informations.



Question 2 – Python (/5)

Expliquez dans les grandes lignes le rôle du petit code Python qui suit et retrouvez ce qui sera affiché à l'écran à l'exécution de chacun des trois derniers groupes d'instructions. Dans l'utilisation d'un dictionnaire (dans lequel tous les éléments « y » sont indexés par un « x »), `items()` itère sur tous les éléments x,y du dictionnaire et `dic[x]` renvoie le « y » correspondant au « x ». L'instruction "`list.append(x)`" rajoute "x" en fin de la liste.

```
*****
```

```
>>> def f2(dic,x):
    list=[]
    for y,z in dic.items():
        if z==x:
            list.append(y)
    list.append(x)
    return list

>>> def f(l1,l2):
    dic={}
    g=0
    for x in l1:
        z=0
        for y in l2:
            if (x%y==0):
                if y >= z:
                    z=y
        if (z != 0):
            dic[x]=z
            if z >= g:
                g=z
    return f2(dic,g)
```

```
>>> l1=[10,8,9,6,12]
>>> l2=[1,2,3,4]
>>> f(l1,l2)
```

```
>>> l1=[5,7,10,13,15,20]
>>> l2=[1,2,5,10]
>>> f(l1,l2)
```

```
>>> l1=[1,10,14,20,24,30]
>>> l2=[2,3,4,5]
>>> f(l1,l2)
```

```
*****
```

Question 3 – Théorie (/3)

Un processeur possède les registres de r1 à r8. Il possède aussi un registre accumulateur qui est la source et la destination de toutes les opérations. Il s'agit d'un processeur RISC dont toutes les instructions élémentaires ne contiennent qu'une opérante et agissent toutes à partir de l'accumulateur. Leur liste suit :

Add rx
Sub rx
Mul rx
Div rx
Load rx
Store rx

« rx » peut être n'importe lequel des 8 registres.

Ecrivez la suite d'instructions élémentaires obtenues par la compilation de l'expression :

$((r1 \times r2 - r3)/(r1 + r4 + r5)) + r6 + r2$

la liste des instructions élémentaires est donnée

on commence tjrs par les multiplication/division ET par l'instruction LOAD

load r1
add r4
add r5

store r7

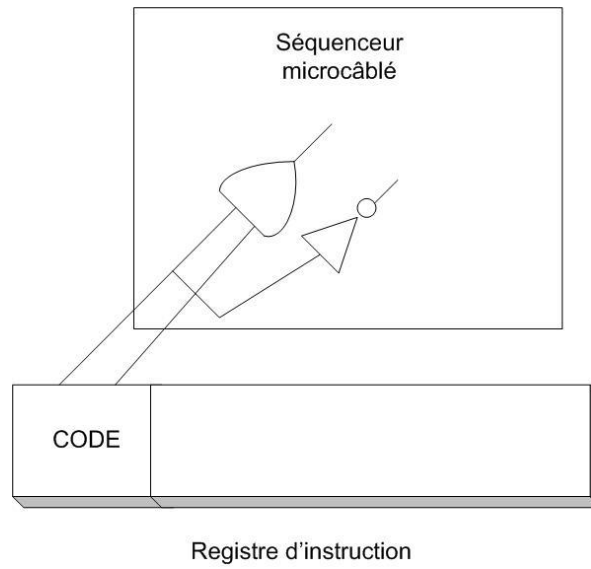
load r1
mult r2
sub r3

div r7

add r6
add r2

Question 4 – Théorie (/4)

Expliquez en quelques lignes comment vous comprenez dans cette figure les connexions existants entre la première partie du registre d'instruction et le séquenceur microcâblé.



Question 5 - Théorie (/3)

Soit un processeur dont la capacité en RAM permet de ne contenir que trois pages. Soit la succession suivante des pages auquel ce même processeur doit pouvoir accéder : {P0,P1,P2,P3,P1,P3,P3,P1,P0,P3,P3,P2,P4,P3}. On imagine deux mécanismes de mémoire virtuel avec deux stratégies de remplacement des pages : la première remplaçant celle arrivée en dernier, la deuxième remplaçant celle arrivée en premier. Laquelle des deux stratégies fonctionne le mieux dans ce cas.

1ère stratégie : "LIFO" (comme en compta)

P0 P1 P2 : départ (les 3 premières de la liste)

P0 P1 P3 (le P3 remplace le P2)

à partir d'ici les nouvelles pages sont déjà dans les 3 pages dans la RAM
donc on ne change rien jusqu'à

P0 P1 P2

P0 P1 P4

P0 P1 P3

=> 4 REMPLACEMENTS de pages

2ème stratégie : "FIFO"

P0 P1 P2 : départ

P1 P2 P3 (le P3 remplace le P0)

à partir d'ici les nouvelles pages sont déjà dans les 3 pages dans la RAM
donc on ne change rien jusqu'à

P2 P3 P0

P3 P0 P4

Technique utilisée :

<https://www.youtube.com/watch?v=KejyTiATz18>

=> 3 REMPLACEMENTS de pages

==> deuxième stratégie

Examen d'introduction à la microinformatique
2ème Bachelor Solvay

Mardi 17/1/2012

Nom :

Prénom :

Question 1 – Python (/6)

Expliquez dans les grandes lignes le rôle du petit code Python qui suit et surtout ce que renverra la dernière fonction « select » si elle reçoit comme premier paramètre un nom de fichier « texte » et comme deuxième une liste de noms de fichier « texte ». L'instruction « `line.lower().split()` » renvoie une liste des mots en caractère minuscule compris dans la ligne. L'instruction « `str=str.replace('a','b')` » renvoie la même chaîne de caractères dans laquelle toutes les occurrences de la lettre 'a' sont remplacées la lettre 'b'.

```
>>> stop=".,"
```

```
>>> def stopW(word):  
    l=word  
    for char in stop:  
        l=l.replace(char,"")  
    return l
```

```
>>> def computeBOW(fileName):  
    W={}  
    for line in open(fileName):  
        for word in line.lower().split():  
            word = stopW(word)  
            if len(word) > 2:  
                if word not in W:  
                    W[word]=0  
                W[word] = W[word] + 1  
    return W
```

```
>>> def PS(fn1,fn2):  
    W1 = computeBOW(fn1)  
    W2 = computeBOW(fn2)  
    res = 0  
    for x in W1:  
        for y in W2:  
            if (x==y):  
                res=res+W1[x]*W2[y]  
            break  
    return res
```

```
>>> def select(filename,filelist):  
    min=0  
    for f in filelist:  
        dist=PS(filename,f)  
        if dist > min:  
            sel = f  
            min = dist  
    return sel
```

Question 2 – Base de données relationnelles (/6)

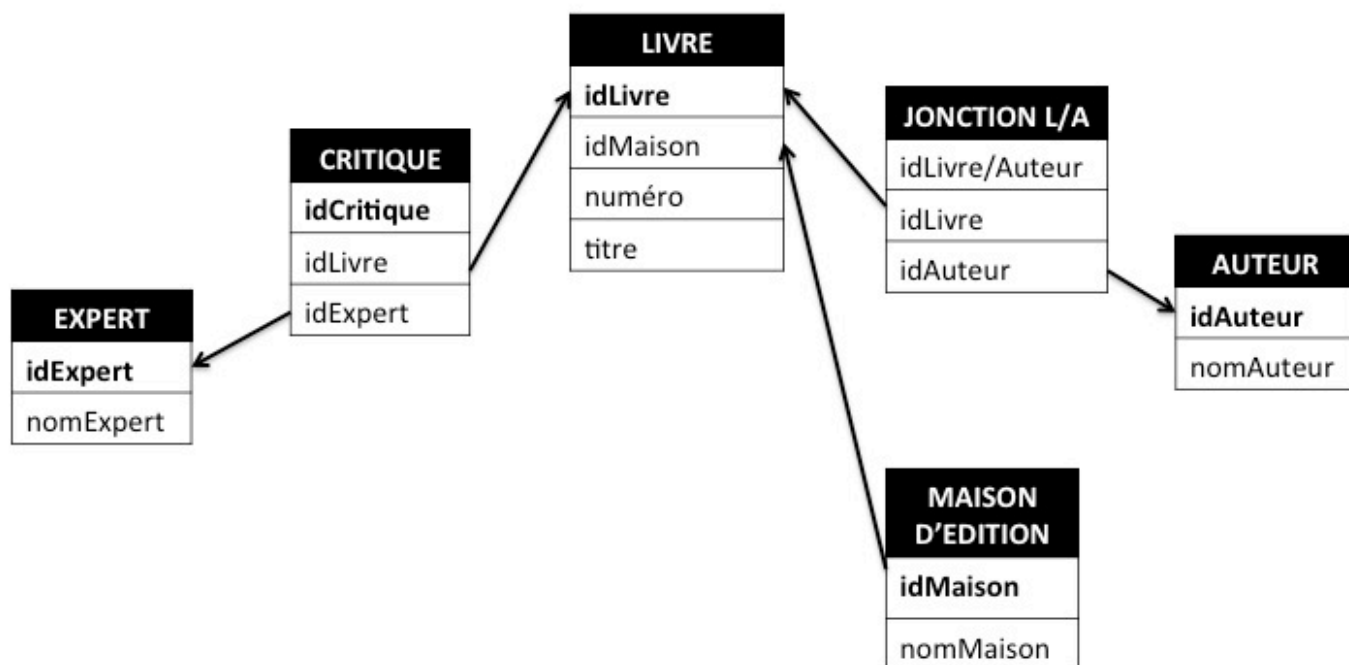
Les responsables d'un magazine littéraire souhaitent constituer une base de données relationnelles dans laquelle ils encoderaient les livres critiqués dans leurs numéros respectifs avec, pour chaque livre, son auteur et sa maison d'édition (un livre peut être écrit par plusieurs auteurs mais sort toujours d'une seule maison d'édition). Ce magazine faisant appel à un même comité d'experts littéraires afin d'évaluer les ouvrages, les responsables souhaitent également pouvoir retrouver les articles critiques rédigés par ces experts pour chacun des ouvrages.

Réalisez le schéma de la base de données relationnelles qui permettra de facilement encoder et retrouver ces différentes informations en y indiquant les attributs les plus importants.

Ecrivez la requête SQL qui permettra l'ajout d'un nouveau livre dans la base de données et celle qui permettra de récupérer toutes les critiques qu'un auteur précis Mr X a reçu pour tous les livres qu'il a publié.

n → 1

Janvier 2012



Question 3 – Théorie (/4)

Expliquez dans la gestion des mémoires informatiques les principes de localité spatiale et temporelle et expliquez en quoi ces deux principes justifient l'organisation hiérarchique de ces mêmes mémoires.

Les mémoires avec le plus de place sont les moins chères mais les plus lentes et sont loin du processeur donc on met toutes les infos dans les grosses mémoires et on copie les infos qu'on utilise le plus souvent dans les petites mémoires qui sont plus rapides et plus proches du processeur. Tout ça pour optimiser au mieux le rapport "vitesse/prix"

Question 4 - Théorie (/4)

On considère 5 processus P1,P2,P3,P4 et P5 dont les caractéristiques sont résumées dans le tableau suivant (un petit numéro de priorité indique une priorité forte).

	Ordre d'arrivée	Durée d'exécution en sec.	Priorité
P1	1	2	2
P2	2	6	4
P3	3	10	3
P4	4	4	5
P5	5	12	1

On suppose un ordinateur doté d'un unique processeur et d'un système d'exploitation autorisant plusieurs politiques d'ordonnancement de ces processus : « par ordre d'arrivée », « par ordre de durée », « par ordre de priorité » et en « round robin », allouant 2 secondes de temps pour chaque processus tour à tour.

Pour chacune de ces politiques calculez le temps moyen que prendra en tout l'exécution d'un processus (le temps total nécessaire à son exécution en ce compris les temps d'attente) et ordonnez par ordre croissant de cette mesure les différentes politiques.

temps moyen = temps d'exécution de chaque processus / 5 processus
!!!! le temps total nécessaire à son exécution en ce compris les temps d'attente)

par ordre d'arrivée :

$$(2 + 8 + 18 + 22 + 34) / 5 = 16,8\text{sec}$$

pa orde de durée :

$$(2 + 6 + 12 + 22 + 34) / 5 = 15,2\text{sec}$$

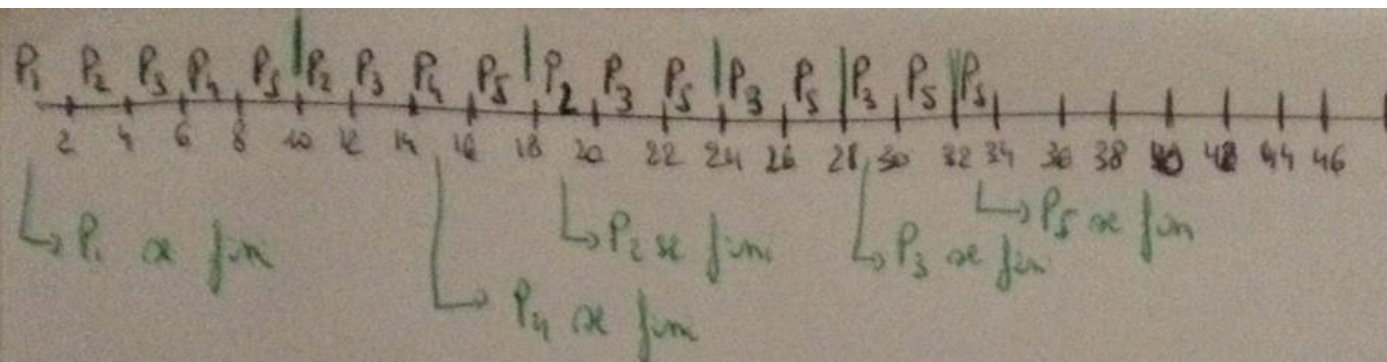
par ordre de priorité :

$$(12 + 14 + 24 + 30 + 34) / 5 = 22,8\text{sec}$$

en round robin :

on alloue 2 sec a chaque processus tour à tour, après chaque round on check combien de processus ne sont pas encore fini...

page suivante...



$$P_1 = 2 \text{ sec}$$

$$P_2 = 20 \text{ sec}$$

$$P_3 = 30 \text{ sec}$$

$$P_4 = 16 \text{ sec}$$

$$P_5 = 34 \text{ sec} \quad \left(\frac{2 \cdot 5 + 2 \cdot 4 + 2 \cdot 3 + 2 \cdot 2 + 2 \cdot 2 + 2}{\text{(De la réponse fautive)}} \right)$$

$$\Rightarrow \frac{P_1 + P_2 + P_3 + P_4 + P_5}{5} =$$

$$= \frac{2 + 20 + 30 + 16 + 34}{5} = 20,4 \text{ sec}$$

Examen d'introduction à la microinformatique Solvay BA2

Mardi 04/09/12

Nom :

Prénom :

Question 1 – Base de données relationnelles (/6) :

Les organisateurs des Jeux Olympiques de Londres 2012 maintiennent toutes les informations concernant les épreuves dans une base de données relationnelle. On y retrouve les différents sports (athlétisme, voile, boxe, tir..), la ville où ils se produisent, et pour chaque sport les disciplines qui lui sont associées (100m, 1000m, laser, poids plume ...). Pour chaque discipline sportive sont reprises les différentes séries (il y a par exemple 8 séries du 100m avant la finale) avec la date et heure précise du départ de la série, dans lesquelles concourent finalement les athlètes. Les athlètes sont également enregistrés avec pour chacun son sexe, son pays d'origine, son âge, son poids... Chaque série reprend les athlètes qui y participent en tenant compte qu'un même athlète peut participer à plusieurs séries. On veut également retrouver pour chaque participation d'un athlète à une série son classement et son résultat (temps, distance,...). Il faut aussi indiquer dans la base de données les équipes qui participent à certaines séries lorsque ces dernières se particularisent justement par la participation d'équipes plutôt que d'athlètes individuels (par exemple, pour l'aviron ou le football), avec pour chaque équipe la possibilité de retrouver les athlètes qui la composent (un athlète n'appartient qu'à une équipe). Tout comme pour les athlètes, une équipe peut participer à plusieurs séries. Finalement, on veut relier à chaque discipline son classement final (par défaut les 100 premiers (athlète ou équipe) de la discipline). Réalisez la base de données relationnelles permettant d'enregistrer et retrouver toutes ces informations en y indiquant les attributs les plus relevant.

Ecrivez une première requête SQL qui permet d'inscrire un athlète donné par son nom à une série et une deuxième qui permet d'obtenir son classement pour cette série. Ces requêtes, bien que pouvant rester approximatives, doivent reprendre toutes les informations nécessaires à leur bon fonctionnement.

Question 2 - Python (/6) :

Expliquez dans les grandes lignes le rôle du code Python qui suit et retrouvez ce qui sera affiché à l'écran à l'exécution de la dernière instruction. La fonction « remove(a) » supprime l'élément « a » de la liste. La fonction « insert(n,a) » insère l'élément « a » en position « n » de la liste. La fonction « append(a) » rajoute l'élément « a » en fin de liste. La fonction « len(lis) » donne la longueur de la liste « lis ». Le « % » est le modulo : le reste de la division.

```
>>> li1=[20,24,8,6,10,32,15,3]
```

```
>>> def fl(l1):
```

```
    pos=0
```

```
    for a in l1:
```

```
        if a >= 20:
```

```
            x=a
```

```
            pos=pos+1
```

```
            l1.remove(a)
```

```
            l1.insert(0,x)
```

```
    l1.append(pos)
```

```
    return l1
```

```
>>> def pg(x,ly):
```

```
    m=0
```

```
    for a in ly:
```

```
        if x%a == 0:
```

```
            if a > m:
```

```
                m=a
```

```
    return m
```

```
>>> def end(lis):
```

```
    dx = {}
```

```
    lx = fl(lis)
```

```
    for a in lx:
```

```
        if a >= 20:
```

```
            loc=[]
```

```
            x=lis[len(lis)-1]
```

```
            i=x
```

```
            while i < len(lis)-1:
```

```
                loc.append(lis[i])
```

```
                i=i+1
```

```
            dx[a]=pg(a,loc)
```

```
    return dx
```

```
>>> end(li1)
```

Question 3 : Théorie (/4):

Expliquez les avantages et défauts des trois mécanismes de stockage physique de fichiers sur les mémoires secondaires (tel que le disque dur) : stockage en continu, stockage en liste liée et stockage à l'aide des tables indexées.

Question 4 – Théorie (/4)

Soit le génome suivant composé de ces fameuses quatre lettres : G,T,A,C.

GGGGGGTTTTAAAAAAAGGGGGCCCCGGGTTTTAAAAAAAGGGGGGTTTAACC
CCGGGG

Proposez un mode de stockage binaire pour ce génome qui serait le plus économe qui soit.

$$\begin{array}{ll} 6G + 5 + 3 + 6 + 4 & = 24 \text{ G} \\ 4T + 3 + 3 & = 10 \text{ T} \\ 8A + 8 + 2 & = 18 \text{ A} \\ 4C + 4 & = 8 \text{ C} \end{array}$$

-> 60 lettres < $2^6 = 64$ => 6 bits

-> 2 bits pour les lettres

ATTENTION : pas le mm exo que l'exam 2008 (écrire plusieurs génomes)

=> REPONSE : On veut pour CE génome

code :

G : 0
A : 10
T : 110
C : 1110

Pour pas qu'il y ai d'ambiguïté entre les lettres.

Explication :

on remplace les lettres du génome par les 1 et 0 suivant le code pour avoir le code binaire...

pour remettre ce code binaire sous forme de GTAC on doit compter le nombre de 1 qu'il y a entre deux 0.

Si il y en a 1 c'est un A, si il y en a 2 c'est un T, si il y en a 3 c'est un C et si il y en a pas c'est un G.

Examen d'introduction à la microinformatique Solvay BA2

Mardi 03/09/2013

Nom :

Prénom :

Question 1 – Base de données relationnelles (/6) :

Une conférence d'économie théorique d'une semaine se déroule dans un ensemble de salles de l'hôtel Sheraton de Bruxelles. La conférence couvre un grand nombre de thèmes : « macroéconomie », « microéconomie », « finance », « théorie des jeux ». La conférence s'organise en un ensemble de sessions se déroulant en parallèle. Chaque session est associée à un thème unique, et une session (telle « macroéconomie 1 », « macroéconomie 2 ») dure exactement une demi-journée. De plus, une session se déroule dans une salle unique. Une session comprend un ensemble d'exposés (dont il est important de savoir à quel moment de la demi-journée ils se dérouleront) et est modéré par deux ou trois chairmans, bien qu'un de ces chairmans puisse avoir à modérer plusieurs sessions (organisés à des périodes diverses). Ces chairmans seront également responsables de la sélection du meilleur exposé de la session. A la fin de la conférence, un prix sera attribué au meilleur exposé de la conférence à choisir parmi les meilleurs des sessions. Plusieurs orateurs peuvent se partager un même exposé et bien évidemment un même orateur peut présenter plusieurs exposés durant la conférence.

Réalisez la base de données relationnelles permettant d'enregistrer et retrouver toutes ces informations en y indiquant les attributs les plus relevant.

Ecrivez une première requête SQL qui permet d'inscrire un nouvel exposé à une session donnée et une deuxième qui permet de retrouver tous les titres des exposés présentés au cours d'une session dont on donne l'intitulé. Ces requêtes, bien que pouvant rester approximatives, doivent reprendre toutes les informations nécessaires à leur bon fonctionnement.

Question 2 - Python (/6) :

Au départ d'une liste (« liste ») contenant un très grand ensemble de mots, décrivez en quelques lignes quel est l'objectif du code Python qui suit, et ce qui se déroulera à l'exécution de la fonction 'M()'. « *randint(a,b)* » renvoie un nombre aléatoire entre a et b inclus. « *x= input('Allez-y')* » affiche 'Allez-y' et met dans la variable « x » ce que l'on entre au clavier, « *len(list)* » donne la longueur de la liste.

```
>>> liste=["pipo", "etudiant", .....]
```

```
>>> def M2(x,y):
    a=input("Allez-y: ")
    j=0
    yy=""
    for le in x:
        if a==le:
            yy+=a
        else:
            yy+=y[j]
        j+=1
    print (yy)
    return yy
```

```
>>> def M1(x):
    y=""
    for a in x:
        y+='-'
    print (y)
    tt=0
    while (tt < 5):
        y=M2(x,y)
        xx=input("Faites une proposition: ")
        if xx==x:
            print("Gagne!!!!!!!!!!")
            tt=5
        else:
            if(tt==4):
                print("Perdu!!!!!!!!!!")
            tt+=1
    vv=input("Une fois encore ?: ")
    return vv
```

```
>>> def M():
    re="oui"
    while (re=="oui"):
        m=liste[random.randint(0,len(liste)-1)]
        re = M1(m)
```

```
>>> M()
```

Question 3 : Théorie (/4):

Soit un processeur stockant les nombres entiers sur 6 bits en notation en « complément à 2 », réalisez l'opération binaire suivante « -17-16 » (indiquez l'opération en binaire) et discutez le résultat en quelques lignes.

en 6 bits complément à 2

-17 : on code 17 : 010001, on inverse => 101110, on ajoute 1 => 101111

-16 : on code 16 : 010000, on inverse => 101111, on ajoute 1 => 110000

on calcule : 101111 (-17)
 + 110000 (-16)

011111 (NAN) -> Not A Number

dépassement de la capacité (Overflow) : opérante de mm signe et signe du résultat différent.

On le voit direct si les 2 chiffres commence par 1 (si complément à 2)

Question 4 – Théorie (/2)

L'adresse IP d'un serveur particulier installé sur le réseau de l'ULB est 164.15.59.76. Expliquez en quelques lignes la traduction binaire de cette adresse et le rôle joué par les 16 premiers bits.

164 : 10100101 par le principe $(1*128+0*64+1*32+0*16+0*8+1*4+0*2+0*1) = 164$
15 : 00001111 $(0*128+0*64+0*32+0*16+1*8+1*4+1*2+1*1) = 15$
59 : 00111011 etc
76 : 01001100 etc

=> 10100101.00001111.00111011.01001100. = 32 bits (DE BASE)

on commence la sequence par 10 => Classe B (voir page 237)

=> 2bits (10) + 14bits (adresse réseau) + 16bits (adresse station) = 32

Question 5 – Théorie (/3)

Remplissez les différentes cases de ces cinq tableaux de sorte à illustrer le fonctionnement de la mémoire virtuelle.

????????????????????
 ?????????????????

[illegible]

[illegible]

Examen d'introduction à la microinformatique Solvay BA2

Mardi 15/01/2013

Nom :

Prénom :

Question 1 – Base de données relationnelles (/6) :

Un directeur d'école décide d'exploiter une base de données relationnelle afin d'y stocker toutes les informations des classes de neige qu'il est en train d'organiser et surtout de planifier activités et participants (enfants et professeurs). Ces classes se structurent autour de deux types d'activité : les activités d'intérieur (cours scolaires, ateliers bricolages, musique, théâtre...) et les cours de ski. Les enfants de l'école sont répertoriés par la classe qu'ils suivent dans l'école, les professeurs par les classes dans lesquelles ils enseignent (une même classe sollicite plusieurs professeurs). Enfants et professeurs participent aux deux types d'activité selon les modalités suivantes. Les cours de ski sont divisés en groupes selon les niveaux (groupe 1, 2, 3 et 4). Les professeurs participent aux cours de ski au même titre que les enfants (c'est-à-dire aussi selon leur niveau). Un cours de ski (d'une demi-journée à enregistrer) ne concerne toujours qu'un seul groupe de ski. Chaque groupe de ski est à la charge d'un moniteur affecté par la station de ski à l'école. Les professeurs de l'école sont également en charge des activités d'intérieur (plusieurs professeurs pour une activité et plusieurs activités par professeur). On désire cependant identifier le professeur responsable de l'activité. Les enfants sont répartis en groupes d'activité, essentiellement selon leur âge et donc la classe qu'ils fréquentent à l'école, plusieurs classes pouvant composer un même groupe d'activité. Une même activité sera suivie par plusieurs groupes d'activités et un même groupe d'activité participera à plusieurs activités à des heures différentes (certaines activités seront inaccessibles à certains groupes, l'horaire des activités pour les groupes devra également être enregistré dans la base de données).

Réalisez la base de données relationnelles permettant d'enregistrer et retrouver toutes ces informations en y indiquant les attributs les plus relevant.

Ecrivez une première requête SQL qui permet d'inscrire un groupe d'activité à une activité donnée et une deuxième qui permet d'afficher le planning des cours de ski d'un enfant donné. Ces requêtes, bien que pouvant rester approximatives, doivent reprendre toutes les informations nécessaires à leur bon fonctionnement.

Question 2 - Python (/6) :

Expliquez le rôle du code Python qui suit et décrivez en quelques lignes ce qui se déroulera à l'exécution de la fonction 'tot()'. « randint(a,b) » renvoie un nombre aléatoire entre a et b inclus. « input('Allez-y') » affiche 'Allez-y' et lit ce que l'on entre au clavier.

```
*****
>>> dico={}
>>> def test(x):
    x1 = random.randint(x,10*x)
    y1 = random.randint(x,10*x)
    z1 = random.randint(1,4)
    r1 = 0
    if (z1 == 1):
        print (str(x1)+' + '+str(y1))
        r1 = x1 + y1
    elif (z1 == 2):
        print (str(x1)+' - '+str(y1))
        r1 = x1 - y1
    elif (z1 == 3):
        print (str(x1)+' x '+str(y1))
        r1 = x1 * y1
    else:
        print (str(x1)+' / '+str(y1))
        r1 = x1 / y1
    rx = input ('Allez-y: ')
    if (float(rx) == r1):
        return 1
    else:
        return 0
>>> def fi():
    na = input('nom: ')
    for x,y in dico.items():
        if x == na:
            return na
    dico[na]=1
    return na
>>> def tot():
    nn = True
    while (nn):
        na = fi()
        jj=0
        xx=0
        while (xx < 20):
            jj+=test(dico[na])
            xx+=1
        if (jj > 10):
            dico[na]*=10
        nn = input('On continue? False=nom, True=oui: ')
```

Question 3 : Théorie (/4):

Un processeur dispose d'une mémoire cache de quatre lignes. Les adresses mémoires sont sur 16 bits. Chaque mot recherché dans la mémoire fait 32 bits, mais la mémoire centrale est adressable par octet (byte). Chaque ligne de la cache peut contenir jusque 4 mots (de 32 bits toujours).

Retrouvez le découpage des adresses dans les trois parties qui caractérisent le fonctionnement de la cache.

!!! on travaille en bytes

on a 4 lignes

-> 2^2 lignes => 2bits de LIGNE

on essaye de calculer la dimension

4 lignes de 4 mots de 32 bits (4 bytes) = $4 \times (4 \times 4) = 64$ bytes (2^6)

64 bytes / 4 lignes = 16 bytes (longueur d'une ligne)

-> 2^4 bytes => 4 bits d'OCTET

(remarque : $2^2 \cdot 2^4 = 2^6$)

longueur adresse mémoire : 16 bits

$16 - 4 - 2 = 10$

-> 2 bits (LIGNE)

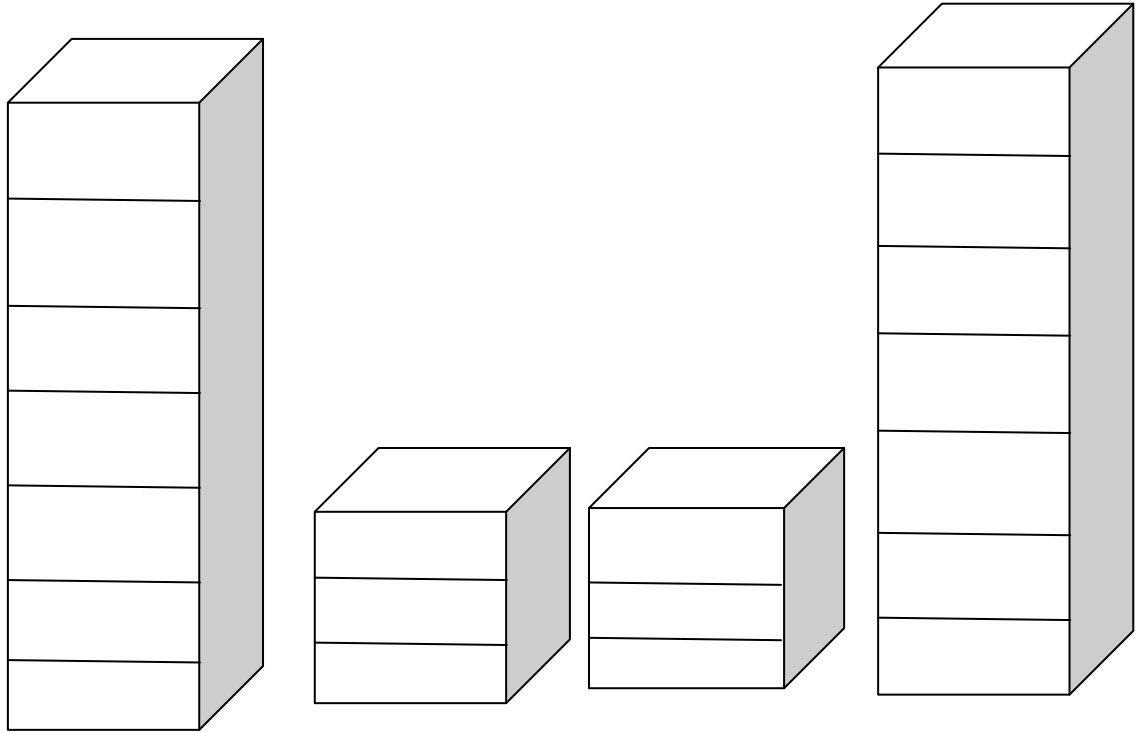
-> 4 bits (OCTET)

-> 10 bits restants (TAG)

même exo que Q4 Janvier 2008

Question 4 – Théorie (/4)

La description du protocole de communication Internet est souvent représenté graphiquement par le schéma suivant. Expliquez en quelques lignes la présence de ces 4 colonnes et leur taille différente.



Examen d'introduction à la microinformatique Solvay BA2

Mardi 02/09/2014

Nom :

Prénom :

Question 1 – Base de données relationnelles (/6) :

L'organisme international du contrôle et de la sécurité aérienne décide de réaliser une base de données relationnelle répertoriant toutes les catastrophes aériennes concernant un avion de ligne et survenues ces dernières années. Dans cette base de données doivent apparaître bien évidemment les catastrophes en question (avec date, nombre de victimes,...), le lieu précis où l'on a retrouvé les débris de l'avion, le type d'avion (Airbus A310, Boeing 747, ...) impliqué dans la catastrophe, l'avion lui-même (avec son nombre d'heures de vol, son âge), la compagnie aérienne possédant cet avion et responsable du vol. Il est aussi nécessaire d'y faire figurer les conditions météo au moment de l'accident (orage, temps serein,...), les pilotes aux commandes de l'avion (le commandant de bord et un ou plusieurs copilotes). Finalement, chaque compagnie aérienne est assujettie à plusieurs sociétés de maintenance chargées de l'entretien et de la supervision des avions de cette compagnie (les sociétés dépendent des types d'avion - une société donnée s'occupe de plusieurs types d'avion mais un type n'est pris en charge que par une société - et bien évidemment une même société pourra avoir plusieurs compagnies aériennes comme clients). On trouvera finalement dans la base de données le type de contrat (fréquence des entretiens, type de responsabilité, montant du contrat) qui lie la compagnie à chacune de ces sociétés d'entretien et la date du dernier entretien effectué pour l'avion à l'origine de la catastrophe.

Réalisez la base de données relationnelles permettant d'enregistrer et retrouver toutes ces informations en y indiquant les attributs qui la font fonctionner et ceux mentionnés dans l'énoncé.

Question 2 - Python (/6) :

Expliquez le rôle du code Python qui suit et ce qu'affichera à l'écran la dernière ligne de ce code.

```
class Les:
    def __init__(self):
        self.ns={}
        self.no = str(input("quoi ? "))
    def int(self):
        self.x=int(input("Combien ? "))
        i=0
        nn=""
        while (i < self.x):
            nn=str(input("qui ? "))
            self.ns[nn]=int(input("Combien ? "))
            i+=1
    def give(self):
        self.et = []
        ny = 0
        self.my = 0
        for x,y in self.ns.items():
            if y < 7:
                self.et.append(x)
            else:
                ny+=1
                self.my+=y
        self.my/=ny

c1={}
c2={}
x=int(input("Combien ? "))
i=0
while (i < x):
    l=Les()
    l.int()
    l.give()
    c1[l.no]=l.my
    c2[l.no]=l.et
    i+=1
print(c1, c2)
```

Question 3 (Théorie) : /2

Pourquoi les systèmes de cryptographie sont-ils passés d'une version dite symétrique à une version asymétrique ?

Question 4 (Théorie) : /2

Pourquoi le code d'une instruction élémentaire doit-il reprendre d'une manière ou d'une autre le mode d'adressage utilisé par cette instruction ?

Parce que les différents modes d'adressage sont juste des différentes interprétations de la même adresse du coup sans savoir le mode d'adressage on peut prendre la mauvaise interprétation et donc trouver une mauvaise adresse...

Question 5 (Théorie) : /2

Pour quelle raison la fréquence des « ratés » en mémoire cache doit-elle être idéalement de l'ordre d'un million de fois inférieure à celle des « ratés » en mémoire centrale ?

Car la vitesse d'exécution s'écroule complètement en mémoire centrale

Pg 111 livre de ref édition 2
chapitre mémoire > transfert de niveau

Question 6 (Théorie) : /2

Expliquez la signification des trois tables des pages représentées ci-dessous. A quoi correspondent les informations données dans les entrées de ces tables ?

0	2	oui
1	4	oui
2	3	oui
3	15	non
4	13	non
5	24	non

0	1	oui
1	5	oui

0	0	oui
1	6	oui
2	37	non
3	28	non

(chaque tableau correspond a un programme)

1ère colonne : adresse logique des pages des 3 programmes

2ème colonne : adresse physique de la page

3ème colonne : nous dit si la page est chargée dans la mémoire RAM ou pas (si pas, l'adresse physique correspond a son adresse dans la mémoire swap)