

vous offre ce qui lui a tant manqué :

LA PÊCHE AUX MOdULES AVEC CORRIGES

Conception : Eric « Rikou » Driutti
Dessins : Gilles « Gillou » Galaud



SOMMAIRE :

Un an à l'INT, ses partiels et un peu plus ...

INFORMATIQUE

IF11 de l'année 1996-1997

SUJET

Durée : 3 heures Date : 27/1/1997 Documents : ceux distribués en cours

Exercice 1 :

Soit M une matrice à deux dimensions (n lignes et p colonnes - n et p strictement positifs) de nombres entiers. L'accès à l'élément de ligne i et de colonne j se note M(i,j). La numérotation des lignes et des colonnes commence à 1. Ecrire l'algorithme (non récursif) d'une fonction prenant en entrée la matrice, n et p et qui calcule en sortie la valeur de la somme de tous ses éléments.

Exercice 2 :

L'objectif de l'algorithme suivant est de (catastrophe la documentation a été perdue !). On sait qu'il travaille sur une matrice M à deux dimensions (n et p) comme celle de l'exercice 1. Que fait cet algorithme ?

```
Travail (n, p, M)
Début
Lire L, H /* L et H sont supposés strictement positifs */
Lire I, J /* I et J sont supposés strictement positifs et
inférieur à n (pour I) et p (pour J) */
Si I + H - 1 > n ou J + L - 1 > p
Alors
Ecrire Erreur
Sinon
Lig <- I
Tant que Lig <= I + H - 1
Faire
Col <- J
Tant que Col <= J + L - 1
Faire
M (Lig, Col) <- 0
Col <- Col + 1
Fin_Tant_que
Lig <- Lig + 1
Fin_Tant_que
Finsi
Fin
```

Exercice 3 :

Ecrire l'algorithme d'une procédure, Eclatement, prenant en entrée un tableau T de k entiers et qui rend en sortie un tableau P, contenant les éléments strictement positifs de T et un tableau N contenant les éléments strictement négatifs de T. Les tableaux P et N ont déjà été initialisés avec la valeur 0 pour tous leurs éléments. On suppose les tailles de P et N compatibles avec ce traitement.
Ecrire l'algorithme Eclatement (T, k, P, N).

Exercice 4 :

A partir de la matrice définie dans l'exercice 1, écrire un algorithme récursif qui calcule la valeur de la somme de tous les éléments de la matrice et préciser les paramètres de l'appel initial de cet algorithme.
Rappel : vous ne devez pas utiliser de structure répétitive.

Exercice 5 :

Le programme en C, donné en annexe 1, génére à la compilation un avertissement (warning - qui en fait est une erreur conceptuelle importante). Pourquoi ?. Malgré cela le développeur passe outre et exécute ce programme. Qu'affiche ce programme lors de son exécution ?

Exercice 6 :

A partir du programme en C (qui est juste!), donné en annexe 2, écrire le code nécessaire en C pour réaliser la procédure Travail dont l'algorithme est défini dans l'exercice 2.

CORRIGE

Exercice 1

Entrée : n le nombre de lignes
p le nombre de colonnes
M la matrice
Sortie : Somme la valeur de la somme de tous les éléments de M
Calcul (n, p, M)
Début
Somme <- 0
Pour i=1 à n
faire
Pour j=1 à p
faire

```
Somme <- Somme + M(i,j)
Fin_Pour
Fin_Pour
retour Somme
Fin
```

Exercice 2

Cet algorithme remplit un rectangle de 0 dans une matrice M à partir d'une position I,J et ce sur H lignes et L colonnes.

Exercice 3

```
Eclatement (T, k, P, N)
Entrée : T : Tableau d'entiers
k : nombre d'éléments dans le tableau T pertinents
Entrée_Sortie P : Tableau dont tous les éléments ont pour valeur 0 en entrée et contenant tous les éléments strictement positifs de T en sortie
N : Tableau dont tous les éléments ont pour valeur 0 en entrée et contenant tous les éléments strictement négatifs de T en sortie
ind_P : indice de variation dans le tableau des éléments positifs
ind_N : indice de variation dans le tableau des éléments négatifs
i : indice de variation dans le tableau des éléments à examiner

Début
ind_P <- 1
ind_N <- 1
Pour i = 1 a k
Faire
Si T(i) > 0
Alors
P (ind_P) <- T(i)
ind_P <- ind_P + 1
Sinon
Si T(i) < 0
Alors
N (ind_N) <- T(i)
ind_N <- ind_N + 1
Finsi
Finsi
Fin_Pour
Fin
```

Exercice 4

```
Calcul (M, ligne, colonne, n, p)
```

```
Début
Si ligne >= n
Alors
Si colonne >= p
Alors
retour M (ligne, colonne)
Sinon
retour M (ligne, colonne) + Calcul (M, ligne, colonne + 1, n, p)
Finsi
Sinon
Si colonne >= p
Alors
retour M (ligne, colonne) + Calcul (M, ligne + 1, 1, n, p)
Sinon
retour M (ligne, colonne) + Calcul (M, ligne, colonne + 1, n, p)
Finsi
Finsi
Fin
```

Condition d'appel : Calcul (M, 1, 1, n, p)

IF11 DE L'ANEE 1997-1998

SUJET

Durée : 3 heures Date : 23/2/98 Documents : les documents mis à disposition dans le cadre du cours

Exercice 1 (sur 4)

Définitions :

Un point est défini comme un couple de coordonnées (abscisse, ordonnée). On supposera qu'une coordonnée est une valeur entière. Un segment est défini comme un couple de point (début, fin). On a toujours début différent de fin. Une figure est composée d'un nombre (toujours inférieur ou égal à 10) de segments tous différents. Un dessin est composé d'un nombre (toujours inférieur ou égal à 10) de figures.

A partir de ces définitions et des notations vues en cours, proposer une structure de données pour modéliser ces informations.

2) On appelle rectangle englobant d'une figure, le rectangle dont les côtés sont fournis

par les valeurs extrêmes (minimum et maximum) des points de cette figure, sachant que ses côtés sont parallèles aux axes du référentiel.
L'évaluation de l'intersection de deux figures peut être simplifiée par l'utilisation des rectangles englobants (il ne peut pas y avoir d'intersection si au minimum les deux rectangles englobants ne s'intersectent pas). Il est donc intéressant de conserver cette information pour chaque figure.
Proposer une modification de la structure de données définie en (1) pour prendre en compte cette nouvelle information et permettant de faciliter l'écriture de l'algorithme de la procédure nommée rectangleEnglobant, qui prend en entrée un dessin, appelé carte, et renvoie ce dessin augmentée des informations relatives aux rectangles englobants évalués pour chaque figure. On suppose les données entrées correctement et valides.

Exercice 2 (sur 6)

On rappelle que le système de numération des romains utilise un ensemble de symboles de base (I, V, X, L, C, D, M) pour respectivement (1, 5, 10, 50, 100, 500, 1000). Chaque symbole est associé à une valeur numérique.

Les nombres sont formés par associations des différents symboles, des plus importants aux plus faibles. Par exemple, 37 = XXXVII
Le principe obéit à la règle de soustraction/addition : tout chiffre dont la valeur est inférieure à celui qui le suit doit lui être retranché. Par exemple, 9 = IX
Il est impossible d'associer plus de trois symboles identiques (4 = IV et non IIII).
On suppose un tableau de 100 caractères appelé, nombre. Le nombre d'éléments pertinents dans ce tableau est noté longueur (£ 100). Par exemple, XIV -> T[1] = 'X', T [2] = 'I', T[3] = 'V'
Ecrire l'algorithme d'une procédure récursive évaluationValeur qui permette de calculer l'équivalent numérique d'un nombre romain et donner l'appel initial de cette procédure.
On suppose que ce nombre est compris entre 1 et 3999, que le tableau contient effectivement un nombre, qu'il est correctement formulé et que longueur est correct.

Exercice 3 (sur 3) REPONDRE SUR LA FEUILLE REMISE SEPAREMENT

Soit l'algorithme suivant :

```
procédure P (Entrée T : tableau [1..100] de Entier, E : Entier, Entrée-Sortie N : Entier)
si N > 0 alors
si E différent de T[N] alors
N <- N - 1
P (T, E, N)
fsi
fsi
fproc P
```

Que fait P ? (en 5 lignes maximum)

Soient T1 un tableau d'entier [1..100] entièrement rempli, E1 un entier. Comment doit on initialiser I lors de l'appel P (T1, E1, I) pour assurer un fonctionnement normal ?

Exercice 4 (sur 3)

Coder la procédure de l'exercice 3 en langage C. On demande aussi d'écrire les définitions (des données, le prototype de la fonction) et l'appel de la procédure qui doit être intégré dans le programme principal. On considère que le tableau T1 et E1 ont déjà été initialisés mais pas I.

Exercice 5 (sur 2) REPONDRE SUR LA FEUILLE REMISE SEPAREMENT

Donner un exemple de problème pour lequel il n'existe pas d'algorithme permettant de le résoudre (en 4 lignes maximum).

Exercice 6 (sur 2) REPONDRE SUR LA FEUILLE REMISE SEPAREMENT

Donnez les résultats affichés par le programme dont le code des procédures est le suivant. La fonction main est donnée sur la feuille remise séparement. Ce programme compile normalement et aucune erreur ne se produit à l'exécution (déroulement tout à fait normal).

```
void f1 (int a, int *b);
void f2 (int *b, int c);
void f3 (int *a, int c);
void f4 (int **b, int a);
void f1 (int a, int *b)
/*****************/
{
a = *b;
} /* f1 */
void f2 (int *b, int c)
/*****************/
{
*b = c;
} /* f2 */
void f3 (int *a, int c)
/*****************/
{
f4 (&a, c);
} /* f3 */
void f4 (int **b, int a)
```

```
/******************/
{
**b = a;
} /* f4 */
```

CORRIGE

Exercice 1 :

1) Constante N = 10
type Point = structure
abscisse : Entier,
ordonnée : Entier
fstruct
type Segment = structure
début : Point,
fin : Point
fstruct
type Figure = structure
nombre_segments : Entier,
traits : tableau [1..N] de Segment
fstruct
type Dessin = structure
nombre_figures : Entier,
représentation : tableau [1..N] de Figure
fstruct

ATTENTION : ne pas oublier de mettre une structure pour gérer une figure et un dessin. "Toujours inférieur ou égal à 10" ne veut pas dire forcément égal à 10. Il est donc important de conserver dans la structure le nombre de représentations pertinentes. Le choix qui consiste à écrire autant de champs que de figure possible ne tient pas la route (imaginez une carte routière avec 100 000 objets possibles...).

Modifications :

Il vaut mieux modifier la structuration des figures et des segments de façon à simplifier l'évaluation du minimum et du maximum des valeurs de début et fin de segment sous la forme d'une opération élémentaire puisque les informations traitées ont la même organisation (une coordonnée). Le problème se ramène donc à la répétition d'une opération élémentaire, le calcul d'un minimum (resp. d'un maximum) sur des coordonnées. Dans un premier temps il n'est pas nécessaire de chercher à optimiser en ré-utilisant les dernières coordonnées comme premières coordonnées du segment suivant car la construction peut ne pas être connexe (et de toute façon la réalité est bien plus complexe que ça puisque dans les figures peuvent apparaitre des trous...).

type Figure = structure
nombre_segments : Entier,
segments : structure
x_début : tableau [1..N] de Entier,
x_début : tableau [1..N] de Entier,
x_fin : tableau [1..N] de Entier,
y_fin : tableau [1..N] de Entier
fstruct
rectangle_englobant : structure
minimum : Point,
maximum : Point
fstruct
fstruct

Algorithme :

procédure rectangleEnglobant (Entrée-Sortie carte : Dessin)
Pour i = 1 à carte.nombre_figures
carte.représentation [i].rectangle_englobant.minimum.abscisse =
Min (Minimum (carte.représentation [i].segments.x_début,
carte.représentation[i].nombre_segments),
Minimum (carte.représentation [i].segments.x_fin,
carte.représentation[i].nombre_segments))
carte.représentation [i].rectangle_englobant.minimum.ordonnee =
Min (Minimum (carte.représentation [i].segments.y_début,
carte.représentation[i].nombre_segments),
Minimum (carte.représentation [i].segments.y_fin,
carte.représentation[i].nombre_segments))
carte.représentation [i].rectangle_englobant.maximum.abscisse =
Max (Maximum (carte.représentation [i].segments.y_début,
carte.représentation[i].nombre_segments),
Maximum (carte.représentation [i].segments.y_fin,
carte.représentation[i].nombre_segments))
carte.représentation [i].rectangle_englobant.maximum.ordonnee =
Max (Maximum (carte.représentation [i].segments.y_début,
carte.représentation[i].nombre_segments),
Maximum (carte.représentation [i].segments.y_fin,
carte.représentation[i].nombre_segments))
fpour i
fproc rectangleEnglobant

fonction Minimum (valeur : tableau [1..N] de Entier, nombre : Entier) : Entier
minimum = valeur [1];
pour i = 2 à nombre

```
si valeur[i] < minimum alors
minimum <- valeur [i]
fsi
fpour i
retourner minimum
ffct Minimum
fonction Maximum (valeur : tableau [1..N] de Entier, nombre : Entier ) : Entier
maximum = valeur [1];
pour i = 2 à nombre
si valeur[i] > maximum alors
maximum <- valeur [i]
fsi
fpour i
retourner maximum
ffct Maximum

fonction Max (valeur1 : Entier, valeur2 : Entier ) : Entier
si valeur1 > valeur2 alors
retourner valeur1
sinon
retourner valeur2
fsi
ffct Max

fonction Min (valeur1 : Entier, valeur2 : Entier ) : Entier
si valeur1 < valeur2 alors
retourner valeur1
sinon
retourner valeur2
fsi
ffct Min
```

Exercice 2 :

```
Constante N = 100
fonction convertion (symbole : Caractere) : Entier
cas symbole ou
'I' : retourner 1
'V' : retourner 5
'X' : retourner 10
'L' : retourner 50
'C' : retourner 100
'D' : retourner 500
'M' : retourner 1000
fcas
ffct convertion
```

```
procédure evaluationValeur ( Entrée nombre : Tableau [1..N] de Caractere,
longueur : Entier,
position : Entier,
Entrée-Sortie valeur : Entier)
si position < longueur alors
si superieur (nombre [position +1], nombre [position]) = VRAI alors
// ou alors convertion (nombre[position + 1]) > convertion (nombre [position])
valeur = valeur - convertion (nombre [position])
sinon
valeur = valeur + convertion (nombre [position])
fsi
evaluationValeur (nombre, longueur, position + 1, valeur)
sinon
valeur = valeur + convertion (nombre[position] )
fsi
fproc evaluationValeur
```

Appel initial : valeur = 0;
evaluationValeur (nombre, longueur, 1, valeur);

ATTENTION : valeur est un paramètre en Entrée-Sortie, vous ne pouvez donc pas mettre de constante dans l'appel initial c'est obligatoirement une variable initialisée à 0. Que vous preniez une approche par la conversion des chiffres faibles ou des chiffres forts en premier, il faut faire attention lors du test de valeur entre deux éléments consécutifs à ne pas dépasser du tableau (surtout si la longueur du chiffre à convertir est ... 1 par exemple : I).

Optionnel :

Ici deux solutions sont possibles : Soit une approche par types abstraits (symbole) et définition d'un opérateur prenant en entrée deux symboles et permettant de faire la comparaison de deux symboles (fonction superieur), Soit utiliser directement la convertion pour tester l'ordre de deux symboles et auquel cas la fonction superieur ne sert à rien.

```
fonction superieur (symbole1, symbole2) : Booleen // VRAI si symbole1 > symbole2
cas symbole1 ou 'I' : retourner FAUX
'V' : si symbole2 = 'V' ou symbole2 = 'X' ou symbole2 = 'L' ou
symbole2 = 'C' ou symbole2 = 'D' ou symbole2 = 'M' alors
retourner FAUX
sinon
retourner VRAI
fsi
'X' : si symbole2 = 'X' ou symbole2 = 'L' ou
symbole2 = 'C' ou symbole2 = 'D' ou symbole2 = 'M' alors
retourner FAUX
```

```
sinon
retourner VRAI
fsi
'L' : si symbole2 = 'L' ou
symbole2 = 'C' ou symbole2 = 'D' ou symbole2 = 'M' alors
retourner FAUX
sinon
retourner VRAI
fsi
'C' : si symbole2 = 'C' ou symbole2 = 'D' ou symbole2 = 'M' alors
retourner FAUX
sinon
retourner VRAI
fsi
'D' : si symbole2 = 'D' ou symbole2 = 'M' alors
retourner FAUX
sinon
retourner VRAI
fsi
'M' : si symbole2 = 'M' alors
retourner FAUX
sinon
retourner VRAI
fsi
fcas
ffct superieur
```

Exercice 3 :

P prend en entrée un tableau de 100 éléments, T, un élément à rechercher E et une borne définissant un (sous-)tableau, N. P provoque une erreur si N > 100. En sortie N vaut N s'il est initialement <= 0, l'indice maximal d'une occurrence de E dans le sous-tableau 1..N, 0 si E n'existe pas.

2) Tableau entièrement rempli => I = 100

Exercice 4 :

Première solution :

Construire un tableau qui prenne en compte un élément de plus et sacrifier l'élément d'indice 0. Le principal avantage est que les spécifications de l'algorithme sont respectées, le codage en est simplifé (et cette technique permet d'utiliser des recherches par sentinelles).

```
#define NBRE 100 /* nombre d'element max du tableau */
int T1[NBRE + 1]; /* tableau dans lequel on effectue une recherche */
/* attention 100 + 1 car la numerotation commence a 0 */
int E1; /* element recherche dans le tableau */
int I; /* indice de la valeur recherchée (dernière occurence) ou 0 si elle n'existe pas */
void P (int T[], int E, int *N);
void P (int T[], int E, int *N)
{
if (*N > 0)
{
if (E != T[*N])
{
*N = *N - 1;
P (T, E, N);
}
}
} /* P */
.....
I = NBRE;
P (T1, E1, &I);
```

Deuxième solution :

l'algorithme s'adapte aux contraintes du langage. En C les indices des tableaux commencent à 0 => les spécifications de l'algorithme ne sont plus respectées en matière de retour si l'élément n'existe pas (puisqu'il rend -1 au lieu de 0) => cette modification doit être répercutée dans tous les logiciels utilisant la procédure P ou par un ajout d'une addition lors de l'envoi du retour.

```
#define NBRE 100 /* nombre d'element max du tableau */
int T1[NBRE]; /* tableau dans lequel on effectue une recherche */
int E1; /* element recherche dans le tableau */
int I; /* indice de la valeur recherchée (dernière occurence) ou -1 si elle n'existe pas */
void P (int T[], int E, int *N);
void P (int T[], int E, int *N)
{
if (*N ³ 0)
{
if (E != T[*N])
{
*N = *N - 1;
P (T, E, N);
}
}
} /* P */
```

.....
I = NBRE - 1; /* attention 100 - 1 car la numérotation commence a 0 */
P (T1, E1, &I);

Exercice 5 :

le problème de l'arrêt du processeur.

Exercice 6 : (suivant les valeurs initiales que vous aviez)

I (5, 8, 10):
Après f1 V1 = 5 V2 = 8 V3 = 10
Après f2 V1 = 5 V2 = 8 V3 = 5
Après f3 V1 = 5 V2 = 8 V3 = 8

II (4, 1, 3):
Après f1 V1 = 4 V2 = 1 V3 = 3
Après f2 V1 = 4 V2 = 1 V3 = 4
Après f3 V1 = 4 V2 = 1 V3 = 1

III (6, 2, 8):
Après f1 V1 = 6 V2 = 2 V3 = 8
Après f2 V1 = 6 V2 = 2 V3 = 6
Après f3 V1 = 6 V2 = 2 V3 = 2

IV(7, 1, 2):
Après f1 V1 = 7 V2 = 1 V3 = 2
Après f2 V1 = 7 V2 = 1 V3 = 7
Après f3 V1 = 7 V2 = 1 V3 = 1

V(9, 3, 5):
Après f1 V1 = 9 V2 = 3 V3 = 5
Après f2 V1 = 9 V2 = 3 V3 = 9
Après f3 V1 = 9 V2 = 3 V3 = 3

Le tournoi Centrale Lyon



IF 12 DE L'ANNEE 1997-1998-09-27

SUJET

Durée : 1h30 heures Date : 27/3/98 Documents : les documents mis à disposition dans le cadre du cours.

Exercice 1 (sur 4)

Vous disposez d'un type abstrait Liste dont les spécifications sont telles que ce type gère une liste triée par ordre alphabétique croissant. Vous avez accès aux fonctions (dont la sémantique a été vue en cours) suivantes :

estListeVide : Liste -> Booleen
valeurElementListe : Liste -> Valeur
suiteListe : Liste -> Liste

En ne parcourant la liste qu'une seule fois, écrire l'algorithme d'une procédure, visualisationInverse, qui prend en entrée une liste et affiche à l'aide de la primitive écrire, les valeurs contenues dans la liste dans l'ordre alphabétique décroissant.

Exercice 2 (sur 4 et 8)

Supposons un système d'exploitation supportant au maximum 100 utilisateurs et gérant des processus à plusieurs niveaux de priorité (3 niveaux ordonnés du moins prioritaire au plus prioritaire : 1, 2 ou 3). Le processeur est unique. Il convient donc de l'attribuer aux différents processus en fonction des priorités. Un processus est caractérisé par un numéro (entier), une action à effectuer (une chaîne de caractères) et un propriétaire (nom d'un utilisateur - chaîne de caractères).
On suppose que les demandes d'un utilisateur donné doivent être traitées en séquence mais une nouvelle demande peut arriver avant que la précédente n'ait été traitée, elle est alors mise en attente.
Pour simplifier on considère que tout processus dont le traitement est commencé, est traité intégralement (il n'y a pas de notion de partage de temps entre les différents processus). Le niveau de priorité est fixé pour un utilisateur une fois pour toutes à son inscription dans le système. Les utilisateurs pour un niveau de priorité donné sont servis à tour de rôle dans l'ordre d'arrivée dans le système de la première demande et à raison de l'exécution d'une demande de processus par utilisateur.

1)Proposez une organisation des données permettant le fonctionnement du système d'exploitation que vous décrirez avec le formalisme de structuration des données vu en cours et à l'aide d'un schéma.

2) Ecrire un algorithme, demandeUtilisateur, qui permet à partir d'un numéro de processus, une action, le nom de l'utilisateur et les données modélisées en (1) d'introduire une nouvelle demande de processus. Vous disposez des fonctions vues en cours sur les différents types abstraits et d'une fonction polymorphique (équivalent du malloc en C) pour obtenir dynamiquement un emplacement mémoire typé dont la signature est :
obtenir : type -> mémoire

Vous rappellerez les signatures des fonctions que vous utilisez. Si vous voulez modifier l'organisation définie en (1) pour faciliter l'écriture de cet algorithme, vous devez clairement expliciter les modifications que vous introduisez.
On considère que les données sont correctement formulées (l'utilisateur existe, les numéros de processus sont uniques, le système est dans un état de fonctionnement normal correctement initialisé...)

Exercice 3 (sur 4)

On suppose que l'on dispose d'une image en noir et blanc modélisée par un tableau à deux dimension de booléen. La valeur VRAI (resp. FAUX) signifie que l'information modélisée par un couple de valeur (ligne, colonne) est de couleur noire (resp. blanche).

Constante TAILLE 100
type ImageType = tableau [1..TAILLE][1..TAILLE] de Booleen
On souhaite écrire un algorithme qui construise un quad-tree simplifié. Un quad-tree est une structure d'arbre définie de telle sorte que le carré modélisé par une feuille de

l'arbre est obligatoirement de couleur homogène (soit tout blanc soit tout noir). Les nœuds (non feuille) représentent des carrés non homogènes.
Vous disposer d'un type abstrait Arbre et d'une instance (déjà crée et initialisée à vide) appelée arbre. Cet arbre gére des carrés définis par la structure suivante :

type PointType = structure
ligne : entier,
colonne : entier
fstruct
type CarreType = structure
coinHautGauche : PointType,
coinBasDroite : PointType
fstruct

Vous disposez des primitives de manipulation de ce type abstrait :

estArbreVide : Arbre -> Booleen
AjouteElementArbre : Arbre x Valeur -> Arbre

Vous disposez d'une fonction homogène qui a la signature suivante :
homogene : CarreType x ImageType -> Booleen
et qui renvoie VRAI si le carré défini dans l'image est de couleur homogène et FAUX sinon.
Vous disposez d'une procédure découpage qui prend en entrée un carré et renvoie en sortie un tableau de quatre sous-carrés.
découpage : CarreType -> Tableau [1..4] de CarreType

Ecrire un algorithme, decomposition, qui prend entrée une image, un carré de référence et un arbre et renvoie en sortie l'arbre modifié représentant l'image sous la forme du quad-tree. L'arbre résultat sera construit en appliquant une division récursive de l'image en 4 carrés et ce pour chaque carré tant que ce carré n'est pas homogène.
On considère que l'image (de type ImageType) est correctement construite, l'arbre (de type Arbre) correctement crée, ... L'appel initial est :

CarreType intial;
initial.coinHautGauche.ligne = 1;
initial.coinHautGauche.colonne = 1;
initial.coinBasDroite.ligne = TAILLE;
initial.coinBasDroite.colonne = TAILLE;
decomposition (image, initial, arbre);

CORRIGE

Exercice 1

```
procédure visualisationInverse (Entrée L : Liste)
si estListeVide (L) = FAUX
alors
visualisationInverse (suiteListe (L))
ecrire (valeurElementListe (L))
fsi
fproc visualisationInverse
```

Commentaires :

A partir des signatures des fonctions valeurElementListe et suiteListe vous pouvez en conclure que la représentation choisie est une liste récursive (transparent 23).
L'algorithme de base de parcours est donné dans le transparent 24. Pour afficher dans le sens inverse il vous suffit de traiter l'élément après avoir parcouru la liste et non pas avant de faire le parcours. Vous devez donc inverser les deux appels dans la partie "alors".
Si vous n'avez pas saisi la récursion vous pouvez toujours utiliser une pile pour empiler les éléments à chaque traitement d'un élément de la liste et faire un affichage par dépilage une fois le parcours effectué, néanmoins cela revient à parcourir deux fois la "liste".

Exercice 2

1) Pour chaque utilisateur il faut tenir un niveau de priorité (constant)
-> à priori on ne connaît pas le nombre exact d'utilisateurs mais on sait au max 100

```
Constante NBRE_CLIENT 100
NBRE_PRIORITE 3
type PrioriteType = structure
utilisateur : chaine de caractère
niveauPriorite : entier
fstruct
type UtilisateursType = structure
nombreUtilisateurs : entier
caractéristiques : tableau [1..NBRE_CLIENT ] de PrioriteType
fstruct
```

Le système gère trois niveaux de priorité -> un tableau de liste de processus utilisateur (nombre indéfini) avec une entrée par niveau de priorité.

```
type GestionProcessusType = structure
demandes : tableau[1..NBRE_PRIORITE] de liste de DemandesUtilisateurType,
clients : UtilisateursType
fstruct
type DemandesUtilisateurType = structure

utilisateur : chaine de caractère
```

```
processus : File de ProcessusType
fstruct
type ProcessusType = structure
numéro : entier
action : chaine de caractère
fstruct

2) Pour faciliter la recherche de l'existence d'un client mais ce n'est pas obligatoire
GestionProcessusType = structure
clientActif : liste de chaine de caractère,
demandes : tableau[1..NBRE_PRIORITE] de liste de DemandesUtilisateurType,
clients : UtilisateursType
fstruct

procédure demandeUtilisateur ( numero : entier,
action : chaine de caractère,
utilisateur : chaine de caractère,
Info_systeme : GestionProcessusType)
Entier niveau;
ProcessusType processusUtilisateur;
DemandesUtilisateurType demandeUtilisateur
processusUtilisateur <- obtenir (ProcessusType);
processusUtilisateur.action = action;
processusUtilisateur.numero = numero;
// Recherche du niveau de priorité
niveau <- rechercheNiveau (Info_systeme, utilisateur);
si existeElementListe (Info_systeme.clientActif, utilisateur) = FAUX
alors
// l'utilisateur n'a pas déjà fait de demande de processus
ajouterElementListe (Info_système.clientActif, utilisateur);
// création de sa liste de processus
demandeUtilisateur <- obtenir (DemandesUtilisateurType);
demandeUtilisateur.utilisateur <- utilisateur;
demandeUtilisateur.processus <- nouvelleFile ();
ajouterElementFile (demandeUtilisateur.processus, processusUtilisateur);
ajouterElementListe (Info_systeme.demandes[niveau], demandeUtilisateur);
sinon
// l'utilisateur avait déjà fait une demande
ajouterElementFile
( rechercherFileUtilisateur ( Info_systeme.demandes[niveau],
utilisateur),
processusUtilisateur)
fsi
fproc demandeUtilisateur
fonction rechercheNiveau ( Info_systeme : GestionProcessusType,
nomRecherche : chaine de caractère) : entier
```

```
// on sait que l'utilisateur existe
indice <- 1
tant que Info_systeme.clients.caracteristiques [indice].utilisateur <> nomRecherche
faire
indice <- indice + 1
ftq
retourner Info_systeme.clients.caracteristiques [indice].niveauPriorite
ffct rechercheNiveau

fonction rechercherFileUtilisateur ( demande : Liste,
utilisateur : chaine de caractère) : File
// on recherche dans la liste des demandes celle concernant l'utilisateur
// on est sure qu'elle existe
si (valeurElementListe (demande).utilisateur = utilisateur
alors
retourner valeurElementListe (demande).processus
sinon
rechercherFileUtilisateur (suiteListe (demande))
fsi
ffct rechercherFileUtilisateur
```

Récapitulatif des fonctions utilisées :

existeElementListe : Liste x Valeur -> Booléen
ajouterElementListe : Liste x Valeur -> Liste
valeurElementListe : Liste -> Valeur
suiteListe : Liste -> Liste
nouvelleFile : -> File
ajouterElementFile : File x Valeur -> File

Commentaires :

La phrase du sujet "Le niveau de priorité est fixé ... système" vous conduit à définir un tableau (nombre max d'utilisateurs connu) de couple (nom, niveau de priorité). Dans le polycopié (p. 14), on vous explique que le partage de ressource (imprimante -> ici c'est le processeur) est assuré par la gestion d'une file. Dans le même paragraphe on vous explique que quand le nombre de priorité est fixé (ici 3) la file générale est représentée par un tableau de file -> ici un tableau de 3 éléments. Pour que la réponse ne soit pas intégralement dans le polycopié on introduit un séquencement par utilisateur "Les utilisateurs pour un niveau ... sont servis à tour de rôle ... à raison d'une demande par utilisateur". Ceci vous conduit donc à définir une liste d'utilisateurs demandant l'exécution d'un ou plusieurs processus.
En final on obtient un tableau de liste (d'utilisateurs) de file (de processus).
Pour le 2.2 : "Vous disposez des fonctions vues en cours sur les différents types abstraits" vous permet de vous affranchir de toute gestion des pointeurs et autres techniques de représentation physique des données pour ne vous concentrer que sur la gestion applicative des types. On vous demande de faire exactement la même chose

que dans l'exercice de gestion des piles (calculatrice) sauf que là vous travaillez sur des listes et des files.

Exercice 3

procédure decomposition (Entrée image ImageType, initial CarreType,
Entrée-Sortie arbre Arbre)

```
CarreType carre;
sousCarres : tableau [1..4] de CarreType
ajouterElementArbre (initial, arbre)
si homogene (initial, image) = FAUX
alors
// un sous- carré réduit à 1 est forcément homogène
découpage (initial, sousCarres)
pour indice <- 1 à 4
décomposition (image, sousCarres [indice], arbre)
fpour indice
fsi
fproc decomposition
```



RATRAPAGE IF11-IF12 SEPTEMBRE 1998

Examen Final Informatique IG1 - Algorithmie

Durée : 2 heures Date : 8/9/1998 Documents : les documents mis à disposition dans le cadre du cours.

Exercice :

Une entreprise dispose en autres de trois fichiers contenant des informations sur les ventes réalisées. Le premier fichier, Produits, contient le code du produit (codeProduit), une chaîne de caractère, le libellé du produit (libelléProduit), une chaîne de caractère, et son prix unitaire (prixProduit), un réel. Ce fichier est séquentiel et trié sur le code du produit. Le deuxième fichier, Ventes, contient le numéro du client (numéroClient), une chaîne de caractère, le numéro de la facture (numéroFacture), une chaîne de caractère, le code du produit (codeProduit) et la quantité livrée (quantitéProduit), un entier. Ce fichier est séquentiel et trié sur le numéro de facture. Il y a donc plusieurs enregistrements si une facture porte sur plusieurs produits. Le troisième fichier, Chrono, contient le numéro de facture (numéroFacture) et la date de la facture (dateFacture). Ce fichier est séquentiel et trié par date de facture. Vous disposez des différents types et primitives de manipulation vus en cours. On considère pour simplifier que les fichiers ne contiennent pas d'erreur et qu'ils sont cohérents entre eux (si un numéro de facture est utilisé dans le fichier Ventes, il existe forcément dans le fichier Chrono, ...).

1.(sur 5 points)

1.1)Proposez une modélisation pour gérer la date de la facture.

1.2)Proposez une modélisation pour gérer l'ensemble des données selon les spécifications établies.

2.(sur 15 points)

Ecrire l'algorithme d'une procédure, Description, qui a pour objectif d'afficher la liste des libellés des produits et le nombre d'exemplaires livrés pour un client donné dans un intervalle de temps donné (dateDébut, dateFin), défini avec la structure proposée au 1.1., à l'aide des trois fichiers dont les noms sont donnés.

CORRIGE

Question 1:

1.1)

Deux façons peuvent être utilisées pour modéliser une date :
La première consiste à utiliser un quantième sous la forme d'un numérique <année><jour dans l'année> ou sa version décodée <année><mois><jour>. Le 2 janvier 1998 est donc codé : 1998002 (version décodée 19980102).
L'avantage de cette méthode est de faire facilement des comparaisons entre dates puisque l'ordre numérique peut directement être utilisé. L'inconvénient de cette méthode est de nécessiter un traducteur pour le décodage d'une date (tant à la saisie qu'à l'affichage).
La deuxième consiste à utiliser une structure sous la forme de trois champs année, mois, jour.
L'avantage de cette méthode est sa facilité de saisie. L'inconvénient de cette méthode est la nécessité de disposer de fonctions de manipulation pour faire des comparaisons.
Nous choisirons de définir une date à l'aide d'un type Date dans le cadre de cet exercice sous la première forme mais la deuxième est acceptable sous la condition d'avoir stipuler qu'il existait des fonctions de manipulation de comparaison et d'avoir donné au minimum leur signature (fonctions que de toute façon vous devez utiliser dans le cadre de cet exercice).

1.2)

Définition des structures associées à un enregistrement de chaque fichier :

```
type Produit_type = structure
codeProduit : chaine,
libelléProduit : chaine,
prixProduit : réel
fstruct
type Vente_type = structure
numéroClient : chaine,
numéroFacture : chaine,
codeProduit : chaine,
quantitéProduit : entier
fstruct
type Chrono_type = structure
numéroFacture : chaine,
dateFacture : Date
fstruct
```

Question 2

Algorithme Description :
Cet algorithme prend en entrée le numéro d'un client, la date de début de la période considérée et la date de fin ainsi que les trois fichiers nécessaires au traitement. La sortie est un effet de bord pour l'affichage.
Le fichier des ventes étant classé par ordre de numéro de facture et rien n'étant précisé sur la sémantique du numéro de facture il faut donc parcourir tout le fichier Ventes à la

recherche des enregistrements (soit pour un même client disséminé dans le temps - et donc tout au long du fichier - soit pour une même facture mais sur plusieurs produits).

A priori on ne connait pas le nombre de produit qui seront commandés par un client. L'utilisation d'une structure dynamique est donc obligatoire (=> pas de tableau). Les informations requises sont le libellé et la quantité totale livrées pour ce libellé dans une période donnée (=> deux champs).

type infoRecherchée_type = structure

libellé : chaine,
quantité : entier
fstruct

Comme les produits sont répartis et ne sont pas liés à une commande précise, la somme sera construite de manière incrémentale, il faut donc être capable d'accéder plusieurs fois de suite (ou non) au même élément. La structure de liste sera donc utilisée (ici nous prendrons une définition de liste itérative (une alternative serait d'utiliser une liste triée sur le code produit).
Le principe retenu est le parcours du fichier Ventes. En cas de fichier vide la liste des information à afficher sera vide elle aussi et le message 'pas de produit pour ce client' sera affichée. Si le fichier n'est pas vide mais que ce client n'a pas fait de commande le fichier sera entièrement parcouru et à la fin la liste sera toujours vide. Si le client a fait des commandes mais que les dates ne correspondent pas la liste sera toujours vide. Si le client a fait au moins une commande elle sera prise en compte par la procédure gérerProduit qui va ajouter le libellé dans la liste des informations à afficher (c'est cette procédure qui prendra en compte la répétition de commande pour un même libellé). Ceci donne donc 4 cas de figure (fichier vide, pas commande du client, commande mais en dehors de l'intervalle de temps et commande dans l'intervalle de temps).

procédure Description (entrée : client : chaine,
dateDébut : Date,
dateFin : Date,
fichierProduits : chaine,
fichierVentes : chaine,
fichierChrono : chaine)
ventes : Fichier (Vente_type)
infoAffichées : Liste de infoRecherchée_type
arret : Booléen
enregistrement : Vente_type
date : Date
ventes <- ouvrir (fichierVentes, lecture)
infoAffichées <- nouvelleListe ()
arret <- FAUX
repeter

```
enregistrement <- lire (ventes)
si estEnFin (ventes) = FAUX
alors
si enregistrement.numéroClient = client
alors
date <- rechercheDate ( enregistrement.numéroFacture,
fichierChrono)
si date >= dateDébut et date <= dateFin
alors
gérerProduit ( rechercheLibellé ( enregistrement.codeProduit,
fichierProduits),
enregistrement.quantitéProduit,
infoAffichées
)
fsi
fsi
sinon
arret <- VRAI
fsi
jusqu'à arret = VRAI
fermer (ventes)
afficherListe (infoAffichées)
fproc Description
```

Ici on peut utiliser une fonction puisque les fichiers sont cohérents entre eux, il ne peut donc pas y avoir de cas où par erreur de saisie un numéro de facture n'existerait pas.
A priori on ne sait pas combien de fois on va effectuer l'opération de recherche -> structure repeter puisqu'il faut faire au moins une fois le traitement (car il y a au moins une vente sinon cette fonction ne serait jamais appellée).

```
fonction rechercheDate ( numéro : chaine, nom_chrono : chaine) : Date
chronos : Fichier (Chrono_type)
enregistrement : Chrono_type
date : Date
chronos <- ouvrir (nom_chrono, lecture)
arret <- FAUX
repeter
enregistrement <- lire (chronos)
si estEnFin (chronos) = FAUX
alors
si enregistrement.numéroFacture = numéro
alors
arret <- VRAI
date <- enregistrement.dateFacture
fsi
sinon
arret <- VRAI
```

fsi
jusqu'à arret = VRAI
fermer (chronos)
retourner (date)
ffonc rechercheDate

La fonction rechercheLibellé est similaire à la fonction rechercheDate sauf qu'elle opére sur un enregistrement Produit_type au lieu de Chrono_type.

fonction rechercheLibellé (numéro : chaine, nom_produit : chaine) : chaine
produits : Fichier (Produit_type)
enregistrement : Produit_type
libellé : chaine
produits <- ouvrir (nom_produit, lecture)
arret <- FAUX
repeter
enregistrement <- lire (produits)
si estEnFin (produits) = FAUX
alors
si enregistrement.codeProduit = numéro
alors
arret <- VRAI
libellé <- enregistrement.libelléProduit
fsi
sinon
arret <- VRAI
fsi
jusqu'à arret = VRAI
fermer (produits)
retourner (libellé)
ffonct rechercheLibellé
La procédure afficherListe doit prendre en compte le cas ou la liste est vide et dans ce cas afficher un message. A priori ce coup-ci on connait le nombre d'élément dans la liste (utilisation d'une structure itérative pour), on doit tous les afficher.
procédure afficherListe (entrée liste : liste de infoRecherchée_type)
element : infoRecherchée_type
position : entier
si estListeVide (liste) = VRAI
alors
afficher 'Pas de produit pour ce client'
sinon
pour position <- 1 à nbElementListe (liste)
element <- valeurElementListe (liste, position)
afficher element.libellé, element.quantité
fpour
fsi

fproc afficherListe

Pour la gestion d'un produit, il ne faut pas oublier de sommer les quantités si le produit a déjà été rencontré. En cas de première rencontre, il faut créer un nouvel élément dans la liste.

```
procédure gérerProduit ( entrée libellé : chaine, quantité : entier
entrée-sortie liste : Liste de infoRecherchée_type)
nvelElément : infoRecherchée_type
nbre, position : entier
arret : Booléen
nbre <- nbElémentListe (liste)
position <- 1
arret <- FAUX
tant que arret = FAUX
faire
si position > nbre
alors /* 1ère rencontre du libellé */
arret <- VRAI
nvelElément.libellé <- libellé
nvelElément.quantité <- quantité
enlister (liste, nvelElément, position)
sinon
nvelElément <- valeurElémentListe (liste, position)
si nvelElément.libellé = libellé
alors
/* le libellé a déjà été rencontré on modifie la quantité */
arret <- VRAI
nvelElément.quantité <- nvelElément.quantité + quantité
delister (liste, position)
enlister (liste, nvelElément, position)
sinon
/* on passe à l'élément suivant dans la liste */
position <- position + 1
fsi
fsi
ftq
fproc gérerProduit
```



Spaw, le déuss de l'info

te souhaite bonne chance.

AM11 RATTRAPAGE SEPTEMBRE 1998-09-27

Polycopiés de cours autorisés Coordonateur : M. SCULLER
Durée : 1 heure

Question I.

Le système décrit par le Schéma 1 est utilisé pour gérer en temps réel la retransmission des alarmes provenant de 3 dispositifs (X,Y et Z sur le Schéma)

Les 3 dispositifs X,Y ou Z signalent l'existence d'une alarme en activant la ligne d'interruption et en positionnant leur ligne d'alarme Ali au niveau logique 1. Les lignes d'alarme Alx, Aly et Alz sont reliées à un port d'E/S que le processeur pourra lire à l'adresse 0300H.
Le systeme à base de processeur de la famille 8086 utilise un contrôleur PIC8259 pour gérer les interruptions. Le registre vecteur d'IT du contrôleur d'IT est initialisé avec la valeur 90H.

I1.
Expliquez le fonctionnement du système lorsqu'un dispositif X,Y ou Z envoie une alarme.

I2.
Ecrire en Assembleur le programme résident qui permettra de gérer ces alarmes.

- Pour la partie qui gère les alarmes, il faut lire la valeur sur le port d'E/S 300H, tester les bits 0, 1 et 2
- Si bit2=1 alors Affichez Z tout en haut a droite de l'écran sinon rien

- Si bitl=1 alors Affichez Y a l'emplacement précédent Z sinon rien
- Si hitO=1 alors Affichez X a l'emplacement précédent Y sinon rien

En mode texte 25x80, le segment video est en B800H.



Amsterdam 1997

EC11

EC11 sans documents, durée : 1h30

SUJET

1. Le multiplicateur keynésien : définition , mode de calcul et intérêt pour la politique économique (4 points)

2. Qu'est-ce que les rendements d'échelle d'une firme ? Comment les mesure-t-on ? (2 points)

3. Décrivez la structure et montrez l'intérêt du TES (tableau entrée-sortie) de la Comptabilité Nationale Française. (4 points)

4. (6 points) Une entreprise produisant un bien x admet pour fonction de coût :
   C(x) = x^1/2 + x^3/2

   a) Calculez le coût marginal Cm et le coût moyen CM. Déterminez le minimum du coût moyen. Les représenter graphiquement.
   b) Déterminez la fonction d'offre O(p), p étant le prix du bien. Définir la courbe d'offre sur le graphique précédent.
   c) La fonction de demande du bien sur son marché s'écrit : D = 900/p^2. Sachant que l'offre totale sur ce marché est constitué de 100 entreprises dont les fonctions de coût sont toutes égales à C(x), déterminer le prix et la quantité d'équilibre du marché. Calculez le profit réalisé par chaque firme à l'équilibre. Commentaires ?

5. (4 points) Voici des chiffres tirés de la comptabilité nationale française (les prix et quantités sont exprimés en unités arbitraires ; les valeurs sont en milliards de Francs)

		1995	1996
Habillement	Prix Quantité Valeur	250 1 250	 0.996 250
Logement	Prix Quantité Valeur	1000 1 1000	 1.034 1060
Total des valeurs		1250	1310

a) Calculez l'évolution des prix de la consommation d'habillement de 1995 à 1996 (en %).
b) Calculez l'indice des prix de la consommation habillement et logement de 1996 (base 100 en 1995), selon Laspeyres.



Le gala 1997
(mais où est caché l'éléphant ? ?)

CORRIGE

1. Proposition 1

le multiplicateur Keynésien est le nombre pour lequel une somme investie par l'état est multipliée pour obtenir le montant de l'activité ainsi générée.
En économie fermée, l'idée est que l'activité de la nation est proportionnelle à l'investissement.

Démonstration : Si l'état investit ΔI dans la construction d'une route, le revenu R1 des employés est ΔI = R1. Les employés vont consommer cR1 et épargner S1 (R1 = cR1 + S1), cR1 devient le revenu R2 d'une autre classe sociale qui à son tour va consommer cR2 et épargner S2 (R2 = cR2 + S2), et ainsi de suite.
Au final, l'activité Y de la nation est Y = Σ (sur i) Ri
= Σ (i de 1 à l'infini) c ^ (i-1) * R1
= R1 * 1/1-c
d'où le facteur Keynésien : k = 1/1-c

Proposition 2

Le multiplicateur Keynésien provient de la théorie de Keynes et permet de calculer le surplus d'activité due à l'investissement de l'état, lorsque celui-ci dépense plus qu'il ne reçoit (déséquilibre budgétaire).

En effet, l'investissement de l'état va être injecté dans l'économie , à un premier stade, une partie de cet investissement sera épargnée et l'autre partie sera consommée, la partie consommée sera alors un revenu à un second stade et le même processus recommencera (partie épargnée, partie investie) et ainsi de suite.

En notant c la proportion marginale à consommer, un calcul simple (suite géométrique) montre que le multiplicateur keynésien est égal à k = 1 / 1- c .

Ce multiplicateur joue un rôle important dans le cas d'une politique de relance d'entreprise par un état , ainsi pour un investissement ΔI initial, le retour sur activité économique est de $\Delta I * k$. Néanmoins, cela oblige l'état à s'endetter, ce qui n'est pas sans poser des problèmes (effet boule de neige de la dette si trop d'emprunts ont été contractés).

En France, le multiplicateur keynésien est de 6.

Question 2

Le rendement d'échelle d'une entreprise est le nombre pour lequel la production d'une entreprise est multiplié quand on augmente le facteur de production.

Si on suppose que la production ne dépend que de L (Labour, travail) et K le capital, $Y = f (K, L)$

$F (\lambda K, \lambda L) = \lambda^k * f (K, L)$

Si k = 1, le rendement d'échelle est constant,

Si k < 1, le rendement d'échelle est décroissant (ce qui est peu intéressant car proportionnellement on produit moins que ce qu'on a injecté),

Si k > 1, le rendement d'échelle est croissant (ce qui est peu intéressant car proportionnellement on produit plus que ce qu'on a injecté).

Question 3

Le TES est un instrument chiffré permettant de connaître les échanges de biens entre les grands secteurs d'activité (agriculture, énergie, industriel, biens et services) à l'échelle de la nation.

Il reprend également les montants de la TVA, de la production, des importations des exportations, des emplois, des consommations finales, la formation brute de capital fixe, les variations de stocks et ce pour chacun des secteurs d'activités mentionnés ci-dessus.

Il montre également les échanges (achats et ventes) entre les secteurs d'activité. Ainsi, chaque branche est représenté par une colonne. L'ensemble des produits vendus par une branche est représentée par une ligne et ce pour chaque

branche. Ainsi en lisant le tableau ligne par ligne, on a les achats de chaque branche (colonne j) pour le produit i.

Et en lisant le tableau colonne par colonne, les achats de la branche (colonne j) en produits de chaque tranche (ligne i).

Le TES permet de calculer certains agrégats (PIB par exemple), il permet également de calculer la matrice des coefficients techniques qui sert de base de comparaison et de simulation.

Question 4

a)

Cm = dC/dx = 0.5 * x^-1/2 + 1.5 * x^1/2
CM = C/x = x^-1/2 + x^1/2

dCM/dx = 0 si et seulement si x = 1 donc CM est minimal pour x = 1.

b)

On doit résoudre Cm = p, ce qui revient à résoudre l'équation suivante :

$9x^2 + (6-4p^2)x + 1 = 0$

d'où

$x1 = (4p^2 - 6 + 4p\sqrt{p^2 - 3}) / 18$
$x2 = (4p^2 - 6 - 4p\sqrt{p^2 - 3}) / 18$

A un prix donné, on veut que l'offre soit la plus élevée donc on obtient O(p) = x2.

c)

Le prix est déterminé quand l'offre est égale à la demande Ototal(p) = D(p)
C'est à dire 100 O(p) = D(p) d'où p = 2.32.
Le profit réalisé pour chaque entreprise est donc p * x – C(x) = 0.42
(x = 1.66).

Question 5

a)

Sachant que le prix est de 250 en 1995 et que pour 0.996 en quantité on a une valeur égale à 250 d'où un prix égal à 250/0.996 = 251.

On a une évolution égale à (251 – 250) / 250 = 0.4 %

b)

On utilise la formule ip = ΣP1Q0 / ΣP0Q0.

Ainsi il vient (base 100 en 1995) pour l'habillement ip = 100.4

Pour le logement, on calcule le prix de 1996 : 1060 / 1.034 = 1025.15 soit un indice des prix égal à ip = 102.515.

Habillement ip = 100.4 Logement ip = 102.515



Le festival Litige

GF 1998

SUJET

(2 **points par question)**

1. Expliquez le modèle du Boston Consulting Group et son intérêt dans la démarche stratégique.

2. Quels sont les deux principaux objectifs du bilan social d'une entreprise ?

3. Que représente le marche potentiel d'une entreprise ? Comment le calcule-t-on ?

4. Citez, en les commentant, deux exemples de positionnement de produit dans une gamme.

5. Expliquez la différence existant entre la méthode de la valeur actualisée et celle du taux de rentabilité interne dans le calcul d'un investissement.

6. Quelles différences essentielles existe-t-il entre un amortissement et une provision ?

7. Comment calcule-t-on la valeur ajoutée d'une entreprise (solde intermédiaire de gestion) ?

8. Quelles différences fondamentales existe-t-il entre le bilan et le compte de résultats ?

9. Quelle différence existe-t-il entre une décentralisation et une déconcentration ?

10. Quelles sont les conditions de base nécessaires à la création d'une Société anonyme ?

CORRIGE

1.Le modèle du BCG prend en compte 4 types de produits :
- produits stars : fort taux de croissance et grand nombre de parts
- produits dilemme : fort taux de croissance et faible nombre de parts
- produits « vache à lait » : faible taux de croissance et grand nombre de parts
- produits poids mort : faible taux de croissance et petit nombre de parts.

Une entreprise doit normalement posséder ces quatre types de produits. L'intérêt de ce modèle dans la démarche stratégique est qu'on sait ce que sont les produits et ce qu'on doit en faire : les produits « vache à lait » financent les produits stars dont la gestion est lourde et qui peuvent perdre ce statut de stars à tout moment. Les produits dilemmes : soit on les vend car pas assez de parts, soit on en fait des produits stars. On essaie de minimiser les produits poids mort.
Ce modèle permet de clarifier la démarche à suivre.

2. Les objectifs du bilan social sont de connaître la situation sociale au cours de l'année dans l'entreprise (point de vue interne) mais aussi de connaître les efforts sociaux consentis par l'entreprise durant l'année pour améliorer la situation sociale extérieure (exemple : le respect de l'environnement). On juge l'impact de l'entreprise sur l'individu.

La situation sociale interne peut être illustrée par :
- le nombre d'embauches et de licenciements,
- le nombre de stagiaires,
- le nombre de jeunes et de femmes etc

3. Le marché potentiel de l'entreprise représente :
- le marché de la concurrence et le marché des non consommateurs relatifs
- le marché des non consommateurs relatifs représente les personnes qui n'achètent pas les produits de l'entreprise ni ceux de la concurrence et qui n'ont pas de raison de les acheter.
- Le marché potentiel représente le marché futur d'une entreprise si elle supplante la concurrence et réussit à persuader les non consommateurs relatifs.

4. Il existe des produits de complémentarité pour approfondir la gamme d'un produit

et des produits d'image pour véhiculer une image de l'entreprise. Par exemple, élaborer un produit très sophistiqué pour véhiculer l'image d'une société moderne voire dynamique.

5. Dans le calcul d'un investissement, on a vu :
- la méthode de la valeur actualisée :
  Valeur actualisée = - I0 + F1/1+i + F2/(1+i)^2 + ... + Fn / (1+i)^n
  I0 = investissement
  Fk = flux de liquidité à la kème année
  i = taux d'actualisation que les entreprises doivent prendre, en général, au moins égal au taux de dévaluation.
- la méthode de la rentabilité interne :

ce taux r est calculé de façon à ce que la valeur actualisée soit nulle, c'est à dire tel que :
I0 = F1/1+r + F2/ (1+r)^2 + ... + Fn / (1+r)^n

6. L'amortissement et la provision représentent tous les deux une perte de valeur.

La différence vient du fait que, dans le cas de l'amortissement, cette perte de valeur est inévitable (due à l'usure du temps) alors que pour les provisions cette perte de valeur n'est pas sûre et certaine (elle est dite potentielle).

7. La valeur ajoutée d'une entreprise représente sa création de richesse. Elle est calculée comme suit :

Valeur ajoutée = Ventes + Productions – (Achats marchandises + ΔStock marchandises + autres charges externes).

8. Le bilan et le résultat sont tous les deux des documents datés. Le bilan est la photographie à l'instant t de la situation économique et financière d'une entreprise. La différence vient du fait que dans un bilan, on ne repart jamais à zéro, le bilan garde une partie de l'histoire de l'entreprise alors que dans le compte de résultat, on inscrit tout ce que l'entreprise a réalisé dans l'année (produits, charges ...).

9. La décentralisation est géographique : c'est une délocalisation, c'est à dire un déménagement des structures de l'entreprise qui ne modifie pas la structure hiérarchique de l'entreprise.

La déconcentration , quant à elle, est une redistribution des pouvoirs au sein des structures de l'entreprise.
On peut faire de la déconcentration tout en restant au siège social de l'entreprise (sans déménager).

10. Pour créer une société anonyme, il faut :
- un groupe d'au moins 7 personnes physiques ou morales (c'est à dire qu'une entreprise peut en faire partie).
- Réunir un capital social d'au moins 250 000 Francs (il représente la garantie financière de l'entreprise).

Les personnes qui apportent les fonds sont des actionnaires : ils détiennent des actions.
L'assemblée générale des actionnaires se réunit une fois par an pour élire un conseil d'administration constitué de 3 à 12 personnes qui va désigner un président directeur général (PDG).
Si l'assemblée générale des actionnaires a plus de pouvoir, elle va élire (une fois par an) un conseil de surveillance constitué de 3 à 12 personnes qui va proposer un directoire (3 personnes) qui sera ensuite avalisé par l'assemblée générale des actionnaires.

RESEAU



RT11

SUJET

Sans documents Coordonnateur : C. Lac courbes de trafic fournies

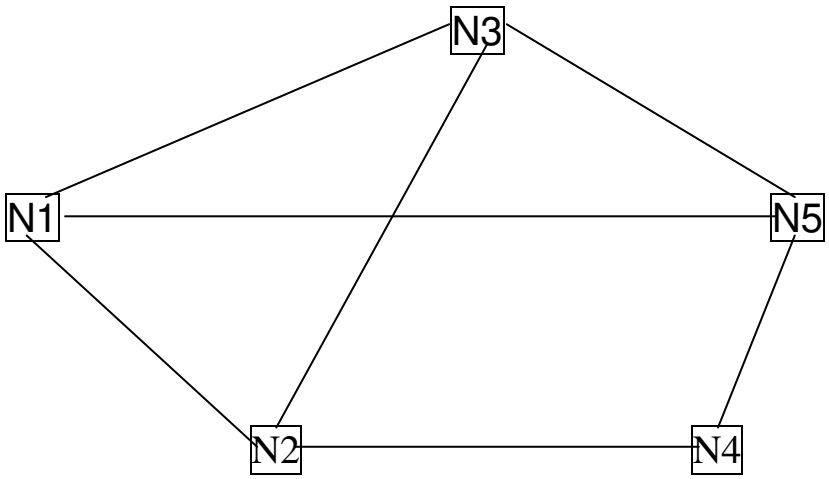
1ère Partie : (9,5 points)

Un réseau téléphonique est constitué de 5 nœuds, notés N1, N2, N3, N4, N5. La matrice des trafics offerts entre ces nœuds est la suivante (tous les chiffres exprimes en Erlangs).

De \ Vers	N1	N2	N3	N4	N5
N1	-	12	5	8	14
N2	8	-	9	11	9
N3	5	10	-	10	18
N4	11	9	14	-	10
N5	20	15	18	25	-

1- Les nœuds du réseau sont supposés totalement maillés (avec un faisceau supposé bidirectionnel entre chaque couple de nœuds). Le contrat de Qualité de Service impose un taux de rejet de 1%. Le réseau est supposé utiliser uniquement l'acheminement direct sans débordement. Expliciter la signification de ce choix. Dimensionner dans cette configuration quatre artères du réseau, au choix.

2- On suppose que la topologie imposée au réseau soit la suivante:



Dire ce que cette topologie impose en terme de fonctionnalités ajoutées dans les nœuds et proposer un acheminement pour tous les trafics (on suppose qu'il n'est pas fait usage du débordement).

· Dimensionner les faisceaux N1-NT2, N1-N5, N1-N3, N2-N4.

3- Le réseau a la configuration totalement maillé de la question 1. On se pose la question de vérifier les données de la matrice de trafic de la première question, en faisant une campagne de mesures sur le trafic échangé entre les nœuds N2 et N4. On compte pour cela le nombre de connexions établies, a plusieurs moments de la

semaine. Les durées moyennes de communication sont connues sans imprécision. On recueille les mesures suivantes:

Date et heure	Nombre de communications	Durée moyenne (min)
Lundi 9h-10h	355	4
Mardi 16h-16h30	178	3
Mardi 21h-22h	185	6

Les mesures sont-elles compatibles avec les chiffres de la matrice ? Evaluer et essayer d'cxpliquer .

2èmé partie: (7,5 points)

Une Ecole est reliée au réseau extérieur par l'intermédiaire d'un I'ABX. Pour dimensionner le nombre de points de sortie, une étude de trafic a donnée les résultats suivants: le trafic est estimé en Erlang.

	8h-9h	9h-10h	10h-11h	11h-12h	12h-13h	13h-14h	14h-15h	15h-16h	16h-17h	17h-18h	18h-1
Administration	20	40	30	40	20	30	40	35	40	20	
Foyer d'étudiants	40	20	5	20	20	30	20	10	10	40	
Enseignants	10	20	10	10	20	10	10	15	10	15	

La durée moyenne d'une communication est de 3 minutes.

1. On désire dimensionner ces points de sortie pour satisfaire le trafic à l'heure chargée.

· Déterminer l'heure la plus chargée.

· Quel est alors le trafic offert ?

· Quel doit être le nombre de points de sortie sachant que l'on se fixe une probabilité de perte $p \leq 0{,}05$?

2. Certains appels « arrivée » passent par le standard où une opératrice met 40 secondes pour aiguiller un appel vers le bon correspondant.

- calculer le nombre d'appels traités par le standard sachant que ce dernier traite 20% des appels du trafic « arrivée » et que le trafic « arrivée » constitue 60%du trafic total (on considérera toujours le trafic à l'heure chargée).
- Quel est le trafic offert au standard ?
- Quel nombre minimal de standardistes doit-on prévoir pour faire face à cette demande ?

- Quelle sera alors la probabilité d'attente et le temps d'attente pour un appel qui attend ?

CORRIGE



La nuit chaude de l'Essec : tout le monde ...

1ère partie

1. De N1 à N2 : T0 = 12+8 = 20 Erlangs. On relève alors sur les courbes de trafic N12 = 30
   De N1 à N3 : T0 = 5+5 = 10 Erlangs p = 1% On relève N13 = 18
   De N1 à N4 : T0 = 8+11 = 19 Erlangs p = 1% On relève N14 = 29
   De N1 à N5 : T0 = 14+20 = 34 Erlangs p = 1% On relève N15 = 46

Si le réseau utilise uniquement l'acheminement direct sans débordement, cela veut dire que les différents nœuds sont directement reliés entre eux et que l'acheminement entre N1 et N2, par exemple s'effectue sans passer par les nœuds intermédiaires pour le trafic rejeté.

2ème partie

1.
L'heure la plus chargée est celle correspondant à la tranche 9h-10h, le trafic offert est de 80 Erlangs, il faut donc au moins 85 organes.
Pour déterminer l'heure la plus chargée, on additionne les trafics dus à 'administration, au Foyer d'étudiants et aux enseignants. La tranche horaire la plus chargée se situe entre 9 heures et 10 heures.
Le trafic offert vaut T0 = 40+20+20=80 Erlangs
Pour trouver le nombre d'appels traités par le standard à l'heure chargée, on utilise la formule T0=λh.
Soit λ=T0/h avec T0=0.2*0.6*80=9.6 Erlangs
H=3/60 heures

On obtient λ = 192 appels par heure à l'heure chargée.

Le trafic offert au standard est obtenu à nouveau grâce à la formule A = λh'

A = 192*40/3600 = 2.1 E

Le nombre minimal de standardistes est donc égal à 3.
On relève alors E(2.1) = 50 %
La formule m = T / N – A donne m = 40 / 3-2.1 = 44s

2.
Il y a (60*20*80) / (100*100*3/60) = 192 appels traités par le standard entre 9h et 10h.
Le trafic offert au standard 192 * 40 / 3600 = 2.13 Erlangs.
Une opératrice peut traiter 3600 / 40 = 90 appels par heure, il faut donc au moins 3 standardistes.
La probabilité d'attente est 0.5 = 50%
Le temps moyen d'attente pour un appel qui attend est 40 / (3 – 2.13) = 46 secondes.
Le temps moyen pour un appel est 50 * 46 / 100 = 23 secondes.

3ème partie : (3 points) SUJET ET CORRIGE

Cocher la (ou les) bonnes réponses, s'il y en a : (les bonnes réponses, lorsqu'on les a trouvées, sont ***** entourées d'étoiles *****).

1. L'ATM est une technique de commutation utilisée dans :
(a) le Frame Relay
(b) NUMERIS
(c) Le B(ISDN)
(d) Le N(ISDN)
(e) Le G(ISDN)

2. Le paquet ATM porte le nom de :
(a) Nodule
(b) Trame
(c) ***** Cellule *****
(d) Alvéole

3. L'adresse du routage à l'interface NNI public ATM fait :
(a) 1 octet
(b) 2 octets
(c) ***** 4 octets *****
(d) 3 octets

4. Le paquet ATM a une taille de :
(a) 16 octets
(b) 2047 octets
(c) ***** 53 octets *****
(d) 64 octets

(e) autre

5. ATM signifie :
(a) Asynchronous Time Modulation
(b) Assisted Transfer Model
(c) Assisted Time Modulation
(d) ***** Asynchronous Transfer Mode *****

6. L'ATM utilise :
(a) le réseau téléphonique commuté upgradé
(b) un réseau spécifique
(c) le réseau TRANSPAC
(d) le réseau Frame Relay

7. L'ATM est une technique hybride entre :
(a) la commutation de trames et la commutation de paquets
(b) ***** la commutation de circuits et la commutation de paquets *****
(c) la commutation de paquets et la commutation Ethernet
(d) la commutation de messages et la commutation de paquets

8. L'ATM est utilisé pour transmettre :
(a) spécifiquement des données
(b) spécifiquement de la voie
(c) spécifiquement de l'image
(d) ***** si autre à préciser ***** (réponse : toute information numérisée)

9. Les débits de l'ATM sont :
(a) < à ceux d'Internet
(b) ***** > à ceux d'Internet *****
(c) + / - équivalents à ceux d'Internet



Le principe du rateau ...

RT12

SUJET

Avec documents
Code Bloc: RTI2
Duree: lh30
Coordonnateur: C. Lac

Exercice 1: (4 points) On désire multiplexer 6 voies BV transmettant des caractères de 8 bits avec les débits binaires 100, 100, 100, 200, 200 et 300 bits par seconde. En utilisant la technique de multiplexage temporel par caractère, un IT peut etre affecté, soit à la voie la plus rapide, soit à la voie la plus lente.

En considérant ces deux modes d'affectation des IT, donner la structure de la trame HV et le débit de la voie HV dans les cas d'une signalisation en bande et hors-bande.

Exercice 2: (4 points) Un code lineaire de longueur 6 et de dimension 3 est constitué à partir de la matrice P suivante:

0 1 1
1 0 1
1 1 0

1) Donner sa matrice génératrice.
2) Etablir les expressions permettant la génération des mots de code.
3) En déduire l'ensemble des mots de code ainsi générés.

Exercice 3: (4 points) Un réseau local en bus a un débit de 10 Mbits par seconde et mesure 800 mètres. La vitesse de propagation des signaux est de 200 mètres par microseconde. Chaque transmission des données est suivie par l'émission d'un accusé de réception de 32 bits. Sachant que la longueur des trames est de 256 bits, calculer:

1) le nombre de bits parcourant le bus à un instant donné;
2) le temps séparant 1'émission d'une trame et la réception de l'accusé entre les deux stations les plus éloignées;
3) le débit efficace du réseau en supposant que chaque trame contient 48 bits de service.

Exercice 4: (8 points) La Banque de Bretagne possède un réseau national de 67 agences interconnectées à la maison mère située à Rennes. Après avoir pris connaissance du document joint résumant les besoins des utilisateurs et donnant quelques éléments des besoins techniques:

I) dessiner l'architecture générale du réseau en se restreignant a quatre agences et le siège, et expliquer brièvement les services fournis par les différents composants (aux utilisateurs et aux entités avec lesquelles ils communiquent) et les protocoles supportés;
2) analyser les outils d'interconnexion nécessaires (description de leur architecture, des protocoles supportés et des services fournis);
3) proposer une architecture du réseau pour le support des futures applications bancaires en mode client-serveur - justifier votre choix.

Source : Réseaux et Télécoms Octobre 1994

EN DIX-HUIT MOIS, LA BANQUE DE BRETAGNE A REMPLACE UN RESEAU DE TERMINAUX BANCAIRES PAR UNE ARCHITECTURE D'INTERCONNEXION DE RESEAUX LOCAUX, INSTALLES DANS 67 AGENCES ET RELIES A L'INFORMATIQUE CENTRALE PAR X25. CETTE REALISATION A VALU A ALAIN OLMOS, RESPONSABLE DE LA PRODUCTION D'ETRE FINALISTE DU PRIX DU MANAGER TELECOM/RESEAUX 1994.

A la Banque de Bretagne, le suivi des comptes s'effectue en temps réel. Le personnel connait, apres n'importe quelle opération, la situation bancaire de chaque client. Lorsque l'un d'entre eux fait une demande de prêt ou de crédit, le guichetier peut recevoir sur son ecran une analyse de ses produits bancaires ou de ses comptes. Et ce, grâce à la modernisation complète de l'équipement informatique et réseaux des 67 agences, entreprise entre la fin de 1992 et le printemps 1994. Les dirigeants de la banque souhaitaient doter leurs agences d'une panoplie d'outils d'aide au développement commercial, car la situation locale n'incite guère à l'immobilisme. A Rennes, métropole bretonne en développement, la concurrence fait rage. Mais les banques régionales, bien implantées, et dont la Banque de Bretagne est l'archetype, tiennent la dragée haute à leurs rivales disposant d'un réseau national de succursales.

Dans cette optique, l'informatique a été complètement revue. L'antenne, datant de dix ans, composée de mini-ordinateurs Nixdorf et de leurs terminaux, n'était plus adaptée aux ambitions définies par l'état-major de la banque. "Les postes de travail doivent être vendeurs et personnalisés par métier. Il faut qu'une présentation rapide de la situation du client apparaisse à l'écran et permette une proposition de vente adaptée", commente Alain Olmos, responsable de la production de l'établissement. Chaque agence est aujourd'hui, équipée de micro-ordinateurs aux capacités et aux applications plus performantes. Ils sont reliés par un réseau local.

INTERAGIR AVEC L'ORDINATEUR CENTRAL

La nouvelle informatique devait surtout interagir avec l'ordinateur central de la banque, à Rennes. Cet aspect répondait à la fois aux exigences d'organisation et de sécurité propres à la banque et à la volonté de préserver l'existant. Toutes les

applications bancaires stratégiques sont, en effet hébergées sur cet ordinateur compatible IBM. La rénovation de l'équipement informatique des agences a conduit a la modernisation du réseau informatique reliant les agences au site central (cf schémas).
En février 1993, la banque a lancé un appel d'offres. Onze sociétés ont répondu. L'intégrateur Axilan a finalement été retenu. Outre la fourniture de matériels et de logiciels, basés sur les logiciels réseau Netware de Novell, cette societé a installé les produits Novell, formé l'équipe informatique et câblé les locaux des agences.
La solution technique d'Axilan repose sur Netware 3.12, comme serveur de réseau local, et sur la passerelle Netware pour SAA de Novell, pour les liaisons entre ceux-ci et le site central. Un logiciel d'emulation 3270 assure l'accès aux applications hébergées sur l'ordinateur hôte. Le câblage Ethernet des 67 agences s'est déroulé entre septembre 1993 et mai 1994. De taille moyenne, et proche de Rennes, l'agence de Fougères a été choisie pour les tests, en Novembre 1993, un mois avant le déploiement de l 'opération. De janvier à mai 1994, 67 sites - de Rennes à Brest et de Nantes à Saint-Malo - ont été équipés, au rythme de quatre agences par semaine.Cette "architecture à base de serveurs de réseau décentralisés indépendants, mais reliés à l'ordinateur hôte, est plus sûre. En cas de panne d'un serveur, les autres continuent de travailler", conmente Célestin Chevalier, responsable réseau à la Banque de Bretagne.
En matière de telecommunications, Transpac relié tous les réseaux locaux au site central de Rennes. Les agences disposent d'accès directs à 9600 bps (contre 2400) et pour les plus petites, d'accès via Numeris par le canal D des accès de base. La banque bénéficie, pour cette infrastructure, de l'offre de réseau X25 privé de Transpac. En bons banquiers, les responsables ont tenu un bilan financier précis du nouveau réseau. Son coût global (materiels, logiciels, prestations externes et câblage) a été évalué à huit millions de francs, alors qu'à elle seule la redevance annuelle de maintenance de l'ancien réseau Nixdorf revenait à 2,5 MF. Le retour sur investissement est prévu sur 36/40 mois. Le réseau privé X25 proposé par Transpac a été négocié pour trois ans sur la base d'un forfait annuel. Ce principe exclut toute taxation au volume et a la durée (sauf pour les accès via RNIS); la Banque de Bretagne économise ainsi jusqu’à 600 000 F par an.

LA GESTION DE RESEAUX CENTRALISEE

Les micro-ordinateurs offrent un potentiel qui sera utilisé progressivement. "Aujourd'hui, les postes des agences travaillent en interaction avec les applications centrales en mode émulation 3270, commente Célestin Chevalier. Aucune application locale n'est utilisée, mais cela fait partie des évolutions prévues."
Certaines applications ont été adaptées ou réécrites, au niveau de l'ordinateur central, en fonction des nouveaux modes d'accès au réseau. En tête des services a venir figurent la possibilité de lancer, depuis le poste de travail, des impressions de données stockées sur le site central et des processus d'accès
aux applications simplifiés. Le réseau s'améliore. Les temps de réponse sont plus perforrnants : environ 2 à 3 secondes : l'accroissement du débit des accès Transpac de 2400 a 96OO bps y a contribué.

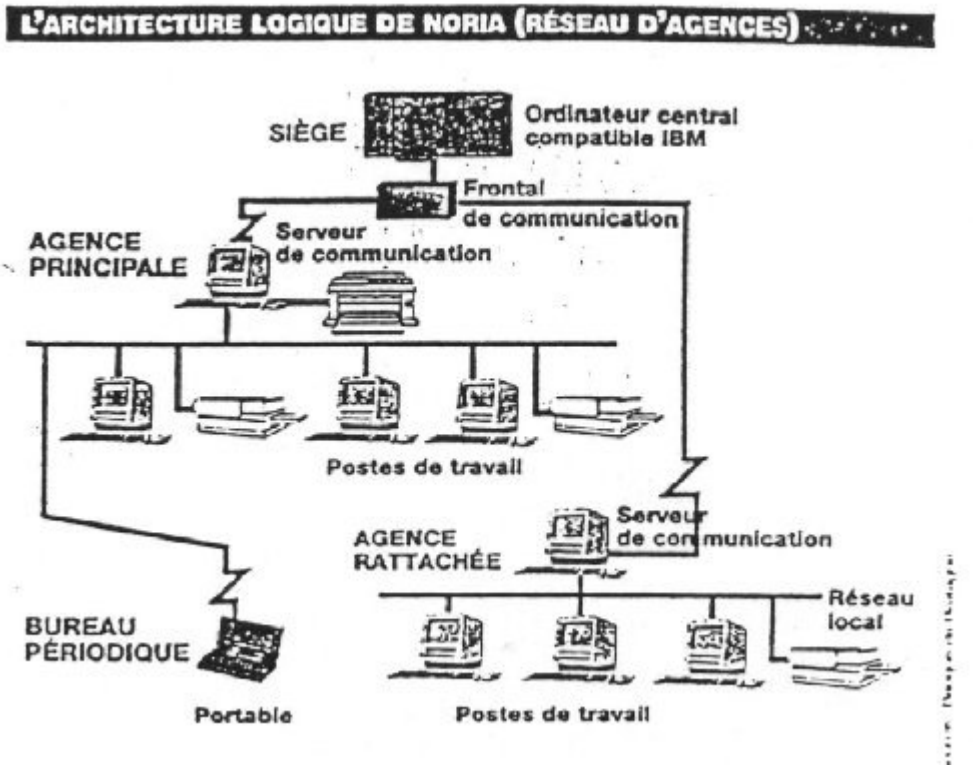
La banque a opté pour une administration de réseaux centralisée. Une console graphique H.-P. Openview sur PC surveille le réseau avec le protocole SNMP *(Simple Netwoork Management Protocol)* et centralise les alertes des équipements (concentrateurs, serveurs ou imprimantes). "Nous sommes dotés de logiciels de prise de controle à distance, nous permettant d 'intervenir dès qu'un incident nous est signalé", souligne Celestin Chevalier. Une cellule d'assistance assure notamment la gestion des appels téléphoniques des utilisateurs.

La premiére étape du déploiement de Noria étant achevée, d'autres evolutions sont envisagées comme le support d'applications bancaires en mode client-serveur. Pour ce faire, la Banque de Bretagne s'est dotée de Navigator de Novell, un outil de téléchargement de logiciels depuis le site central vers les postes des agences.

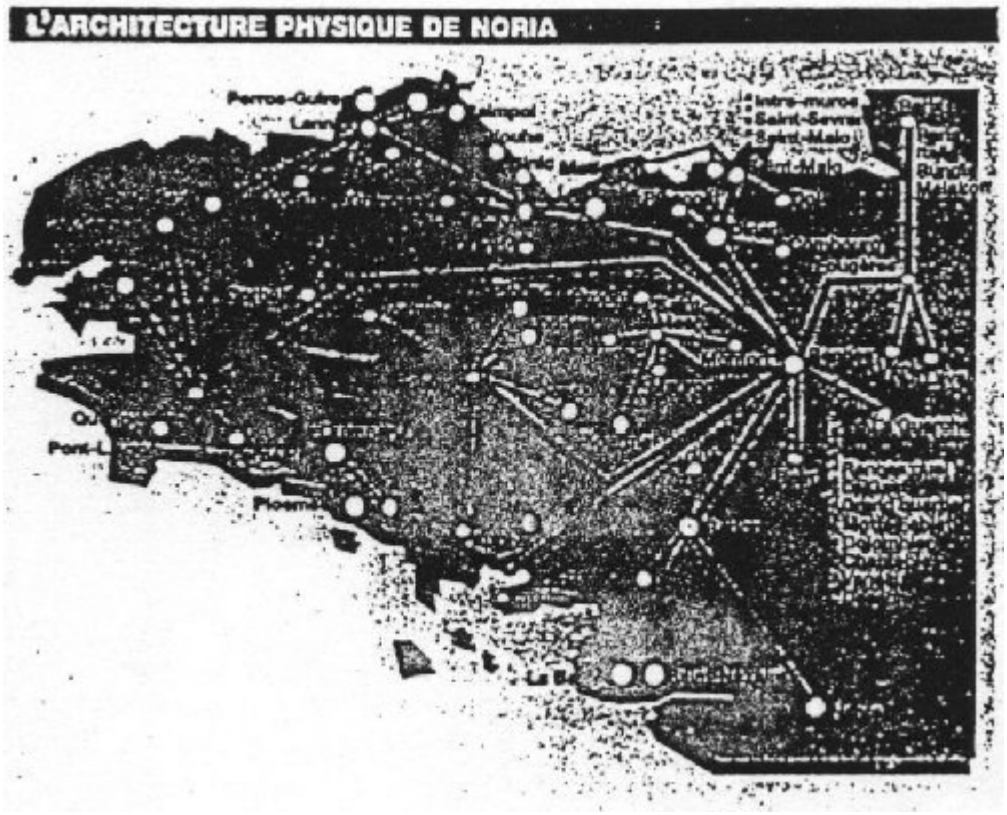
Le réseau informatique en chiffres

300 micro-ordinateurs Compaq
Prolinea 486/25 *sous* Windows 3.1
et MS-Dos 6.0, et une émulation 32270 (Attachmate). Les cartes d'interface Ethernet sont de type
Alliied Telesis AT 15O0.
67 serveurs de réseaux locaux Compaq Deskpro 486/33 et 486/66 sous Netware 3.12 (Netware pour
SAA et Host Print de Novell) et des cartes X25.
67 réseaux Ethernet 10Base-2 et 10 Bases-T (paire de type téléphonique) avec concentrateur Fibermux.
200 imprimantes (Oki et Hewlett Packard).
53 accès directs 9 600 bps à Transpac côté agences + des concentrateurs CLE 300 pour les guichets automatiques bancaires (conversion SDLC-X25).
12 accès Numéris canal D à Transpac pour les petites agences.
4 accès X25 à 19,2 Kbps sur le site central.
1 ordinateur hôte Hitachi Data Système HDS EX 80 (80 Go sur disque) compatible IBM avec un processeur frontal de communication 3745 d'IBM.

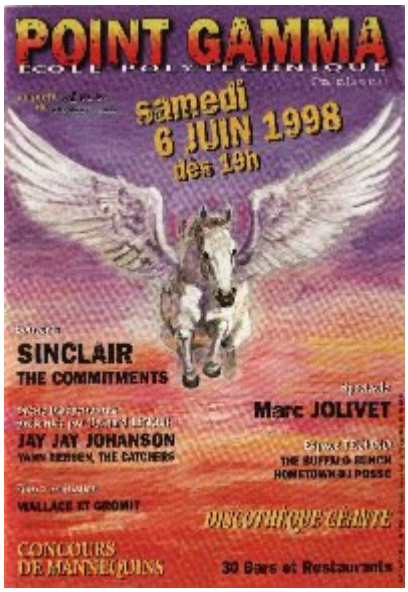
(On ne voyait rien sur les dessins ...)



Equipés de logiciels Netware pour SAA, de Novell, les serveurs de réseaux locaux distants accédent aux applications du site central.



La Banque de Bretagne utilise l'offre de réseau privatif X25 de Transpac pour relier ses agences, 67 serveurs sont ainsi reliés à l'ordinateur central de Rennes.



CORRIGE

Pour ce module, où tout le monde s'est pris des tôles, la correction proposée provient du croisement de plusieurs copies ayant obtenues le maximum des points pour chaque question ... néanmoins ça paraît incompréhensible ! Désolé, bon courage !

Exercice 1.

Affectation selon la voie la plus lente d = 100 bits/s.
10 IT : voie 1 : 1 IT, voie 2 : 1 IT, voie 3 : 1IT, voie 4 : 2 IT, voie 5 : 2 IT, voie 6 : 3 IT + 1 IT de verrouillage de trame.

Signalisation en bande : 1 IT de signalisation.

V-1-2-3-4-5-6-7-8-9-10-S : 12 IT

D = 8 * 12 * d = 1 200 bits/s.
Efficacité = somme des di / D = 0.83.

Signalisation hors bande : pas d'IT de dignalisation : 11 IT soit D = 9 * 1 100 bits/s, e = 0.91.

Affectation selon la voie la plus rapide : d = 300 bits/s.
On prend 1 IT par voie + 1 IT de verouillage de trame

Signalisation hors bande : 1 IT.

V-1-2-3-4-5-6-5 : 1 trame = 8 IT

D = 8 * 8 * d = 2 400 bits/s, e = 0.41
Signalisation en bande : 1 trame = 7 IT, D = 9 * 7 * d = 2 100 bits/s , e = 0.47.

Exercice 2.

$$H = \begin{matrix} 1\,1\,1\,0\,1\,0\,0 \\ 0\,1\,1\,1\,0\,1\,0 \\ 1\,1\,0\,1\,0\,0\,1 \end{matrix}$$

n = 7, r = 3 donc m = n – r = 4

1. La longueur du code est 7 et sa dimension est m = 4 et son rendement est R=m/n = 0.57.

La matrice génératrice est :

$$G = \begin{matrix} 1\,0\,0\,0\,1\,0\,1 \\ 0\,1\,0\,0\,1\,1\,1 \\ 0\,0\,0\,1\,0\,1\,1 \end{matrix}$$

2. On remarque que

$$G = \begin{matrix} x^6 + x^2 + 1 \\ x^5 + x^2 + x + 1 \\ x^4 + x^2 + 2 \\ x^3 + x + 1 \end{matrix} = (x^3 + x + 1) * \begin{matrix} x^3 + x + 1 \\ 1 + x^2 \\ x \\ 1 \end{matrix}$$

donc le polynôme générateur est x^3 + x + 1, c'est un code cyclique.

A partir de la matrice H, on a x5 = x1 + x2 +x3
x6 = x2 + x3 + x4
x7 = x1 + x2 + x4

3. Les mots de code sont :

0 0 0 0 0 0 0	1 0 0 0 1 0 1
0 0 0 1 0 1 1	1 0 0 1 1 1 0
0 0 1 0 1 1 0	1 0 1 0 0 1 1
0 0 1 1 1 0 1	1 0 1 1 0 0 0
0 1 0 0 1 1 1	1 1 0 0 0 1 0
0 1 0 1 1 0 0	1 1 0 1 0 0 1
0 1 1 0 0 0 1	1 1 1 0 1 0 0
0 1 1 1 0 1 0	1 1 1 1 1 1 1

Exercice 3

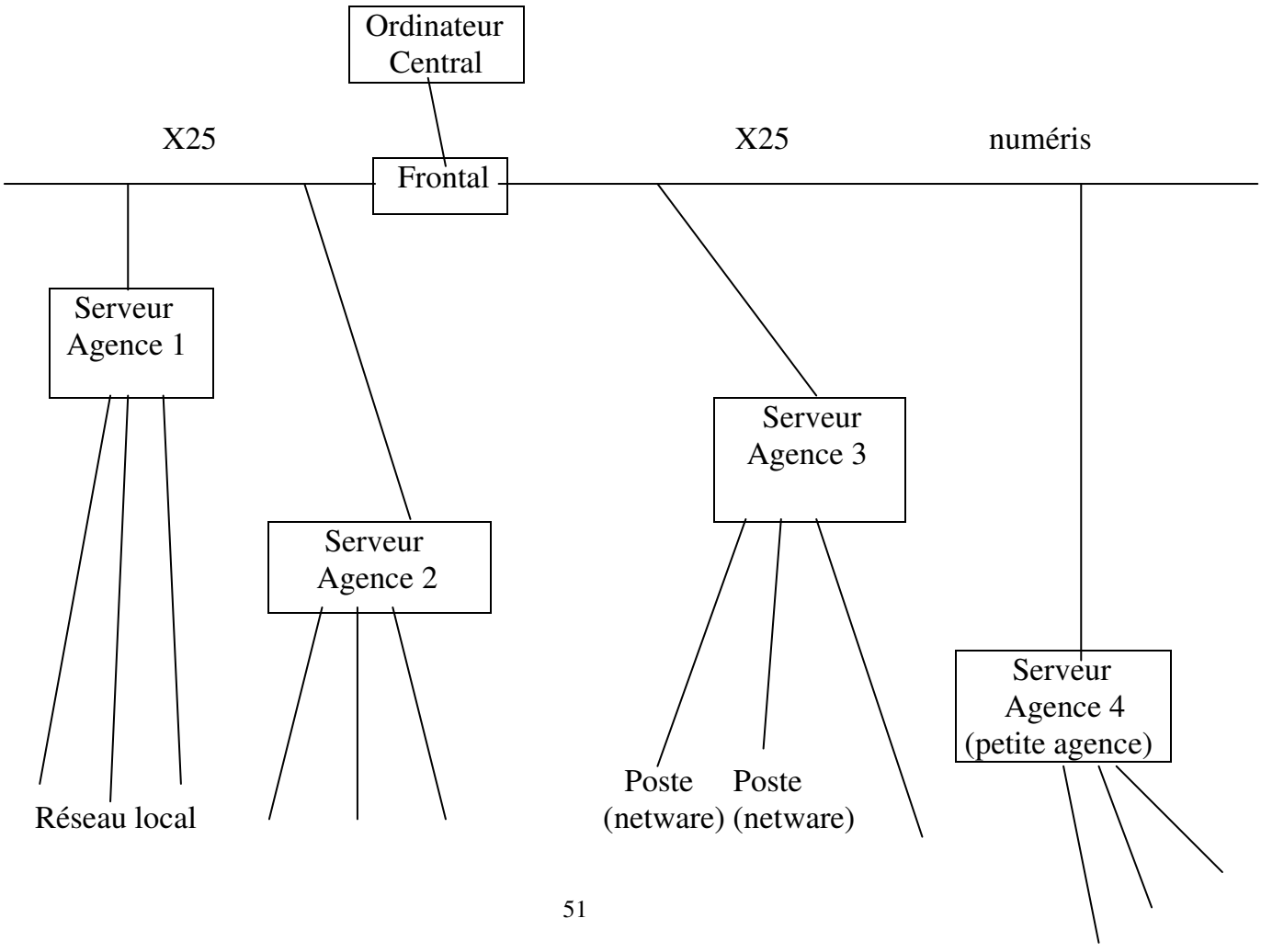
1. Au maximum 800 m de long, une propagation à 200 m/micro seconde donc un bit reste au plus 4 micro secondes dans le bus. Le débit est de 10^7 bits/s par conséquent il y aura 40 bits dans le bus.

2. Chaque trame (transmission de données) suivie par 32 bits d'accusé de réception.
Durée d'émission de la trame : 256 bits à 10Mbits/s donc 25.6 ùicrosecondes.
Temps de transit des données : 4 microsecondes
Emission de l'accusé : 32 bits et donc 3.2 microsecondes.
Temps de transit de l'accusé : 4 microsecondes
D'où une durée totale de 36.8 microsecondes.

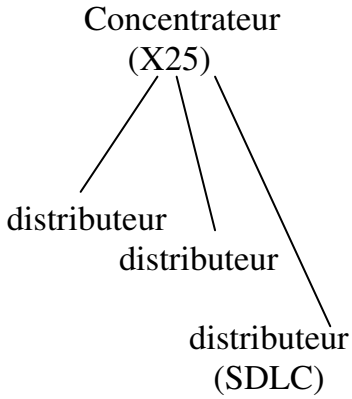
3. Débit efficace (256 – 48) bits de données en 36.8 microsecondes donc débit efficace de 5.65 Mbits/s.

Exercice 4.

1.



+ netware



Les différents postes sont reliés à l'ordinateur central où sont stockées toutes les données et les programmes (applications bancaires). Il n'y a pas encore de communications directes entre les agences (pas encore de client/serveur) d'où un passage par l'ordinateur central, la supervision du réseau à distance à partir de l'ordinateur central (administration distante).

2.

serveurs -> liens entre X25 et netware, accès au réseau Transpac
frontal -> gère le réseau (X25)
concentrateurs -> pour les billeteries (distributeurs) automatiques d'où le passage automatisé de SDLC en X25 avec concentration/ressources.

3.

Si possible, recablage en anneau avec sécurisation (cablage double) + mécanisme à jeton non adressé (dynamique déterministe).

RT14



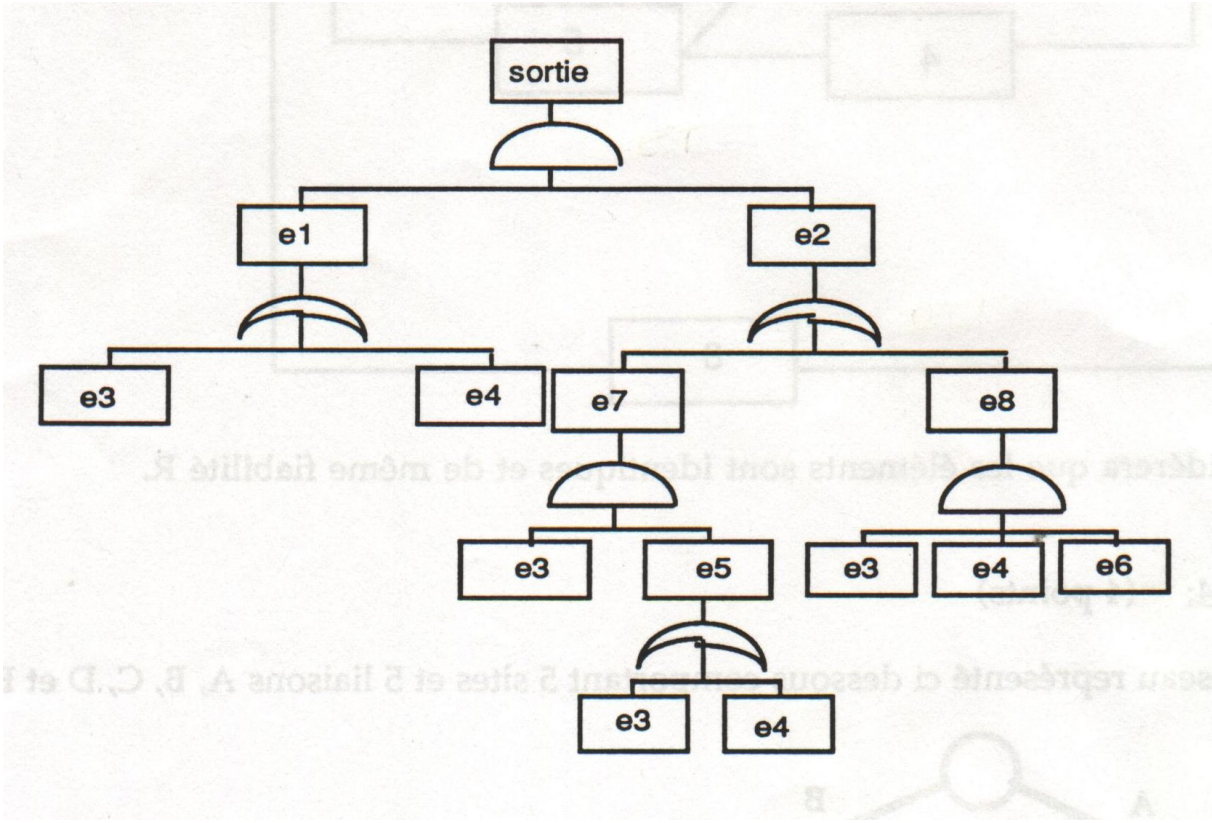
Le gala 1997

SUJET

Module : Fiabilisation et sécurisation des réseaux Code Bloc : RT14
Avec Documents
Coordonateur : C. Lac
Durée : 1h30

Exercice 1. (4 points)

A partir de l'arbre de défaillance suivant, développer l'expression booléenne de l'événement sortie.
Réduire le résultat qfin d'en déduire la (les) coupe(s) minimale(s).
Retrouver ce résultat par l'application de l'algorithme MOCUS.

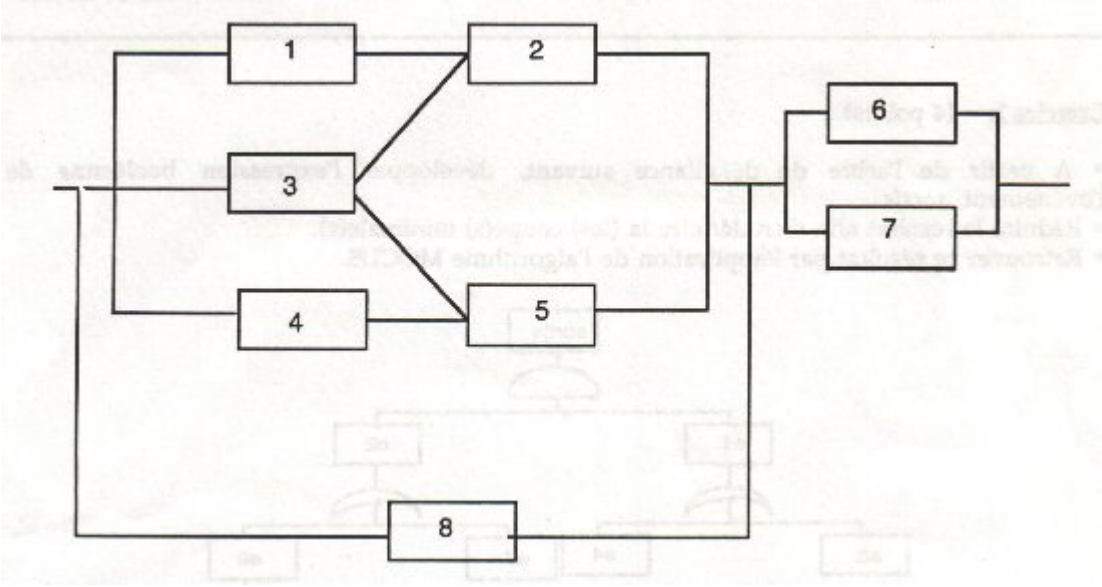


Exercice 2. (8 points)

Considérons un matériel auquel est adjoint un matériel de secours identique au premier, qui n'est mis en fonctionnement que lorsque le dispositif en service tombe en panne, sachant que le remplacement du matériel en service (lorsqu'il est défaillant) par le matériel de secours ne s'effectue pas instantanément : c'est le cas d'un échange standard permettant la remise en service d'une fonction à partir d'un matériel disponible en stock.
Supposons que la durée de remplacement suive une loi exponentielle de moyenne θ1

(taux de remplacement = ε) et que la durée de réparation d'un matériel suive une loi exponentielle de moyenne θ2 (taux de réparation = μ). Il y a un seul réparateur chargé du remplacement et de la réparation. Le taux de défaillance est constant et égal à λ. Calculer la disponibilité du système, le taux de défaillance et la probabilité ed manquer de pièces de rechange.

Exercice 3. (4 points)

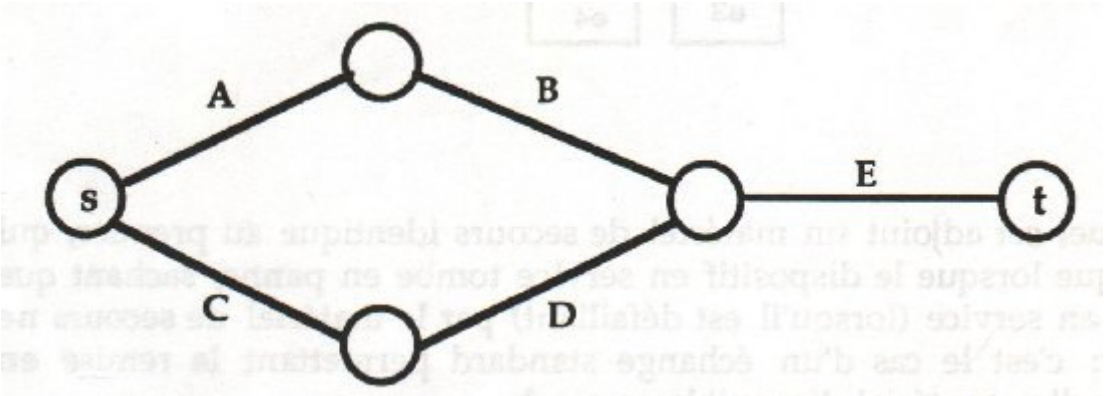


Calculer la fiabilité du système suivant :

On considérera que les éléments sont identiques et de même fiabilité R.

Exercice 4. (4 points)

Soit le réseau représenté ci dessous comportant 5 sites et 5 liaisons A, B, C, D et E :



En déduire les 8 états de succès (sans faire l'énumération totale des états) en précisant les éléments de votre raisonnement.
Donner l'expression de la fiabilité polynomiale entre s et t.
Calculer cette fiabilité sachant que la probabilité de fonctionnement d'une liaison est de 0.8.



ELECTRONIQUE
SN11

SUJET

Module: EN II
Durée: 1,5 heures
Documents autorises
Date: 27 Février 1 998
Coordonnateur: J-C. LAFONT

I. LOGIQUE SEQUENTIELLE: 6 points

Réaliser un compteur synchrone avec des bascules D capable de réaliser le cycle suivant: 0; 4; 6; 3; S; 2; 1; 0 ;... (La bascule de poids le plus faible est A).

Donnez les équations Di de chaque bascule et le câblage du compteur avec le minimum de porte logique.

II.GRAFCET:8 points

Un ingénieur débutant vient de réaliser le GRAFCET correspondant au cahier des charges du Distributeur de Timbres, donne ci après. Ne maîtrisant pas bien le GRAFCET, il s'adresse à un ingénieur INT (vous même) pour l'aider dans cette tache.

II. I. Vérifier si le GRAFCET présenté (voir pages suivantes) correspond bien au cahier des charges du distributeur de timbres.

Corriger les éventuelles erreurs (uniquement sur les actions et les réceptivités, mais sans modifier ou simplifier le GRAFCET: aucune suppression ou rajout d'étapes n'est autorise) sur papier (feuille jointe) en précisant:

- pour chaque transition ti à modifier, les équations des réceptivités corrigées,

- pour chaque étape a modifier, les équations des actions corrigées.

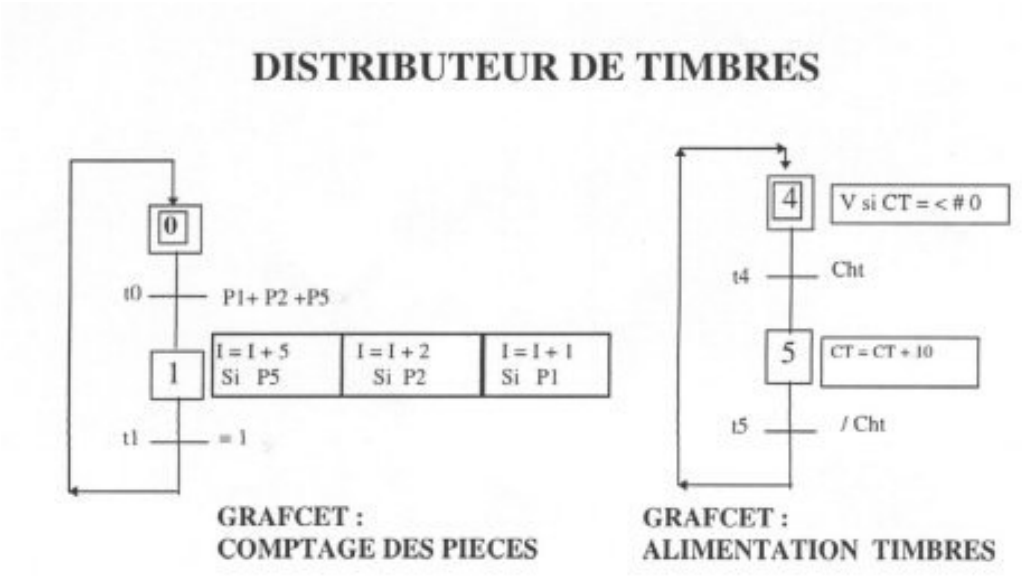
II.2. Donner les équations littérales des actions obtenues après correction: L; LPI; LD .

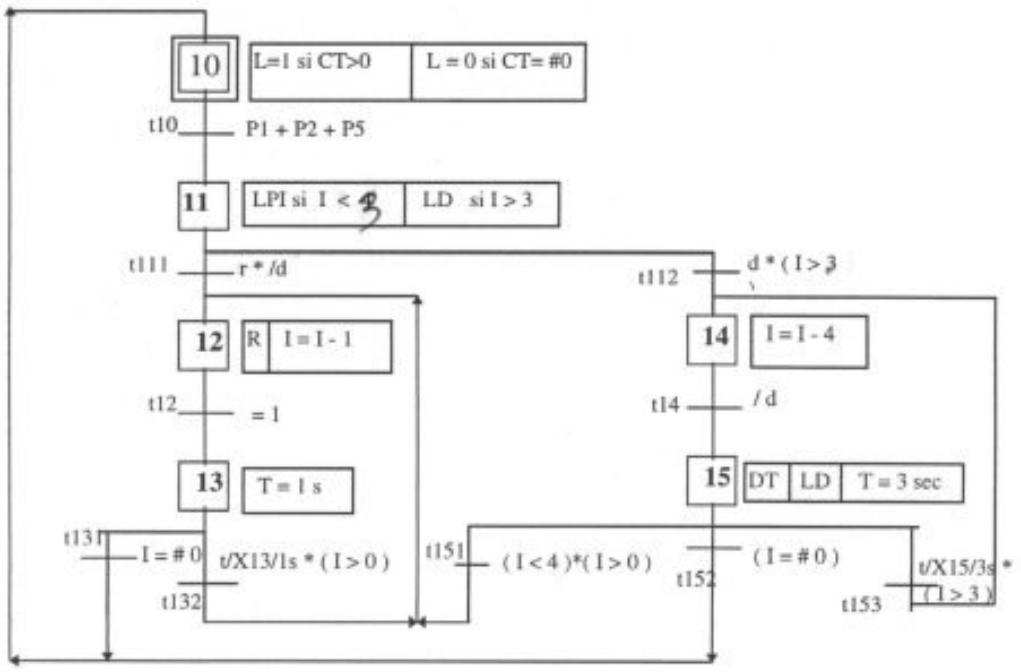
II.3 Donner les équations des bascules D (bascules D sans Reset) permettant de gérer le GRAFCET de comptage des pièces (les actions et le logigramme ne sont pas à réaliser).

III.MICROCONTROLEUR:6 points

Le programme principal a stocké, dans la zone de banque 3, 16 bits de données (dl5 à d0) dans les registres R7 (d15 à d8) et dans le registre R6 (d7 a d0). Mais seules les données allant de d15 à d4 sont significatives. II fait appel au sous programme COMP pour les comparer à des valeurs (ai) stockées dans la mémoire programme à l'adresse tab et tab+1. Mais les valeurs (ai) stockées se présentent sous la forme [0 0 0 0 a15 al4 al3 al2] pour l'adresse tab et [a11 a10 a9 a8 a7 a6 a5 a4] pour 1'adresse suivante.

Donner l'organigramme et le sous programme COMP (commenté) capable de faire la comparaison des données di (dl5 à d4) et ai (al5 à a4) et qui positionne le bit Pl.0 à 1 si les données di et ai sont identiques et à 0 dans le cas contraire.





GRAFCET : DISTRIBUTION DES TIMBRES et REMBOURSEMENT

DISTRIBUTEUR DE TIMBRES

Donner le ou les GRAFCET permettant de gérer un distributeur de timbres qui peut fournir des timbres a 3 F en introduisant des pièces de 2F, 1F ou de 5 F (p2, p1, p5).

Un système de détection fournit un niveau logique "1", sur la variable correspondant à la pièce introduite (pi), pendant le temps de passage de celle-ci dans l'orifice d'introduction des pièces.

Une lampe L de disponibilité est allumée s'il y a au moins un timbre dans la machine. S'il n'y a plus de timbres la lampe L est éteinte et un commutateur V est positionné à 1 pour aiguiller directement les pièces qui seraient introduites, alors que l'appareil indique qu'il est hors service (L=0), vers la zone de remboursement.

S'il y a des timbres, une lampe LPI s'allume, dès que le paiement est commencé, si le paiement est insuffisant et s'éteint dès que le paiement est suffisant.

Tant que la demande de distribution (bouton d = 1) n'est pas faite par le client, les pièces introduites peuvent être restituées, en appuyant sur un bouton r (r= I).

La restitution s'effectue en pièces de 1F. La commande de la trappe de distribution de pièces de 1F (TR) doit rester présente 1 seconde puis disparaître au moins 1 seconde pour chaque pièce rendue. Lorsque l'opération de remboursement est terminée le système est alors réinitialisé.

Si le paiement est suffisant, une lampe LD s'allume et la distribution (DT) peut être commandée après que le client ait appuyé (d = 1) et relâché (d = 0) le bouton d. La trappe de distribution du timbre DT doit être alors activée pendant 2 secondes puis désactivée au moins une seconde avant une nouvelle distribution .

Quand la distribution est terminée, la lampe LD s'éteint, le surplus est restitué (dans les mêmes conditions que pour une demande de remboursement) et le système est réinitialisé.

Un agent de la société a accès à la réserve de timbres et il peut rajouter 20 timbres a chaque enclenchement de CHT..

REMARQUE: On considérera, pour simplifier, que le stock de pièces de 1F n'est jamais épuisé.



Le voyage à Barcelone

CORRIGE

I. LOGIQUE SEQUENTIELLE :

Compteur synchrone avec des bascules D, déroulant le cycle : 0 ; 4 ; 6 ; 3 ; 5 ; 2 ; 1 ; 0 ...

N	C	B	A	C+1=DC	B+1=DB	A+1=DA
0	0	0	0	1	0	0
4	1	0	0	1	1	0
6	1	1	0	0	1	1
3	0	1	1	1	0	1
5	1	0	1	0	1	0
2	0	1	0	0	0	1

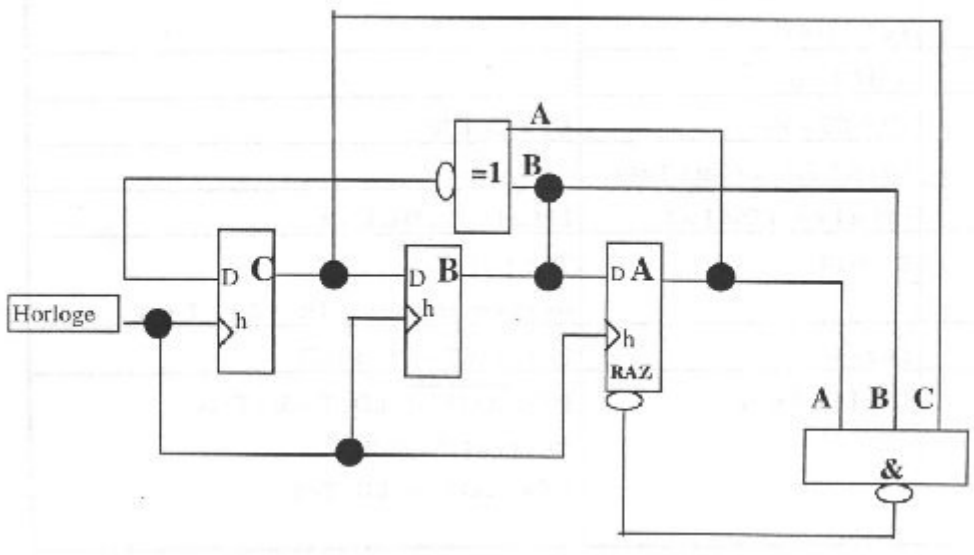
1	0	0	1	0	0	0

On obtient en utilisant la case optionnelle 7 :
DA = B
DB = C
DC = /A*/B + A*B -> A XNOR B (A coïncidence B)

Ces équations étant obtenues il faut étudier le comportement du compteur dans le cas où l'état 7, qui n'appartient pas au cycle, apparaîtrait.

N	C	B	A	DC=C+1	DB=B+1	DA=A+1
7	1	1	1	1	1	1
7	1	1	1			

Conclusion : Le compteur reste bloqué avec ses équations.
On constate que cette valeur bloque le compteur dans cet état. Aussi ce compteur doit être modifié (par le mode synchrone en modifiant une équation ou par une RAZ asynchrone).
Une solution est de mettre sur les entrées d'un NAND les sorties A, B, C des bascules et câbler la sortie de la porte NAND sur la RAZ d'une bascule. Ainsi si la configuration 7 apparaît, elle s'auto-détruira pour retrouver une configuration du cycle.



Question II.1 : Correction du GRAFCET.

Transitions	Réceptivités	Correction éventuelles
0	P1+P2+P5	
1	1	/P5*/P2*/P1
4	Cht	
5	/Cht	
10	P1+P2+P5	OK ou souhaitable : (I>0)*(CT>0) ou (P1+P2+P3)*(CT>0)
111	R*/d	
112	d * (I > 3)	D*(I>2)
12	1	T/x12/1s
131	I = #0	OK mais souhaitable (I=#0)*(t/x13/1s)
132	(t/x13/1s) * (I > 0)	
14	/d	OK Ou /d*(t/x14/1s) si une tempo sup de 1s en X14
151	(I < 4) * (I > 1)	(t/x15/3s)*(I>#0)*[(I<3)+(CT=#0)
152	I = #0	(I=#0)*(t/x13/3s) ou (t/x15/2s) si tempo de 2s en X15
153	T/x15/3s * (I > 3)	(t/x15/2s)*(I>2)*(CT>0) si tempo de 2s en X15

ETAPES	**Actions associées**	Corrections éventuelles
0		
1	I=I+5 si P5 ; I=I+2 si P2 ; I=I+1 si P1	
4	V si CT>0	
5	CT=CT+10	CT=CT+20
10	L=1 si CT>0 ; L=0 si CT=#0	
11	LPI si I<4 ; LD si I>3	LPI si I<3 ; LD si I>2
12	R ; I=I-1	TR ; I=I-1 ; ou suivant réceptivité : TR ; I=I-1 ; T=1s
14	I=I-4	I=I-3 ; CT=CT-1 ; LD
15	DT ; LD ; T=3s	DT si non (t/x15/2s) ; LD ; T=3s ; T=2s

II.2. Donner les équations littérales des actions obtenues après correction : L ; LPI ; LD

Set de L = X10 * (CT=#0) ; Action mémorisée -> Mise à 1 de L

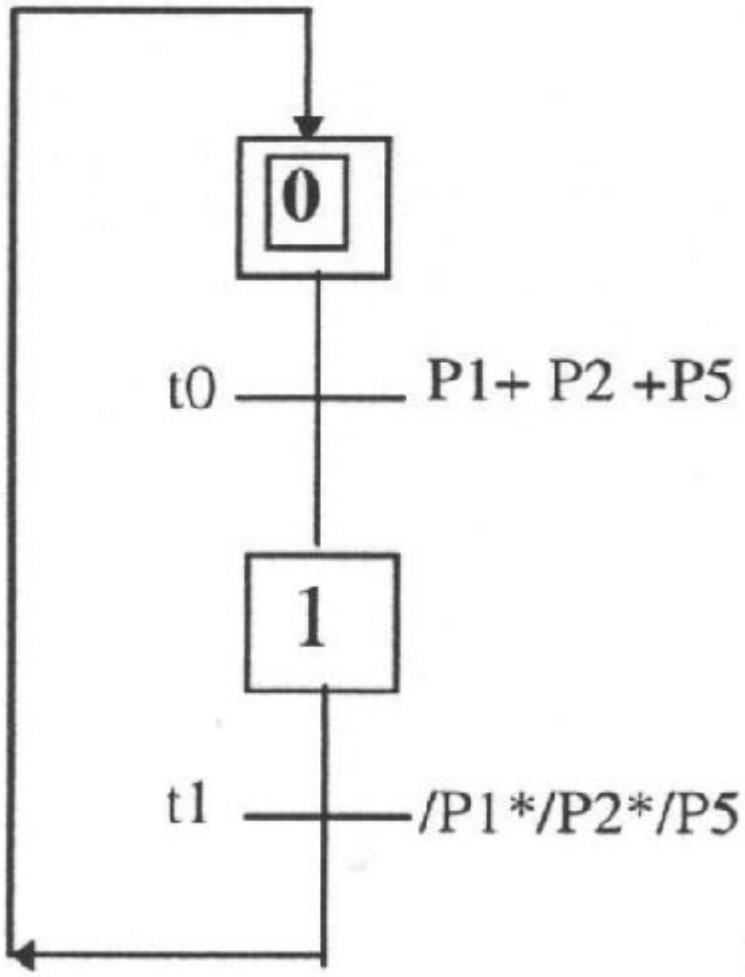
Reset de L=X10 * (CT=#0) ; Mise à 0 de L

LD=[X11 * (I>2)] + X14 + X15 ; Action maintenue de type répétée dans plusieurs étapes

LPI=X11 * (I<3) ; Action conditionnelle maintenue pendant la durée de l'étape

II.3 Donner les équations des bascules D (bascules D sans Reset) permettant de gérer le Grafcet de comptage des pièces (les actions et le logigramme ne sont pas à réaliser)

GRAFCET : COMPTAGE DES PIECES



PROGRAMMATION D'UN GRAFCET

Phase 1 : Initialisation X0 = 1 ; X1 = 0
Phase 2 : Evolution du GRAFCET

ai = Activation de Xi ; di = Désactivation de Xi

a0 = X1 * (/P1*/P2*/P5) ; d0 = X0 * (P1+P2+P5)
a1 = X0 * (P1+P2+P5) ; d1 = X1 * (/P1*/P2*/P5)

PHASE 3 : Actions
Cette phase n'étant pas à réaliser

EQUATIONS DES BASCULES D GERANT LE GRAFCET (Sans RAZ)

D0 = [a0 + /d0.X0] + init -> D0=(X1*(/P1*/P2*/P5)) + ((/P1*/P2*/P5)*X0) + init

D1 = [a1 + /d1 * X1]*/init -> D1=[(X0*(P1+P2+P5)) + ((P1+P2+P5)*X1)]*/init

Remarque : comme il n'y a pas de RAZ dans ces bascules, init apparaît dans l'équation de D.

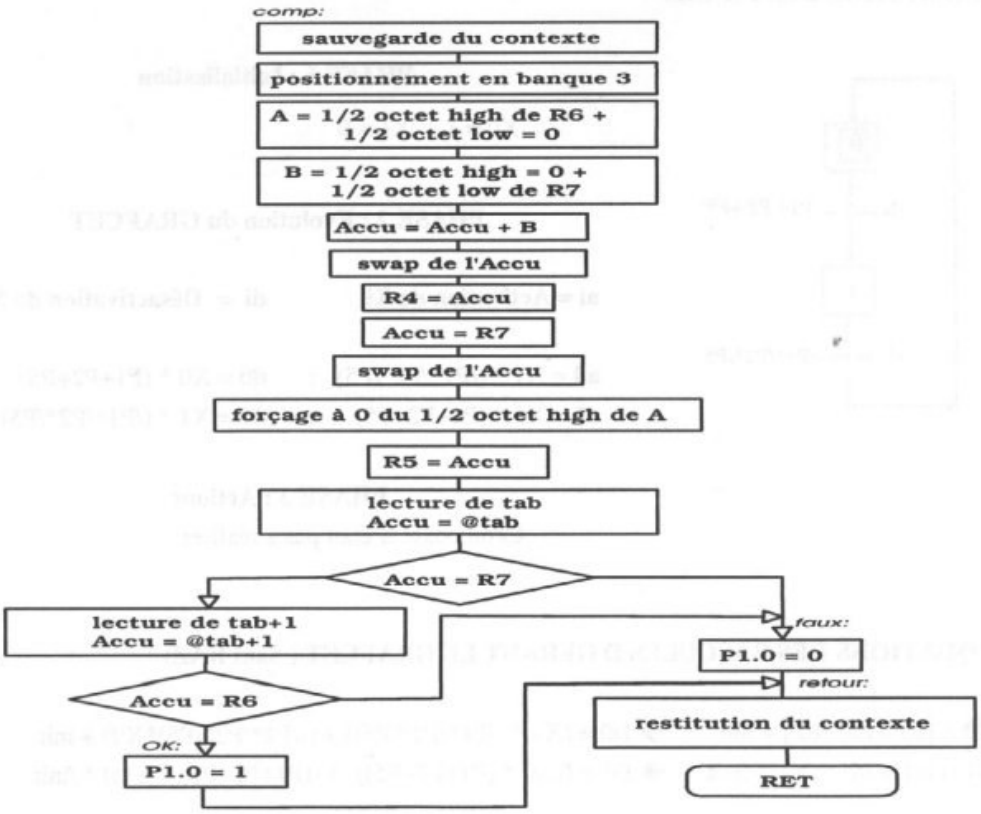
II. MICROCONTROLEUR : 6 points

Donner l'algorithme (sous forme d'Organigramme) et le sous-programme

Il existe de nombreuses solutions possibles, avec des décalages de bit ou des calculs sur demi-octets. trois solutions proposées.

1ère solution proposée :

1. remettre en forme les données contenues par R7 et R6.
2. le résultat de la remise en forme sera mis dans R5 et R4.
3. comparaison des données de la table avec R5 et R4.



comparaison de R7 et R6 avec les octets placés aux adresses tab et tab+1 de la mémoire programme.

principe utilisé : réorganisation de R7 et R6 et stockage dans R5 et R4 pour comparer a TAB et TAB+1

comp:

```
 ; sauvegarde du contexte

    push psw
    push acc
    push b
    push dph
    push dpl

 ;  positionnement sur la banque 3
```

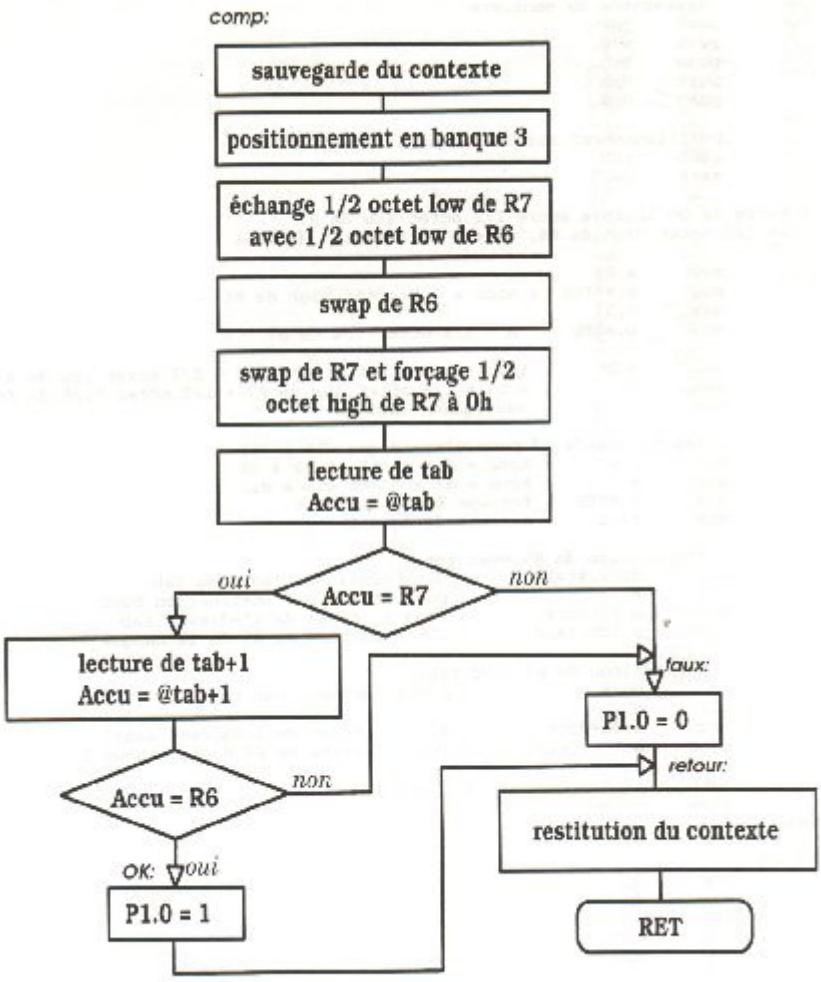
```
    setb rs1
    setb rs0

;  faire le OU logique entre 1/2 octet low de R7
;  et 1/2 octet high de R6, mettre le résultat dans R4

    mov a,R6
     anl a,#0Fh  ;  Accu = 1/2 octet high de R6
     mov b,R7
     anl b,#0Fh  ;  B = 1/2 octet low de R7

      orl  a,b    ;   Accu = 1/2 octet high de R6 + 1/2 octet low de R7
      swap a      ;   Accu = 1/2 octet low de R7 + 1/2 octet high de R6
      mov  R4,a   ;   sauvegarde dans R4

; traitement de R7 pour n'avoir que d15 à d12

       mov       a,R7     ;   Accu = d15 à dl2 et d3 à d0
      swap       a        ;   Accu = d3 à d0 et d15 à dl2
       anl            a,#0Fh   ;   forçage à 0 de a7 à a4
       mov     r5,a     ;   sauvegarde dans R5

 ; comparaison de R5 avec tab :
     mov    dptr,#tab   ;   dptr contient l'adresse tab

     clr a              ;   pour utiliser l'instruction MOVC
     movc a,@a+dptr     ;   Accu = ‘case’ de l'adresse tab:

     cjne a,1Dh,faux    ;   1Dh = adresse de R5 de la banque 3

 ; comparaison de R4 avec tab+1

     inc   dptr         ;   positionnement sur tab+1
     clr   a
     movc  a,@a+dptr    ;   Accu = 'case' de l'adresse tab+1
     cjne  a,lCh,faux   ;   1Ch = adresse de R4 de la banque 3
 ok :
     setb  p1.0             ;   c’est bon
     sjmp  retour

faux :

      clr   p1.0

retour :
     pop        dpl
```

```
pop     dph
pop     b
pop     acc
pop     psw
```

; Remarque: Le temps d'exécution de ce programme est de 64 cycles machine.

2ème solution proposée :

mise en forme de R7 et de R6 identique à la table TAB
comparaison de la table avec R7 et R6



Sous programme de la solution 2

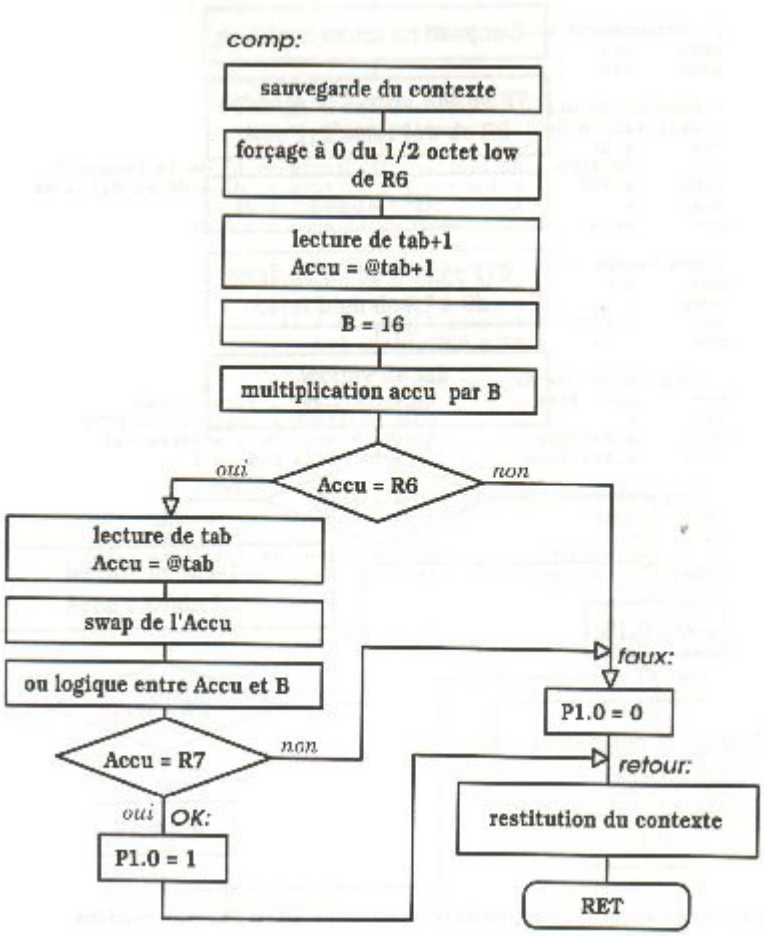
comparaison de R7 et R6 avec les octets placés aux adresses tab et tab+1 de la mémoire programme

principe utilisé : réorganisation de R7 et R6 pour comparer a TAB et TAB+1

```
comp:

; sauvegarde du contexte
      push psw
      push acc
      push dph
      push dpl

; positionnement sur la banque 3
      setb rsl
      setb rs0

; échange de d11 à d8 (R7) avec d3 à d0 (R6)
; utilisation de l'instruction xched    a,@Ri

       mov            a,R6
       mov            R0,#1Fh   ;  R0 contient l'adresse de R7 de la banque 3
       xchd           a,@RO     ;  R7low <-> R6low, Accu = d7 à d4 et d11 à d8
      swap            a         ;  Accu = d11 à d8 et d7 à d4
      mov             R6,a      ;  chargement de R6 avec d11 à d4

 ; traitement de R7 pour n'avoir que d15 à d12
      mov   a,R7      ;  Accu = d15 à d12 et d3 à d0
      swap  a         ;  Accu = d3 à d0 et d15 à d12
      anl   a,#0Fh    ;  forçage à 0 de a7 à a4
      mov   r7,a      ;  R7 = 0000 d15 à D12

 ; comparaison de R7 avec tab

      mov   dptr,#tab  ;  dptr contient l'adresse tab
      clr   a          ;  pour utiliser l'instruction MOVC
      movc  a,@a+dptr  ;  Accu = 'case' de l'adresse tab
      cjne  a,1Fh,faux ;  1Fh = R7 de la banque 3

; comparaison de R6 avec tab+1

     inc   dptr       ;  positionnement sur tab+1
      clr   a
      movc  a,@a+dptr  ;  Accu = 'case' de l'adresse tab+1
     cjne  a,1Eh,faux ;  1Eh = R6 de la banque 3

ok :
```

```
        setb            p1.O    ;  c'est bon
        sjmp  retour

faux :

        clr   p1.O

retour:

        pop   dp1
        pop   dph
        pop   acc
        pop   psw
        ret
```

; Remarque: Le temps d'exécution de ce programme est de 56 cycles machine.



```
; Sous programme de la solution 3

; principe de cette solution:
; aller chercher tab+1 et tab pour les comparer à R6 et R7
; utilisation de la multiplication pour créer le décalage de 4 eb
; Nota les instructions cjne a,Ri et anl Ri,#data n'existent pas
 ; le traitement se fait sur R7 et R6 en se servant de leurs adresses
 ; il n'est donc pas nécessaire de se positionner en banque 3.

Comp :

; sauvegarde du contexte
        push psw
        push acc
        push b
        push dph
        push dpl

        anl     1Eh,#0F0h     ;   raz des 4eb low de R6
        mov   dptr,#tab+1     ;   dptr contient l'adresse tab
        clr     a             ;   pour utiliser l'instruction MOVC
        movc a,@a+dptr        ;   Accu = 'case' de l'adresse tab
        mov  b,#16            ;   pour décalage vers la gauche
        mull  ab              ;    Accu = 1/2 octet low de tab+1 x 16
                                  ;     B low = 1/2 octet high de tab+1
        cjne   a,1Eh,faux     ;    Accu = R6 ?
        mov   dptr,#tab       ;    si oui traitement de tab
        clr     a
        movc  a,@a+dptr
        swap  a          ; Accu high = octet low de tab
        orl     a,b       ; fusion entre Accu et B
        cjne    a,1Fh,faux   ; Accu = R7 ?

ok :                                    ; R7 et R6 sont bons
      setb            p1.O              ; c'est bon
      sjmp            retour

faux:                                   ; R7 et R6 sont non conformes

     clr              p1.O

retour :                        ;  restitution du contexte

      pop             dpi
      pop             dph
      pop             b
```

```
pop     acc
pop     psw
ret
```

; Remarque: Cc sous programme utilise moins d'instructions que les précédents, mais le temp
d'exécution de ce programme est de 59 cycles machine.



QCM SF11

SUJET ET CORRIGE

Durée : 45 minutes. Les bonnes réponses sont ***** entourées d'étoiles *****

1 Dans un semi-conducteur dopé P, les porteurs majoritaires sont des :

éléctrons
***** trous *****
ions positifs issus du dopant

2 Dans la zone de charge d'espace, la charge fixe est constituée par :

les porteurs majoritaires
les porteurs minoritaires
***** les ions issus des dopants ******

3 Dans un semi-conducteur, la diffusion des porteurs est due à l'existence entre deux zones de :

un gradient de température
****** un gradient de concentration ******
un gradient de volume

4 Dans une jonction P-N:

L' anode est la zone N
***** L'anode est la zone P *****
*****La cathode est la zone N *****

5 Une diode au Silicium parcourue par un courant de 10 mA offre une résistance dynamique de :

26 ohms
***** 2.6 ohms *****
26 kohms

6 Un transistor à effet de champ se commande par:

Un courant
***** une tension *****
une résistance

7 Un transistor a effet de champ régi par l'approximation parabolique:

prend une forme conique
a sa tension Vgs qui dépend du carré du courant Id
***** a son courant Id qui dépend du carré de la tension Vgs *****

8 Un amplificateur opérationnel auto-compensé possède:

une fréquence de coupure basse très élevée
***** une fréquence de coupure haute très basse *****
une fréquence nominale intermédiaire

9 Un système bouclé est:

un système qui ondule
un système qui assure une stabilité inconditionnelle
***** un système qui possède une boucle reliant la sortie à l'entrée *****

10 Un condensateur se comporte comme:

un court-circuit à fréquence nulle
un circuit ouvert à fréquence infinie
***** un court circuit à fréquence infinie *****

11 Dans un condensateur en régime d'impulsion, lors du changement d'état, il y a :

conservation du courant I
***** conservation de la charge Q *****
conservation de la tension V

12 Le circuit de polarisation est obtenu en:

court-circuitant les sources de tension continue
***** remplaçant les condensateurs par des circuits ouverts *****
remplaçant les condensateurs par des circuits fermés

13 Le schéma équivalent dynamique petits signaux d'un circuit amplificateur à transistors s'obtient en :

ouvrant les sources de tension continues
***** court-circuitant les sources de tension continues *****
remuant le circuit

14 Au voisinage de la fréquence de coupure haute d'un amplificateur, les capacités prépondérantes sont:

les capacités de liaison
***** les capacités internes (de jonction) des circuits actifs *****
les capacités de découplage

15 Le théorème de Miller permet de:

remplacer un quadripôle par un dipôle
accroître le réalisme de la stabilisation capacitive
***** en dérivation remplacer une capacité de rétroaction par deux capacités *****

16 Un système stable est décrit dans le plan de Nyquist par une courbe représentant l'évolution de la transmittance en boucle ouverte (A*béta) tracée pour des fréquences croissantes, qui :

entoure le point critique
***** n'entoure pas le point critique *****
passe par le point critique

17 Un amplificateur différentiel fonctionne en régime linéaire:

***** dans une faible plage de tension E centrée sur l'origine *****
dans une plage infinie de tension
pour un courant de référence quasi statique

18 Un oscillateur sinusoïdal de type Colpitts est constitué d'un amplificateur et d'un circuit de retour qui comprend :

trois cellules R-C passe bas
***** une self et deux capacités *****
un quartz et deux capacités

19 Un signal sinusoïdal modulé en fréquence a un spectre constitué de:

une seule raie à la fréquence porteuse
une infinité de raies contiguës d'amplitude croissante
***** de raies dont l'amplitude dépend de l'indice de modulation *****

20 Un dispositif permettant de moduler la fréquence d'un oscillateur sinusoïdal est :

une résistance à capacité variable
***** une diode a capacité variable *****
une fréquence à capacité variable.

MATHS

MA11

Durée : 1heure 30, Documents autorisés, Coordonateur : Arif Mardin.

EXERCICE 1

Utiliser le théorème des résidus pour montrer que $\int_0^{+\infty} \frac{\ln(1+x^2)}{1+x^2} dx = \pi \ln 2$. On pourra considérer avec profit l'intégrale de la fonction $f(z) = \frac{\log(z+i)}{z^2+1}$ sur un contour convenable.

EXERCICE 2

Soit $\mathcal{H}$ un espace de Hilbert complexe et séparable. On désigne par A un opérateur linéaire borné (continu). Montrer que:

1. $Ker A^* = (\mathrm{Im} A)^{\perp}$.
2. Montrer que si A est normal, on a $\|Ax\| = \|A^* x\| \quad \forall x \in \mathcal{H}$ et que par conséquent on a $Ker A = (\mathrm{Im} A)^{\perp}$.
3. Si A est normal et s'il existe $\alpha > 0$ tel que $\|Ax\| \geq \alpha \|x\|$, alors A est injectif, d'image dense (Il suffit de montrer que $A^{\perp} = \{0\}$), inversible (puisque $\mathcal{H}$ est complet), et d'inverse borné par α^{-1}.

$\uparrow$ $(\mathrm{Im}\, A)^{\perp}$

MA11

Durée : 3 heures, Documents autorisés, Coordonateur : A. Mardin

SUJET 1:

Soit A un opérateur linéaire borné auto-adjoint dans un espace de Hilbert complexe séparable, $\mathcal{H}$. Soit k un réel positif tel que $-kI \leq A \leq +kI$.

1) Etablir l'identité

$$4\,\mathrm{Re}\langle Ax, y\rangle = \langle A(x+y), x+y\rangle - \langle A(x-y), x-y\rangle$$

et montrer que

$$|\mathrm{Re}\langle Ax, y\rangle| \leq \tfrac{1}{2}k\left[\|x\|^2 + \|y\|^2\right] \quad \forall x, y \in \mathcal{H}.$$

En déduire que $\|A\| \leq k$, et que

$$\|A\| = \min\{a : a \in \mathbb{R},\ -aI \leq A \leq +aI\}$$
$$= \sup\{|\langle Ax, y\rangle| : x \in \mathcal{H},\ \|x\| = 1\}$$

2) A se dit <u>compact</u> si l'on a la propriété suivante: Pour toute suite x_n qui converge faiblement vers x, la suite Ax_n converge fortement vers Ax, c'est-à-dire:

$$\forall z \in \mathcal{H}: \lim_{n\to\infty}\langle z, x_n - x\rangle = 0 \Rightarrow \lim_{n\to\infty}\|Ax_n - Ax\| = 0.$$

En utilisant les hypothèses et résultats précédents, montrer qu'il existe une suite $x_1, x_2, \ldots$ de vecteurs dans $\mathcal{H}$ telle que la suite réelle $\{\langle Ax_n, x_n\rangle\}$ converge, avec limite $r = \pm\|A\|$. Montrer que $\|Ax_n - rx_n\| \to 0$ quand $n \to +\infty$.

—

SUJET 2 :

1) A tout t réel positif, on associe la fonction φ_t de la variable complexe z définie par :

$$\varphi_t(z) = \frac{e^{tz}}{z^6+1}.$$

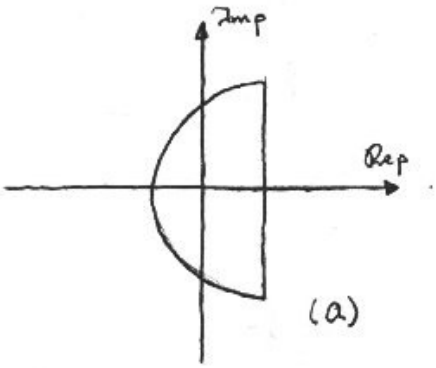
On pose $z_k = \exp[i(\pi/6 + k\pi/3)]$. Quels sont les pôles de φ_t ? Vérifier que le résidu de φ_t au pôle $\bar{z}_k$ ($k=0, 1$ ou 2) est égal au conjugué du résidu de φ_t au pôle z_k ($k=0, 1$ ou 2). En déduire l'original de la fonction F définie par

$$F(z) = \frac{1}{z^6+1}.$$

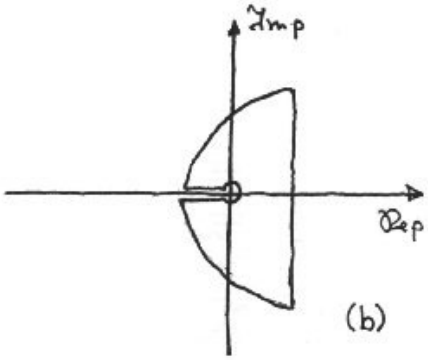
2) Calculer, en utilisant la formule de Bromwich, les originaux des fonctions :

$$F_1(p) = \frac{p}{(p+1)^3(p-1)^2} \quad ; \quad F_2(p) = \frac{\exp(-ap^{1/2})}{p} \quad a \in \mathbb{R}^+.$$

On utilisera le contour (a) pour le calcul de $\mathcal{L}^{-1}F_1$, et le contour (b) pour $\mathcal{L}^{-1}F_2$:



(a)



(b)

—

SUJET 3:

1) Calculer le développement en série de Fourier de la fonction périodique, de période $2L$, qui vaut

$$2i\pi t \exp\left(\frac{i\pi t}{2L}\right)$$

sur l'intervalle $[-L,+L]$. En déduire la somme

$$\sum_{n=-\infty}^{+\infty} \frac{1}{(2n-1)^2}$$

2) Soit f une fonction continue, nulle en dehors de l'intervalle $[-L,+L]$. Montrer que sa transformée de Fourier est indéfiniment dérivable, et que

$$\hat{f}'(x) = \frac{8L}{\pi} \sum_{n=-\infty}^{+\infty} \frac{(-1)^n}{(2n-1)^2} \hat{f}\left(x - \frac{2n-1}{4L}\right)$$

En déduire que

$$|\hat{f}'(x)| \leq 2\pi L \int_{-\infty}^{+\infty} |f(t)|\, dt.$$

—

PS11

Module PS 11 : Probabilités
Durée : 1,5 H
Sans document
Date : 27.1.1998
Coordonnateur : DELMAS Jean Pierre

Justifier chaque réponse de façon concise. ***Répondre sur le questionnaire même***, si possible dans les limites qui vous sont proposées. Il est conseillé de faire un rapide brouillon sur une feuille à part avant de reporter votre réponse. Les trois exercices sont "indépendants".

1] Théorème de transfert

Qu'appelle-t-on théorème de transfert pour le calcul de l'espérance mathématique d'une variable aléatoire $Y = g(X)$, lorsque l'on connaît la loi de probabilité de la variable aléatoire X ?

Si X est une variable aléatoire de loi uniforme sur $[0,2\pi]$ et $Y = \cos^2(X)$, calculer $E(Y)$ et $var(Y)$.

2] Changement de variable

Soit X_1 et X_2 des variables aléatoires indépendantes. X_1 est continue de loi uniforme sur $[-1,+1]$, et X_2 est discrète de loi uniforme à valeurs dans $\{-1;+1\}$. Déterminer la loi de probabilité de la somme $Y = X_1 + X_2$ et du produit $Z = X_1X_2$.

3] A propos du mythe des martingales

On considère un joueur de casino qui joue à un jeu composé de plusieurs parties successives indépendantes. Au cours d'une partie donnée, le joueur a une probabilité p [resp. $1-p$] de gagner [resp. de perdre] ($p <$ **Erreur !**). S'il gagne, il reçoit le double de sa mise, et le jeu s'arrête (c'est là un point essentiel !). S'il perd, il rejoue une nouvelle partie avec une mise doublée. Le jeu se poursuit tant que le joueur n'a pas gagné une partie. La mise initiale (de la 1ère partie) est a.

3.1] On appelle G, la variable aléatoire représentant le bénéfice net du joueur à l'issue du jeu. Déterminer sa loi de probabilité. Remarque ?

3.2] On appelle N la variable aléatoire représentant le nombre de parties jouées par le joueur. Déterminer sa loi de probabilité.

3.3] Calculer l'espérance mathématique E(N) et la variance var(N) de N. Quel est le rôle de p. Conclusions

3.4] Commentaires sur ce placement financier !

(on donne à tout hasard les formules : pour $|x|<1$, **Erreur !**kx^{k-1} **=Erreur !** et **Erreur !**$k(k-1)x^{k-2}$**=Err**

SIGNAL

Module TS 11 : Théorie du Signal et communications analogiques

1ére partie (signal à temps continu)

Durée : 1,5 H

Sans document

Date : 25.5.1998

Coordonnateur : DELMAS Jean Pierre

Justifier chaque réponse de façon concise. ***Répondre sur le questionnaire même***, si possible dans les limites qui vous sont proposées. Il est conseillé de faire un rapide brouillon sur une feuille à part avant de reporter votre réponse. Les questions ne sont pas totalement indépendantes.

Considérons donc la transmission numérique d'une suite $a_k \in \{0,a\}$ de symboles binaires ($a>0$) sous la forme équiprobables et indépendants de débit **Erreur !**bits/s à l'aide d'une transmission bande de base sur cable sous forme d'une modulation d'amplitude d'impulsions lorsque le canal de transmission perturbe la transmission par addition d'un bruit blanc gaussien.

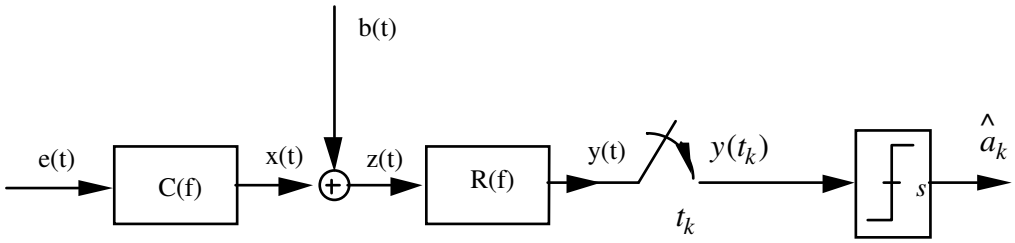
Le signal émis est :

$$e(t) = \textbf{Erreur !} a_k \mathrm{rect}_T(t-kT)$$

On rappelle la notation $\mathrm{rect}_T(t)$ = **Erreur !**.

I Lorsque le canal de transmission est peu dispersif (canal transparent dans le support fréquentiel de $\mathrm{rect}_T(t)$, c'est à dire débit binaire **Erreur !**faible par rapport à la bande passante du canal de transmission), le canal peut être modélisé par un simple affaiblissement α et un simple retard τ (α et τ étant fonction de la longueur du cable de transmission). Donner l'expression de la fonction de transfert $C(f)$ et la réponse impulsionnelle $c(t)$ de ce cable.

II La théorie de la réception optimale en présence de bruit blanc gaussien centré montre que le ***récepteur optimal*** (celui qui minimise la probabilité d'erreurs) doit être constitué d'un filtre de réponse impulsionnelle $r(t)$ = **Erreur !**rect$_T$(t− **Erreur !**) suivi d'un échantillonneur aux instants t_k et d'un détecteur à seuil s. ($y(t_k)$ au dessus du seuil, le détecteur décide **Erreur !**



III Justifier qualitativement la structure du récepteur qui reçoit $z(t)$. On montrera en particulier que :

$y(t)$ = **Erreur !**$z(\square)$d$\square$

On utilisera par exemple la transmission de la séquence {+a, 0, 0, 0,+a,+a, 0} pour $k\square\{0,1,2,3,4,5,6\}$. Représenter graphiquement le signal $e(t)$ ainsi que $y(t)$ en absence de bruit $b(t)$ pour $t\square[0,8T]$.

IV Représenter $y(t)$ en présence de bruit additif $b(t)$ dans le cas où une erreur intervient sur le bit associé à a_3. Expliquer qualitativement le comportement de ce récepteur face au bruit blanc gaussien centré qui s'ajoute à $x(t)$.

V Le bruit $b(t)$ (bruit thermique d'amplification) est un signal aléatoire blanc dans la bande de fréquence du signal $e(t)$, c'est à dire que l'on peut supposer que sa densité spectrale de puissance est

$S_b(f)$ = **Erreur !**

avec B >> **Erreur !**et son amplitude suit une loi de probabilité gaussienne centrée, c'est à dire de densité de probabilité :

$f(x)$ = **Erreur !**e$^{-\textbf{Erreur !}}$ avec $\square^2 = N_0B$.

On appelle $n(t)$ le signal issu du filtrage de $b(t)$ par le filtre de fonction de transfert $R(f)$. Montrer que les valeurs des signaux présents à l'entrée du détecteur à seuil sont respectivement :

$Y = A + N$

avec $A = \square a_k$ soit : $A\square\{ 0 ; +a\square\}$ avec des probabilités égales à **Erreur !**et N variable aléatoire gaussienne centrée dont on déterminera la variance $\square^2_{,n}$ = E[$n^2(t_k)$]. Quelles hypothèses peut-on faire sur les variables aléatoires A et N ? En donner les raisons.

Après avoir décomposé l'évènement {erreur} à l'aide d'intersections et d'unions d'événements {+a émis}, { 0 émis}, {N<− **Erreur !**} et {N>+ **Erreur !**}, déterminer la

probabilité d'erreur de la transmission numérique. ***On prendra soin d'expliquer les diverses étapes du raisonnement***.
On exprimera cette probabilité d'erreur à l'aide de la fonction $Q(x)$ **Erreur !**

Il ne nous reste plus qu'à te souhaiter Bon Stage !

