RAPHAEL LAURENT

DIDACTICIEL POUR UN APPRENTISSAGE STRUCTURE DE L'ORTHOGRAPHE D'USAGE

Ecole Industrielle de
l'Etat PERUWELZ
Annee Scolaire 1989-1990

juin 1990

RAPHAEL LAURENT

DIDACTICIEL POUR UN APPRENTISSAGE STRUCTURE DE L'ORTHOGRAPHE D'USAGE

Ecole Industrielle de
l'Etat PERUWELZ
Annee Scolaire 1989-1990

juin 1990

Nous remercions vivement Monsieur R. RIVIERE et P. RIVIERE pour leur appui et la sympathie qu'ils nous ont manifestée.

INTRODUCTION

Le travail que je vais vous présenter s'inscrit dans le cadre de l'enseignement assisté par ordinateur (**E.A.O.**). Son contenu porte sur l'acquisition progressive et rationnelle d'un vocabulaire orthographique de base . Il se réfère à deux recherches complémentaires : une exploitation pédagogique et une approche informatique à des fins de programmation de l'apprentissage de l'orthographe d'usage .

Sur le plan des intentions pédagogiques, les fondements de la recherche ont été développés dans un ouvrage publié par R. RIVIERE : " Vocabulaire de base du français écrit " (Editions De Boeck-Wesmael, 1987) . Seuls les 500 premiers mots fondamentaux programmés pour les 2 premières années de l'enseignement primaire ont fait l'objet d'une étude approfondie . Celle-ci s'est développée selon 2 approches différentes mais complémentaires . Dans un premier temps, dans une perspective d'organisation thématique, les mots ont été répartis en 50 textes pour des exercices lacunaires à raison de 10 mots par thème . Lors d'une seconde approche, les mêmes mots ont été systématiquement repris dans 50 séries de 10 phrases lacunaires pour des exercices plus personnalisés d'imprégnation, de structuration, de remédiation et de révision.

Sur le plan de la programmation informatique des intentions pédagogiques, la recherche s'est centrée sur la conception et la réalisation d'un didacticiel aux fins d'expérimentation sur 2 types de population : des adolescents de l'enseignement secondaire professionnel spécial et des élèves de 3e année primaire de l'enseignement ordinaire . Il en a été rendu compte dans un mémoire de maîtrise en informatique et éducation, présenté en décembre 1987 à l'Université de l'Etat à Mons par P. RIVIERE " Conception d'un didacticiel pour un apprentissage structuré et progressif d'un vocabulaire orthographique de base " .

Des contraintes pédagogiques et budgétaires ont conduit l'auteur à limiter ses ambitions à l'utilisation des COMMODORE 64 et 128 appareils les plus commercialisés à l'époque dans les écoles, et à utiliser le basic 2.0 comme langage de programmation . Une procédure originale en matière de traitement des réponses et de choix des feed back y est proposée dans une perspective de remédiations spécifiques par des branchements différenciés (distinction entre erreur de type sémantique ou de type orthographique et éventuellement faute de frappe).

Une expérimentation poursuivie pendant toute une année scolaire auprès d'élèves d'enseignement spécial a permis de conclure à l'efficacité du didacticiel pour des adolescents en difficulté d'apprentissage .

Chapitre 1

1.1 Buts et Contraintes

Le but de ce travail a été de mettre au point un didacticiel utilisable par un élève de troisième année primaire, avec :

- un minimum d'interventions du maître dans les exercices .
- une amélioration de l'analyse de la réponse (rapidité et efficacité)
- un temps d'attente pour l'élève le plus court possible .
- un minimum de manipulations des disquettes .
- une plus grande lisibilité des résultats d'un élève .
- une meilleure présentation des exercices .

Le didacticiel comporte, entre autres, une partie élève (exerciseur) mais aussi une partie maître qui permet a celui-ci de se rendre compte des éventuelles difficultés de l'élève et donc de pouvoir y remédier .

En plus du didacticiel il m'a fallu aussi développer un éditeur pour créer et modifier les différent fichiers .

Les principales difficultés rencontrées lors du développement de ce didacticiel étaient :

- l'analyse de la réponse qui donne le type de faute (frappe, orthographe, sens) remédiations en fonction du nombre et du type de faute.
- la mise au point de l'éditeur et des différents fichiers ainsi que de leurs index respectifs .
- la gestion des différents écrans et de leur succession, en fonction des réponses de l'élève .
- l'optimalisation des temps d'attente .
- la saisie de la réponse .
- la gestion des résultats d'un élève et de leur exploitation dans la partie maître .

1.2 Analyse Fonctionnelle

1.2.1 Etude de l'existant

Le programme existait déjà sur COMMODORE 64 en basic 2.0 compilé . Comme le COMMODORE 64 possède un clavier QWERTY celui-ci a dû être redéfini ainsi que les différents caractères qui le composent . Les fichiers "utilisateur" (textes, phrases, feed back, mots) du COMMODORE 64 étaient placés sur plusieurs disquettes(10) étant donné la capacité du lecteur de disquette limitée à 170 Ko .

Raison du choix du COMMODORE 64 :

Dans son mémoire P. RIVIERE développe l'argumentation suivante .

Le choix d'un ordinateur repose, d'une manière générale, sur l'importance relative accordée à deux facteurs :

- d'une part, les possibilités offertes par la machine, sur le plan du traitement de l'information et des capacités de mémoire;

- d'autre part, des réalités dans lesquelles s'inscrit l'E.A.O. : coût du matériel, facilité d'utilisation, justification psycho-pédagogiques, implantation dans les écoles, langage de programmation utilisé ...

Compte tenu du rapport "performance/coût" et de la constante évolution du marché de l'informatique, ils nous a semblé que le COMMODORE 64 répondait le mieux à nos attentes pour les raisons suivantes :

- les COMMODORE 64 et 128 sont effectivement les appareils les plus commercialisés, ceux qui se trouvent le plus souvent dans les écoles des deux réseaux, ceux qui émargent le plus fréquemment du budget de l'Education Nationale,

- le COMMODORE 64 n'est pas trop coûteux et il est suffisamment performant et réputé fiable pour répondre à nos exigences de programmation tant pour le contenu (corpus de 500 mots) que pour ses possibilités de traitement de la matière,

- un programme conçu pour COMMODORE 64 (indépendamment du fait qu'il tourne aussi sur COMMODORE 128) est facilement adaptable pour être utilisé sur des ordinateurs plus coûteux et plus performants, ce qui malheureusement ne se révèle pas vrai pour l'opération inverse .

Enfin, comme nous voulions nous placer dans l'optique des services qu'un ordinateur peut rendre à l'enseignement, il nous a paru préférable plutôt que de viser des produits haut de gamme pour appareils sophistiqués de rester dans des limites raisonnables tout en visant une efficacité maximum compte tenu des réalités budgétaires .

Les fichiers élèves comprennent les information suivantes :

- l'identification des différents élèves : nom,prénom
- les réponses erronées fournies par chaque élève avec, en référence, le mot correctement orthographié
- la comptabilisation des phrases lacunaires pour chaque erreur rencontrée
- le temps mis par chaque élève pour effectuer chaque exercice

Les fichiers élèves sont organisé de la façon suivante (système CODASYL) :

- dans le premier, le fichier du maître, est inscrit le nom de chacun des élèves
- dans le second, le fichier des esclaves, sont enregistrées toutes une série d'informations concernant l'activité des élèves.

Les fichiers utilisateurs se divisent en fichiers textes et phrases :

1) Textes

Les informations sont enregistrées les unes après les autres dans un fichier à accès séquentiels dont le nom a été précisé au moyen d'un code à 3 caractères : 1 lettre, 1 trait, 1 chiffre impair .

2) Phrases

Elles ont toutes été regroupées dans un fichier commun à 5 séquences successives de manière à pouvoir être utilisées telles quelles dans la phase "révision". Le programme "lect ph" permet de retrouver directement et facilement la série de phrases qui correspondent aux besoins de l'apprenant . Les squelettes relatifs aux différents mots sont stockés dans le fichier du même nom . Quant aux feed back, ils ont été également regroupés dans un même fichier d'accès direct .

1.2.1.1 Critiques

Les critiques porteront sur le choix du matériel et sur résolution des différents aspects du problème .

Choix du matériel.

Le commodore 64 est un micro-ordinateur possédant 64 ko de mémoire vive et une vitesse d'horloge de 1 Mhz . Ces caractéristique ajoutées aux performances du lecteur de disquette 1541 donnent des temps d'accès et de traitement relativement importants . Ces données impliquent des temps d'attente pouvant aller au-delà de la minute, ce qui est très long pour un élève .

Résolution des différents aspects du problème.

Les observations qui suivent découlent directement ou indirectement du choix du matériel .

Le langage utilisé est le basic 2.0 qui est limité dans ses possibilités .

L'utilisation du système CODASYL optimalise l'occupation du disque mais augmente les temps d'accès .

L'utilisation du mode d'accès séquentiel au lieu du direct .

La longueur du squelette ne doit pas être trop importante pour ne pas avoir des temps d'analyse de la réponse trop longs, ce qui nuit à l'exactitude de celle-ci.

L'affichage des textes se fait sur 40 colonnes ce qui limite fortement la taille des textes et nuit à une bonne présentation .

Les fichiers d'exercices se trouvent sur 10 disquettes; donc, beaucoup de manipulations de celles-ci et aussi plus de possibilités d'erreur(s).

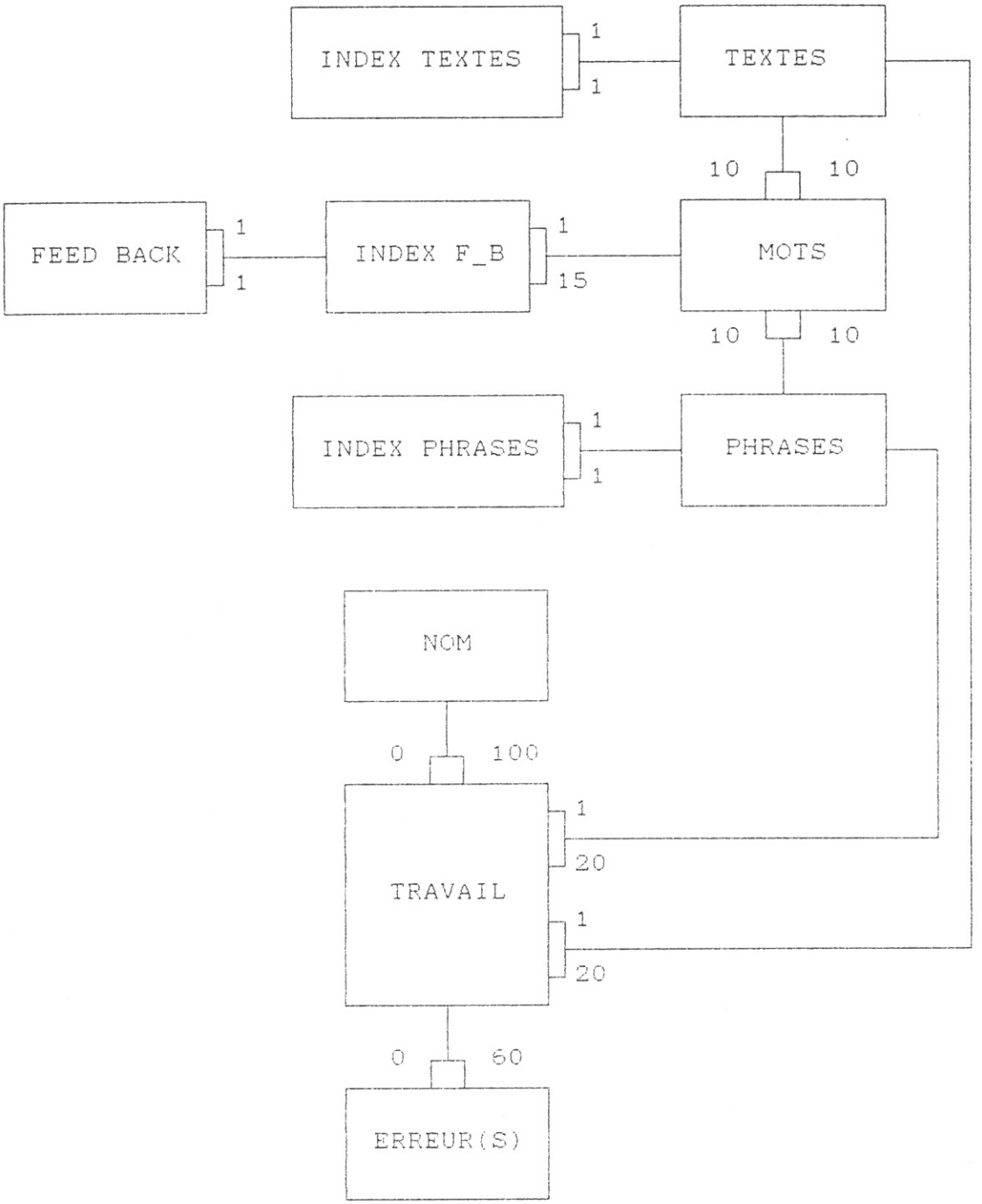
Le temps de chargement et de préparation des textes et phrases prend parfois plusieurs minutes .

La mise en page des textes n'est pas faite dans le programme; donc, temps d'attente supplémentaire .

Les résultats des élèves ne peuvent être enregistrés que sur 5 textes et 5 séries de phrases . Donc, il faut supprimer les résultats pour passer à la suite et ceci 10 fois .

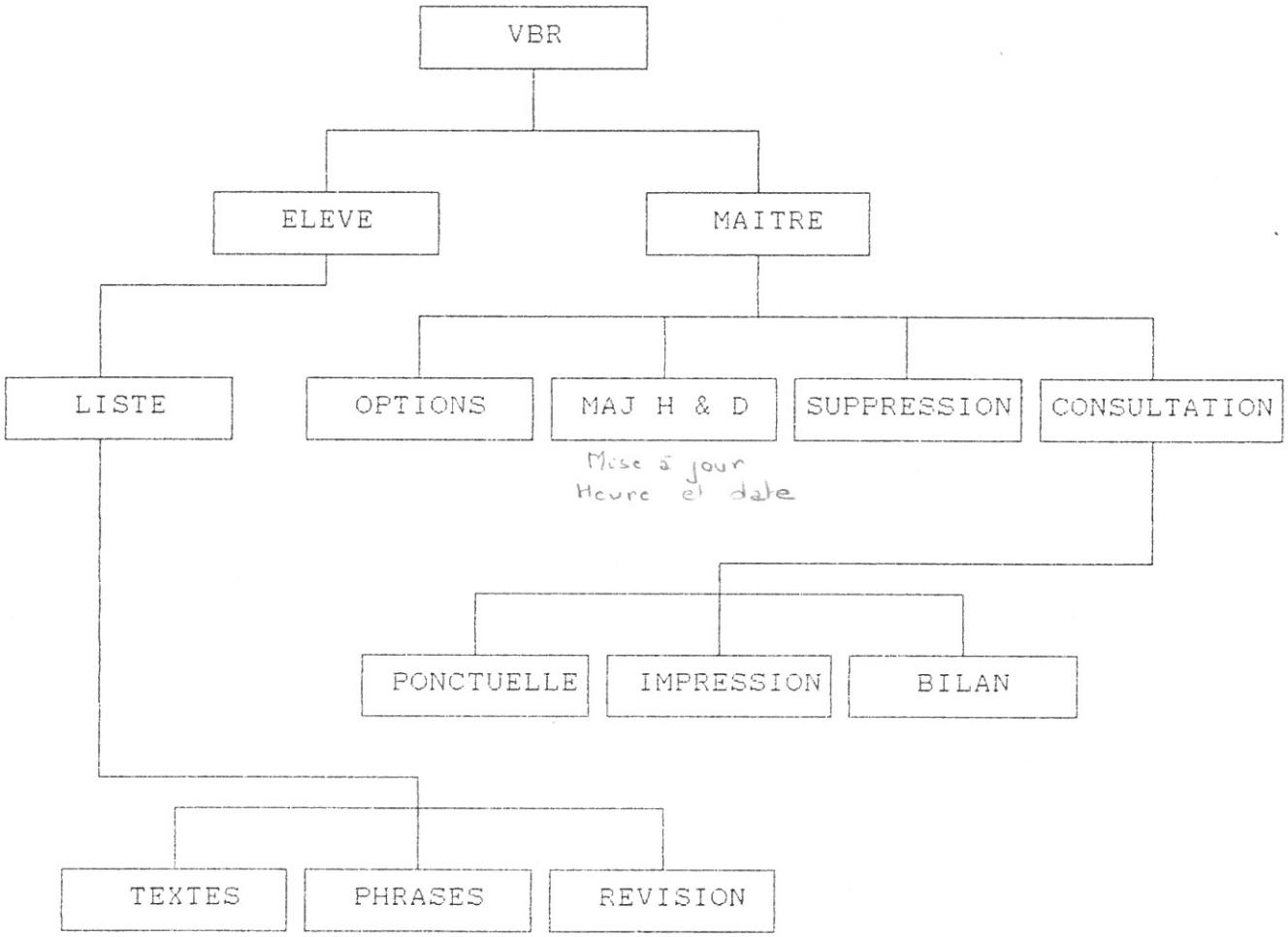
Conclusion : Le reproche le plus important que l'on peut formuler concernant ce programme est que le matériel choisi est inadapté au problème, et aussi de n'avoir pas su diminuer les temps d'attente en choisissant des solutions moins gourmandes en temps .

1.2.2 Entités et Relations



1.2.3 Fonctions & Extraction

Les fonctions et extractions pour le programme VBR s'organisent comme suit :



1.2.3.1 Fonctions Elèves

Liste des élèves : charger la liste des élèves , afficher la liste,

- bouger la barre de sélection dans la liste avec les flèches,
- choisir un nom dans la liste avec return,
- quitter la liste avec esc, et revenir au menu principal,
- ajouter un nom a la liste avec [A].

Presser [A] si votre nom n'est pas dans la liste

Ajout d'un élève à la liste : introduction du nom et du prénom jusqu'à ce que l'élève réponde par "o" à la question ‹‹ est-ce correct (Oui/Non) : ? : ›› .

Bonjour . Comment t'appelles-tu ?

Ecris ton prénom (ex. Jean ou Marie) :

RAPHAEL

Continuer : ↵

Ecris ton nom (celui de ta famille) :

LAURENT

Continuer : ↵

Vérification

PRENOM : RAPHAEL

NOM : LAURENT

Est-ce correct (Oui ou Non) ? :

Exercice textes : affichage de la note explicative pour les 5 premiers textes,

- quitter les explications et revenir à la liste des élèves avec esc,

- continuer et passer au texte avec return,

ROGER, lis bien le texte qui va suivre

car 10 mots vont disparaître

et tu devras les retrouver.

QUITTER : [ESC] CONTINUER : ↵

Chargement dans le fichier texte du contenu approprié; puis, préparation du tableau avec le texte, les mots étudiés, les différents squelettes ainsi que les adresses des feed back.

Affichage des phrases du texte, avec passage à la phrase suivante en pressant return , avec retour à la liste des élèves en utilisant [ESC] .

Un coq vous parle

Je n'ai pas de bouche mais ma tête se termine par
un bec court et pointu.
Mes plumes protègent bien mon corps contre le froid.
Les poules sont mes meilleures amies: elles me suivent
partout.
Toute la journée, nous passons notre temps à gratter la
terre avec les griffes de nos pattes.
La fermière nous dit souvent: Je vous aime bien tous.
Toi, mon coq, pour ta crête rouge et tes belles plumes
brunes.
Et vous, mes poules, pour vos bons oeufs frais.
Chaque matin, qui chante cocorico ?
C'est moi, votre ami le coq.

QUITTER : [ESC] CONTINUER : ↵

Apparition du texte lacunaire après la dernière phrase, début de l'exercice .

Un coq vous parle

0.0 / 0

Je n'ai pas de bouche mais _____ tête se termine par un bec court et pointu.
Mes plumes protègent bien _____ corps contre le froid.
Les poules sont _____ meilleures amies: _____ me suivent partout.
Toute la journée, nous passons _____ temps à gratter la terre avec les griffes de _____ pattes.
La fermière nous dit souvent: Je vous aime bien tous.
Toi, mon coq, pour _____ crête rouge et _____ belles plumes brunes.
Et _____, mes poules, pour vos bons oeufs frais.
Chaque matin, qui chante cocorico ?
C'est moi, _____ ami le coq.

ECRIS TA REPONSE

CONTINUER : ↵

- à la première faute d'orthographe, affichage du message d'erreur " CE MOT EST MAL ORTHOGRAPHIE " puis de " Essaie encore une fois, PRENOM "

- après la 2ème faute d'orthographe, affichage du message d'erreur " CE MOT EST MAL ORTHOGRAPHIE " puis de " Regarde bien le mot qui va suivre . " et enfin du mot attendu pendant 5 s,

- après la 3ème faute d'orthographe, affichage du message d'erreur " CE MOT EST ENCORE MAL ORTHOGRAPHIE " puis de " Regarde encore le mot qui va suivre . " et enfin du mot attendu avec la partie fautive mise en évidence,

- après la 4ème faute d'orthographe, affichage du message d'erreur " CE MOT EST TOUJOURS MAL ORTHOGRAPHIE " puis de " regarde pour la dernière fois et aussi longtemps que tu veux le mot qui va suivre . " et enfin du mot attendu aussi longtemps que l'élève le désire,

- à la 5ème faute, affichage du mot attendu dans le texte; puis, passage à la suite de l'exercice ,

- à la première faute de sens, affichage du message d'erreur " CE N'EST PAS LE MOT ATTENDU', puis de " Ecris un autre mot .",

- après la 2ème faute de sens, affichage du message d'erreur " CE N'EST PAS LE MOT ATTENDU " , puis de " Tu as 10 secondes pour lire la LISTE qui va suivre . " et enfin de la liste des mots attendus en majuscules et triés par ordre alphabétique,

- après la 3ème faute de sens, affichage du message d'erreur " CE N'EST PAS ENCORE LE MOT ATTENDU ", puis de " Tu as 20 secondes pour lire la LISTE qui va suivre. ", et enfin de la liste des mots attendus,

- après la 4ème faute de sens, affichage du message d'erreur " CE N'EST PAS ENCORE LE MOT ATTENDU ", puis de " Tu as 5 secondes pour choisir entre 2 mots. " et, enfin, de deux mots extraits de la liste des mots attendus (le mot attendu + un autre mot pris dans la liste),

- après la 5ème faute de sens, affichage du message d'erreur " CE N'EST TOUJOURS PAS LE MOT ATTENDU ", puis de " Recopie le mot qui va suivre. " et enfin du mot attendu.

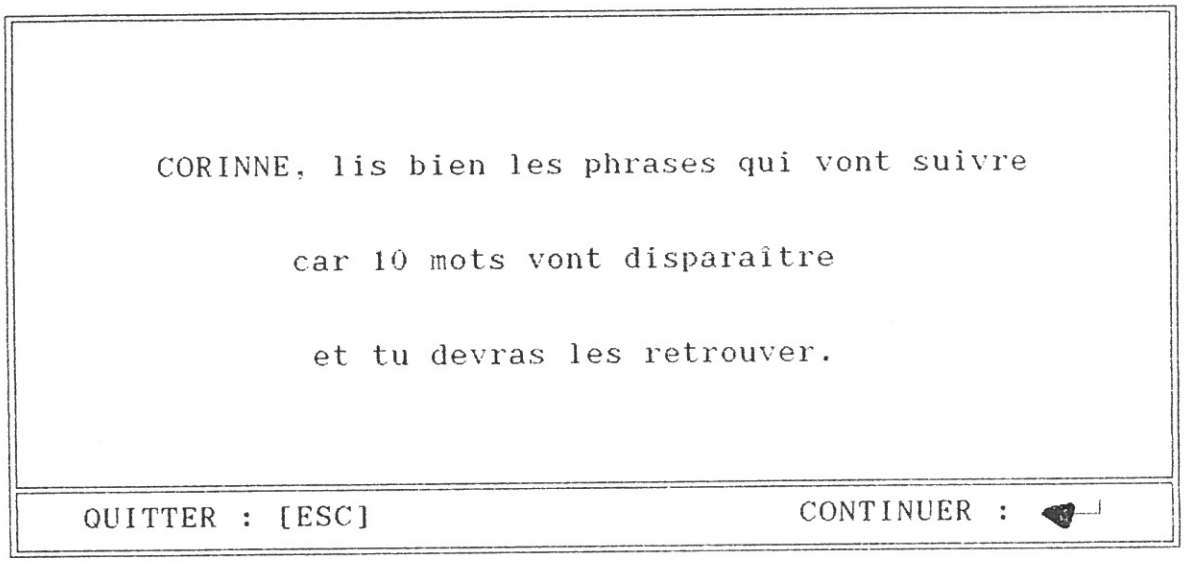
Après le test, écriture dans le fichier de l'élève des différents renseignements sur ses prestations

Et pour finir incrémentation du dernier exercice fait dans le fichier général des élèves .

Exercice phrases : affichage de la note explicative pour les 5 premières phrases,

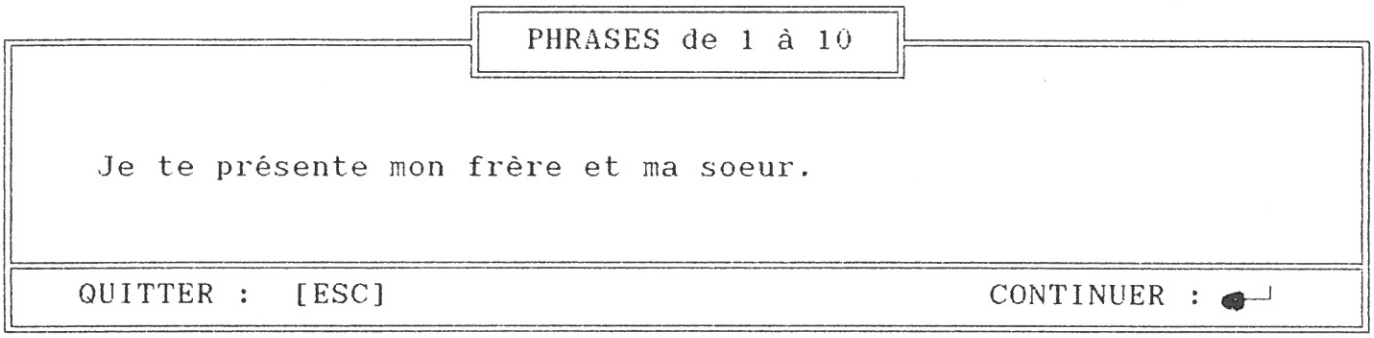
- quitter les explications et revenir à la liste des élèves avec esc,

- continuer et passer aux phrases avec return,

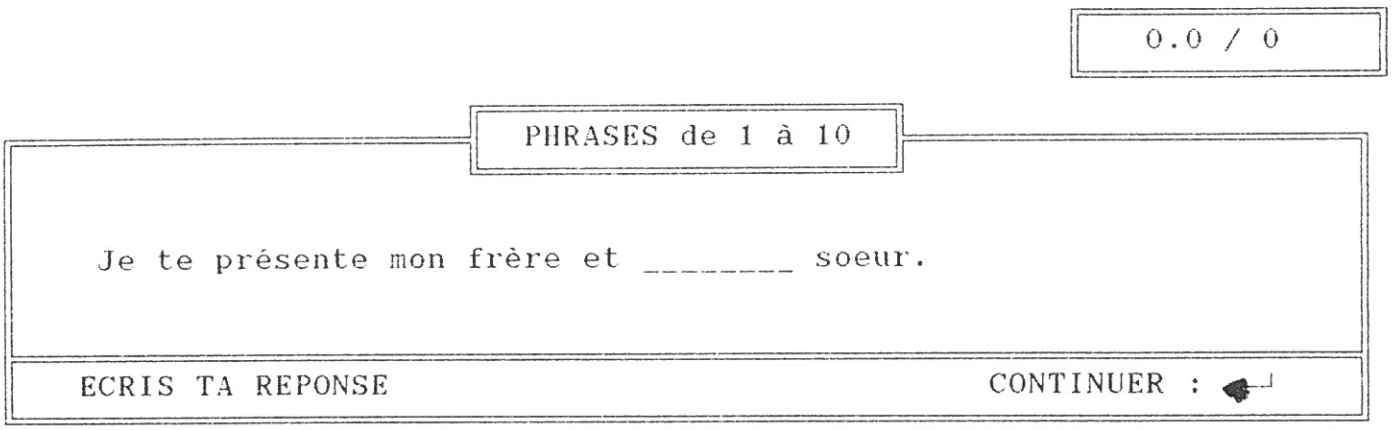


Chargement dans le fichier phrases du contenu approprié; puis, préparation du tableau avec les phrases, les mots étudiés, les différents squelettes ainsi que les adresses des feed back.

Affichage des phrases , avec passage à la phrase suivante en pressant return , avec retour à la liste des élèves en utilisant [ESC] .



Apparition de la première phrase lacunaire après la dernière phrase de la série, début de l'exercice .



- à la première faute d'orthographe, affichage du message d'erreur " CE MOT EST MAL ORTHOGRAPHIE ", puis de " Essaie encore une fois, PRENOM "

- après la 2e faute d'orthographe, affichage du message d'erreur " CE MOT EST MAL ORTHOGRAPHIE ", puis apparition du ou des feed back

- après la 3e faute d'orthographe, affichage du message d'erreur " CE MOT EST ENCORE MAL ORTHOGRAPHIE ", puis apparition du ou des feed back avec mise en évidence de la partie fautive,

- après la 4e faute d'orthographe, affichage du message d'erreur " CE MOT EST TOUJOURS MAL ORTHOGRAPHIE ", puis idem 3, mais le feed back reste affiché ,

- à la 5e faute affichage du mot attendu dans la phrases en inverse; après un certain temps, passage à la suite de l'exercice .

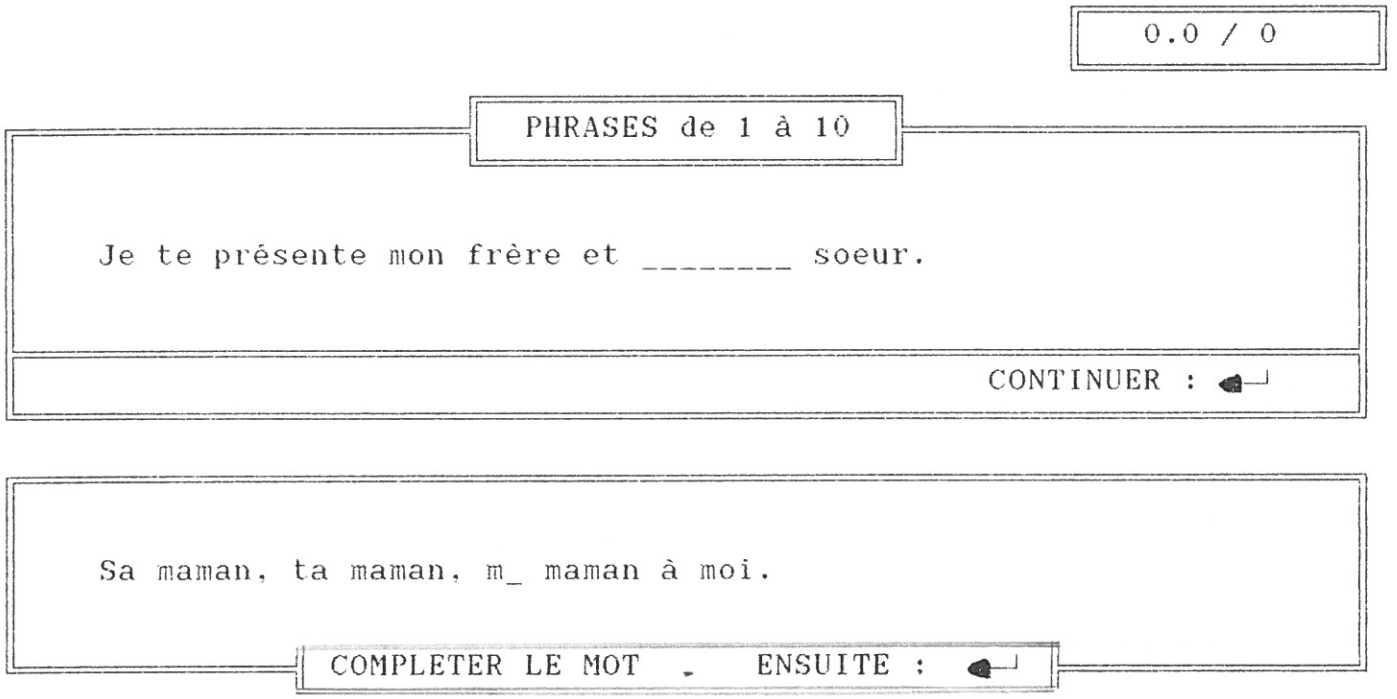
- à la première faute de sens, affichage du message d'erreur " CE N'EST PAS LE MOT ATTENDU', puis de " Ecris un autre mot .",

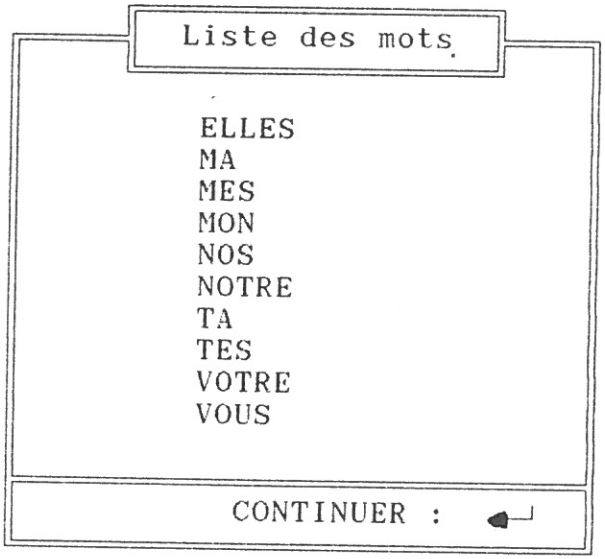
- après la 2e faute de sens affichage du message d'erreur " CE N'EST PAS LE MOT ATTENDU " , puis de " Tu as 10 secondes pour lire la LISTE qui va suivre . " et enfin de la liste des mots attendus en majuscules et triés par ordre alphabétique,

- après la 3e faute de sens, affichage du message d'erreur " CE N'EST PAS ENCORE LE MOT ATTENDU ", puis de " Tu as 20 secondes pour lire la LISTE qui va suivre. ", et enfin de la liste des mots attendus,

- après la 4ème faute de sens, affichage du message d'erreur " CE N'EST PAS ENCORE LE MOT ATTENDU ", puis de " Tu as 5 secondes pour choisir entre 2 mots. " et, enfin, de deux mots extraits de la liste des mots attendus (le mot attendu + un autre mot pris dans la liste),

- après la 5ème faute de sens, affichage du message d'erreur " CE N'EST TOUJOURS PAS LE MOT ATTENDU ", puis de " Recopie le mot qui va suivre. " et, enfin, du mot attendu.





Après le test, écriture dans le fichier de l'élève des différents renseignements sur ses prestations

Et pour finir incrémentation du dernier exercice fait dans le fichier général des élèves.

Exercices de révision : idem phrases mais l'étude porte sur toutes les phrases dans lesquelles ont été commises des fautes d'orthographe . Donc, il peut y avoir de 0 à 50 phrases à revoir.

Suppression :

On affiche la liste des mots et après le choix du maître du nom à supprimer , on demande une validation de la suppression .

SUPPRESSION

Liste des élèves

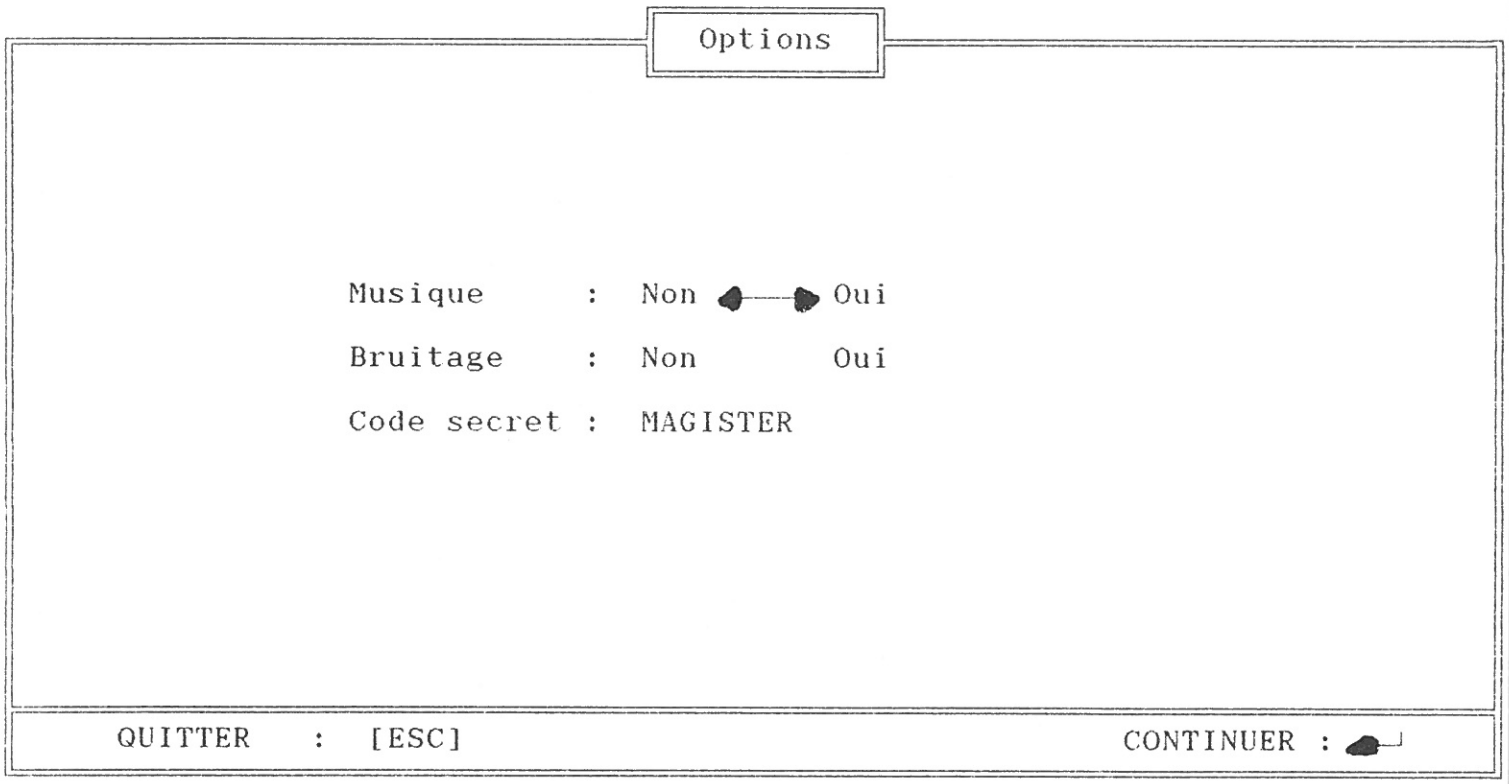
BEHAR	JACQUES	JOFFRE	JEAN-MICHEL
COAT	CORINNE	LAURENT	FRANCOISE
DJIBRIL	FREDERIC	LAURENT	RAPHAEL
DUPUY	PAUL	LERMUSIAUX	PASCAL
GARRET	YANN	MAMANE	BETTY
GOLDBERGER	ROGER	MAUROIT	VICTORIA
GORS	MICHELLE	RIVIERE	ROBERT
GORS	RENE	ROUX	IVAN
HEUILLARD	YVES	SCHIFF	HEINRICH
JALUZOT	FRANCIS		

BOUGER : ↑↓ CHOISIR : ↵ QUITTER : [ESC]

En êtes-vous sûr (Oui/Non) ? :

Options

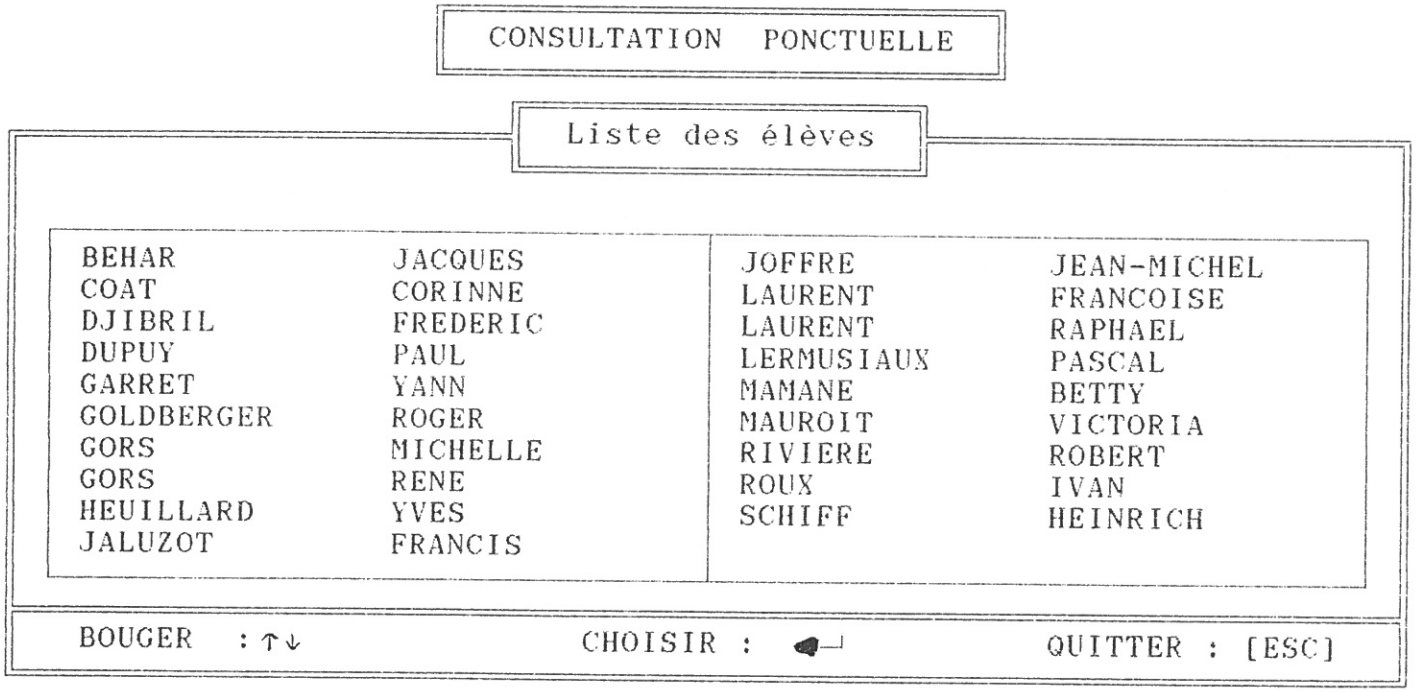
Dans les options, on peut choisir que le programme fonctionne avec ou sans musique et/ou bruitage; on a aussi la possibilité de changer le mot de passe .



Consultation ponctuelle :

Affichage des résultats de l'élève pour un texte ou une série de phrases .

La consultation est composée de deux parties : une liste où l'utilisateur choisit un nom d'élève; puis, une consultation ponctuelle des résultats de l'élève .



Le maître peut passer à l'exercice suivant, au précédent , au premier, au dernier, possibilité d'imprimer les résultats, et enfin de revenir à la liste des élèves .

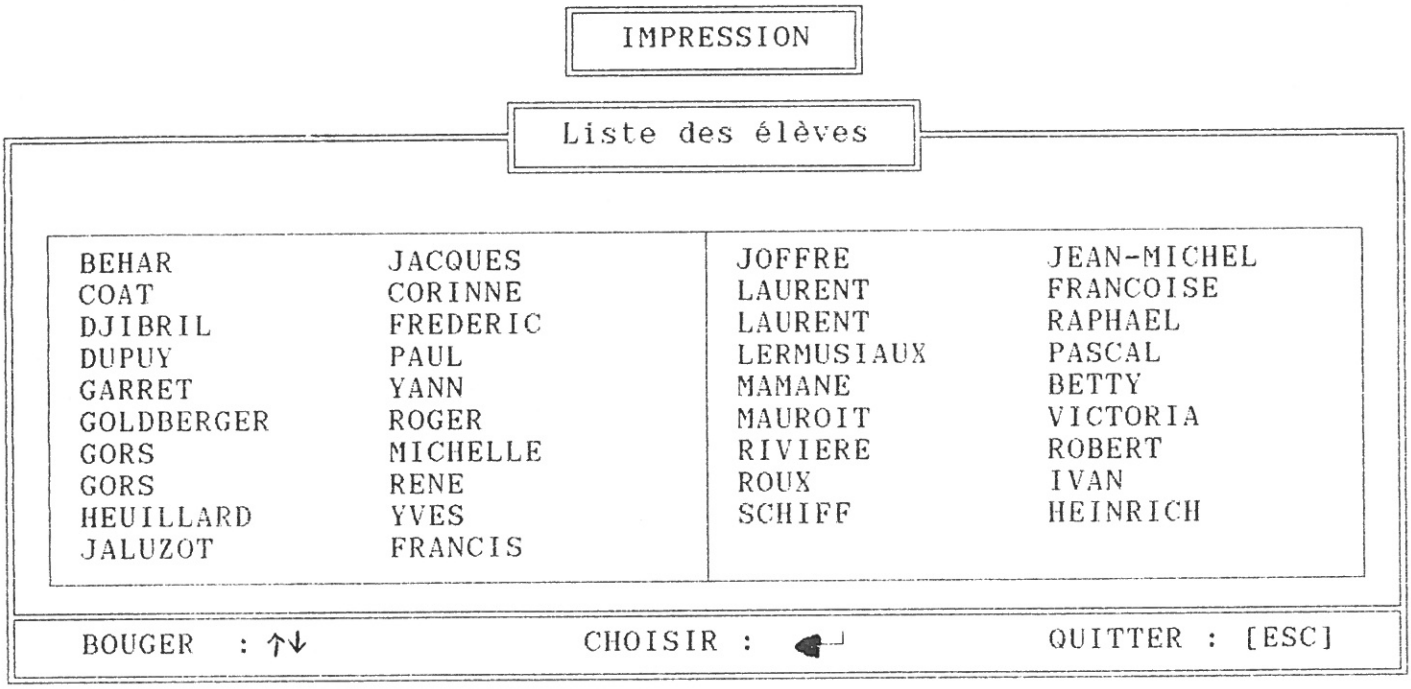
NOM DE L'ELEVE	DATE DE L'EXERCICE	HEURE	SCORE	TEMPS
AUROIT VICTORIA	17 avril 1990	10:22:10	19.0	0:01:32

Erreur de sens	Texte n°7	Erreur d'orthographe
	chambre	chabre
	large	
	fenêtres	fenêtre
	soleil	
	murs	
	avec	
	fleurs	
endorir	endormir	
	chaudes	
	nuit	

[S]uivant [P]récédent [D]ébut [F]in [I]mprimer [Q]uitter

Impression :

L'impression est composée de deux parties : une liste où l'utilisateur choisit un nom d'élève; puis, des résultats à imprimer .



Puis il doit indiquer la série de 10 résultats (textes + phrases) par les quels il veut commencer ainsi que la dernière .

Impression

De la série (1 à 2) : 1

à la série (1 à 2) : 2

Exemple d'impression :

NOM DE L'ELEVE	DATE DE L'EXERCICE	HEURE	SCORE	TEMPS
MAUROIT VICTORIA	17 avril 1990	10:22:10	19.0	0:01:32

Erreur de sens	Texte n°7	Erreur d'orthographe
	chambre	chabre
	large	
	fenêtres	fenêtre
	soleil	
	murs	
	avec	
	fleurs	
endorir	endormir	
	chaudes	
	nuit	

Bilan :

Le bilan affiche le nom de l'élève avec, en face de celui-ci, le numéro du dernier exercice fait (texte, phrases ou aucun exercice effectué).

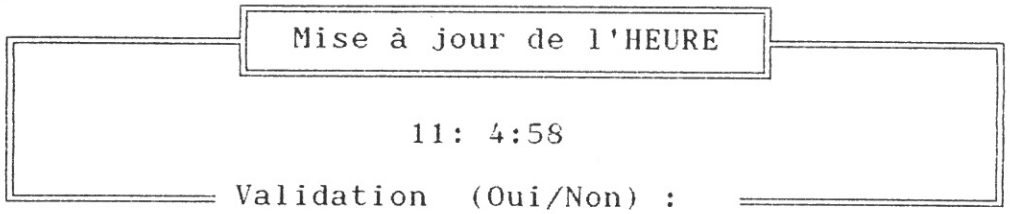
Bilan

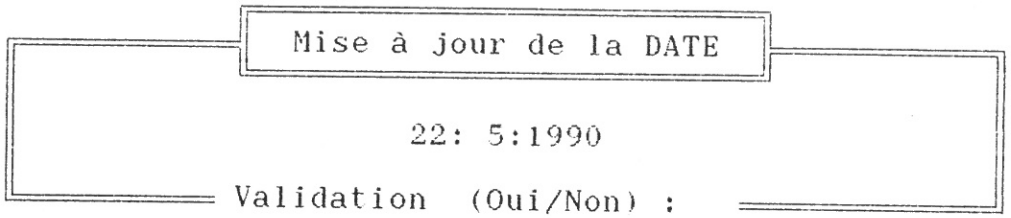
BEHAR JACQUES	Texte n°2
COAT CORINNE	Texte n°1
DJIBRIL FREDERIC	Texte n°2
DUPUY PAUL	Phrases de 41 à 50
GARRET YANN	Texte n°5
GOLDBERGER ROGER	aucun exercice effectué
GORS MICHELLE	Phrases de 1 à 10
GORS RENE	aucun exercice effectué
HEUILLARD YVES	aucun exercice effectué
JALUZOT FRANCIS	Texte n°1
JOFFRE JEAN-MICHEL	aucun exercice effectué
LAURENT FRANCOISE	aucun exercice effectué
LAURENT RAPHAEL	Texte n°5
LERMUSIAUX PASCAL	Texte n°1
MAMANE BETTY	Texte n°1
MAUROIT VICTORIA	Texte n°13
RIVIERE ROBERT	aucun exercice effectué
ROUX IVAN	aucun exercice effectué
SCHIFF HEINRICH	aucun exercice effectué

Quitter : [ESC]

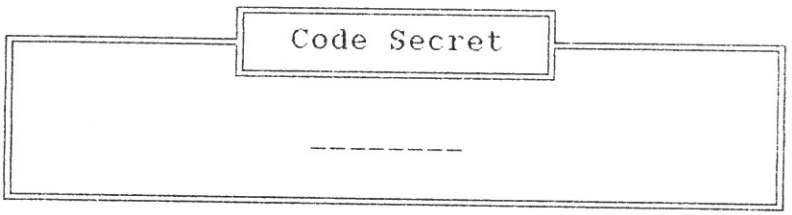
Mise à jour de l'heure et de la date système :

Introduction de l'heure jusqu'à ce que l'utilisateur déclare l'heure valide (idem pour la date).

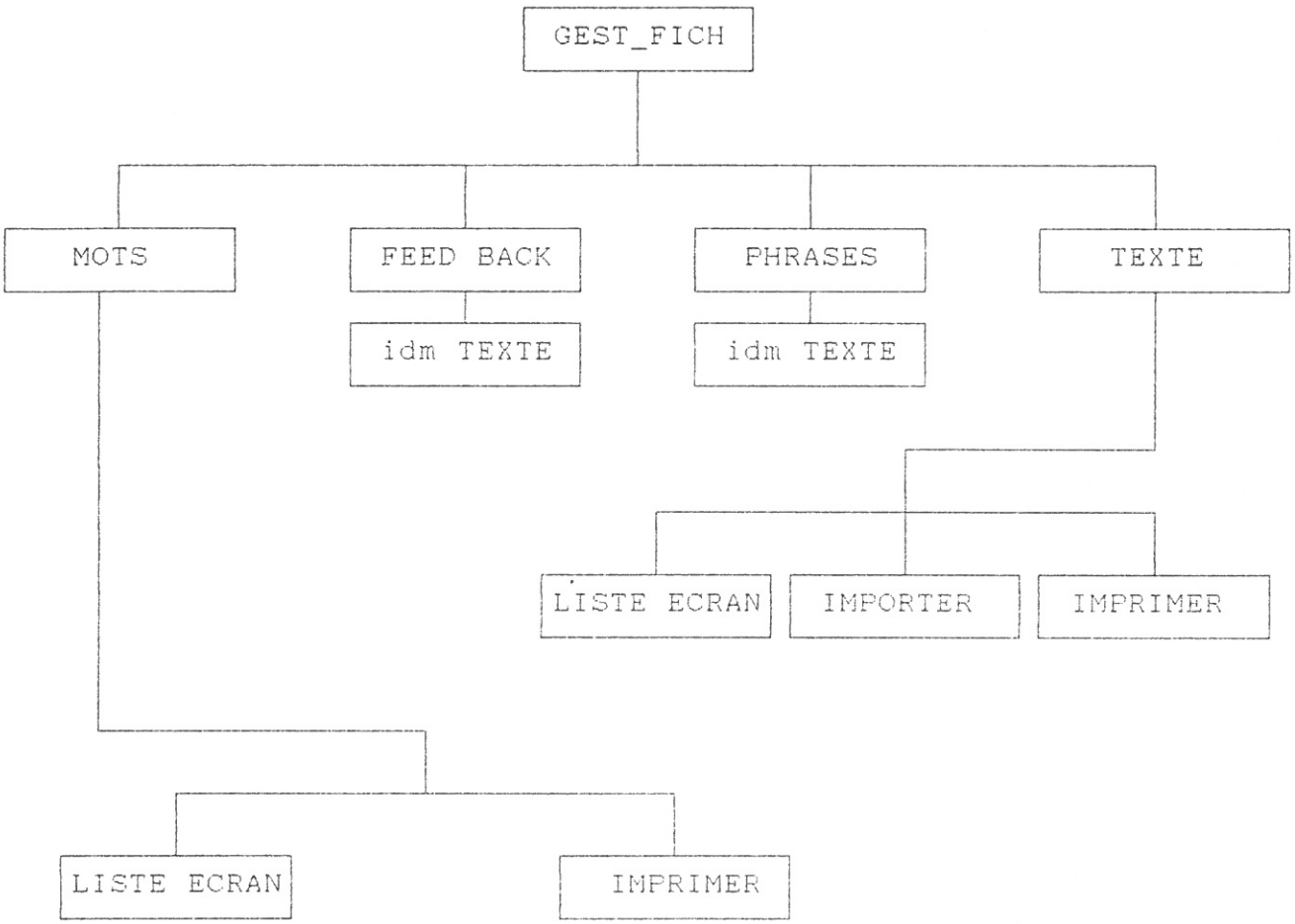




Code secret : permet de limiter l'accès à la partie maître .



Les fonctions et extractions pour le programme GEST_FICH s'organisent comme suit :



1.2.3.4 Fonctions Editeur

Editeur texte

- **Total écran** : éditeur texte avec possibilité de supprimer une ligne, de la modifier , d'insérer une ligne , et aussi d'ajouter une ou plusieurs lignes à la fin du fichier texte .

EDITION textes

de la joie et il y a de l'(amusement) pour les petits
comme pour les grands.
#Le sapin de Noël
Il a neigé toute la (semaine), pendant sept longs jours,
du dimanche matin jusqu'au (samedi) soir.
En cette veille de Noël, Pierre, le (fils) du boulanger
accompagne son père dans la forêt. (Quelques) coups de
hache ont suffi pour (couper) un petit sapin. Pierre
n'a plus qu'à le ramener à la maison sans (casser) une
seule petite branche.
Il ne reste plus (maintenant) qu'à l'installer dans un
coin de la (salle) à manger et à le (garnir) pour le
décorer avec des boules brillantes et des fils d'or et
d'(argent).
#

MODIF.	: [M]	AJOUTER	: [A]	SUPP.	: [SHIFT-F1]
BOUGER	: ↑↓	INSER.	: [F1]	QUITTER	: [ESC]

- **Importer** : permet d'utiliser un traitement de texte pour la remise en forme du fichier et ensuite de remettre le fichier ascii sous fourme de fichier pascal ainsi que de recréer l'index du fichier texte .

- **Imprimer** : comme son nom l'indique imprime une série de textes dans un intervalle donné par l'utilisateur .

IMPRESSION Textes

Début d'impression : 1

Fin d'impression : 20

QUITTER : [ESC]

Editeur phrases

idem textes .

Editeur feed back

idem éditeur texte mais création et structure d'index différents

Editeur mots

- **Total écran** : éditeur texte avec possibilité de supprimer une ligne, de la modifier , d'insérer une ligne , et aussi d'ajouter une ou plusieurs lignes à la fin du fichier texte .

EDITION mots

MOT : était

SQUE : 0,'/é,e,è,ê,a/0,i,î,r,s/t,d,tt,tr/a,e,é,è,ê/it,0/

ADR : 750 750 750 751 752 752 0

MODIF. : [M] AJOUTER : [A] SUPP. : [SHIFT-F1]
BOUGER : ↑↓ QUITTER : [ESC]

1.3 Analyse Organique

1.3.1 Description des Programmes

Le but de la description des programmes n'est pas de reprendre tout ou partie des fonctions/extractions mais bien d'expliquer certains points de la programmation qui me semblent intéressants.

Le langage utilisé pour développer ce programme est le TURBO PASCAL 5.5 ainsi que le turbo professionnel 4.0. La technique des overlay a dû être utilisée, par le programme VBR, pour raisons de mémoires vives insuffisantes.

Il y a deux programmes d'une part le programme VBR qui comprend, un module maître et un module élève , d'autre part il y a aussi un éditeur (GEST_FICH) qui a permis de créer les fichiers nécessaires a VBR .

La fonction PRINTERREADY permet de savoir si l'imprimante est prête ou non selon que la fonction renvoie true ou false. Pour cela, on utilise l'interruption $17 (unit proc).

La procédure PIOFEL a deux fonctions. Elle vérifie si le drive A:\> est prêt à fonctionner; si non, message d'erreur jusqu'à introduction d'une disquette . Puis, elle vérifie si le fichier fel.dat se trouve sur la disquette; si non, elle crée le fichier fel.dat ainsi que les répertoires per, ere, rev (unit proc) .

La procédure PVBCD attend un temps spécifié par le paramètre d (d x 1.5 ms) puis avec la procédure PBUFFC vide le buffer du clavier (unit proc) .

La procédure PPRENDECR se trouve dans le programme VBR. Elle sauvegarde la page écran actuel en mémoire et la restitue quand on le désire selon que le paramètre " pre " est true ou false . Cette procédure fonctionne aussi bien en mode hercules ($B000:0000) qu'en cga,ega et vga ($B800:0000) (unit proc).

La procédure PANA est composée de deux parties distinctes :

- la procédure PANAMOT qui décompose le squelette du mot étudié et le place dans un tableau qui permettra par la suite de former les différentes possibilités de fautes ,

exemple : mot : ma

sque : m,n/0,'/a,as,at,ât,à/

	c1	c2	c3	...	c15
l1	"m" "n"	"" "'"	"a" "as" "at" "ât" "à"	... ...	
l10					

- la procédure PANAREP regarde d'abord s'il ne s'agit pas d'une faute de frappe (triplement d'une consonne ou doublement d'une voyelle) . Si non la longueur du mot est comparée à celle de la réponse et, si celle-ci diffère trop, on considère qu'il y a une erreur de sens . Enfin, si aucun de ces tests ne s'avère vrai, on commence l'analyse de la réponse c-à-d qu'on essaie toutes les combinaisons de fautes possibles pour créer un mot. On regarde si celui-ci se trouve dans la réponse : si oui, le paramètre ortho est true; si non, il est false. La première colonne à être modifiée est la dernière car la plupart des fautes se trouvent en fin de mot (ici, la c3) .

De plus, les fautes les plus courantes, se trouvent près de la ligne 1 . Tout cela, pour améliorer le temps d'analyse . Si, à la fin, aucune combinaison ne convient, on considère qu'il s'agit d'une faute de sens .

La procédure pindex (textes et phrases) détermine le début du texte (#) , le nombre de phrases les mots étudiés et leur emplacement dans le fichier "mots" .

La procédure pindex (feed back) détermine le début et le nombre de phrases .

La procédure pindex (mots) donne simplement une liste alphabétique des mots étudiés .

1.3.2 Description des Fichiers

Les informations se divisent en deux parties : d'une part, les informations qui concernent les exercices et, d'autre part, les informations concernant les résultats de l'élève.

Tous les fichiers utilisés sont du type Table c-à-d

- Durée de vie longue
- Ajout aléatoire de données
- Volumes variables
- Mode d'accès : direct

1.3.2.1 Fichiers Utilisateur

Fichier textes : chaîne de 60 caractères

Fichier indt :

- début du texte, ind.deb : word
- nombre de phrases, ind.nbre : byte
- adresse des mots étudiés dans le fichier mots, ind.lmo : array[1..10] de word

Exemple :

```
#Un coq vous parle
Je n'ai pas de bouche mais (ma) tête se termine par
un bec court et pointu.
Mes plumes protègent bien (mon) corps contre le froid.
Les poules sont (mes) meilleures amies: (elles) me suivent
partout.
Toute la journée, nous passons (notre) temps à gratter la
terre avec les griffes de (nos) pattes.
La fermière nous dit souvent: Je vous aime bien tous.
```

Toi, mon coq, pour (ta) crête rouge et (tes) belles plumes brunes.
Et (vous), mes poules, pour vos bons oeufs frais.
Chaque matin, qui chante cocorico ?
C'est moi, (votre) ami le coq.

Fichier phrases : chaînes de 60 caractères

Fichier indp :

- début des phrases, ind.deb : word
- nombre de phrases, ind.nbre : byte
- adresse des mots étudiés dans le fichier mots, ind.lmo : array[1..10] de word

Exemple :

```
#PHRASES de 1 à 10
Je te présente mon frère et (ma) soeur.*
Ce livre est à moi, c'est (mon) livre.*
Je vais mettre (mes) plus beaux souliers.*
Regarde ces pommes. Comme (elles) sont bien mûres!*
Mon frère et moi, nous conduisons (notre) soeur à l'école.*
Nous viendrons avec (nos) enfants: notre fils et
notre fille.*
Ne bouge pas. Reste un peu à (ta) place.*
J'ai invité (tes) parents, ton père comme ta mère.*
Comment allez-(vous), cher ami ?
Je vais bien. Merci.*
Cher Monsieur, j'ai bien reçu (votre) lettre.*
```

Fichier f_b : chaînes de 60 caractères
Fichier indf_b : début du feed back, ind.deb : word
nombre de phrases, ind.nbre : byte

Exemple:

La montre n'est pas à (m)oi, elle est à (m)a maman.*
S(a) maman, ta maman, m(a) maman à moi.*
Voici (m)a mère et (m)on père.
Ce sont mes parents.*
C'est t(on) bouton ou m(on) bouton ?*
Ces livres sont à (m)oi.
Ce sont (m)es livres.*
Je prends l(es) livres.
Ce sont m(es) livres.*
Regarde les b(elle)s robes.
Comme (elle)s sont jolies!*
Tous ensemble, il(s) chantent et elle(s) chantent.*
Le piano est à (n)ous.
C'est (n)otre piano.*
Ce n'est pas v(o)tre piano.
C'est n(o)tre piano.*
Nous regardons par la fenê(tre).
Nous voyons arriver no(tre) ami.*
C'est (n)otre nouveau chien.
Ce sont (n)os nouveaux chiens.*
Voici trois gros (os) pour n(os) trois chiens.*

Fichier mots :

- mot étudié , mots.mots : chaînes de 15 caractères
- squelette , sque : chaînes de 60 caractères
- adr des f_b, adr : array [1..15] de word

Fichier indm : liste des mots utilisés, triés par ordre alphabétique , mot : chaîne de 15 car

Exemple :

mot : ma

sque : m,n/0,'/a,as,at,ât,à/

adr : 1 1 2 0

mot : morceau

sque : m,n,m'/o,au,eau,ô/r,n,0/c,s,ç,z/eau,0/

adr : 1312 1313 1313 1314 1315 0

mot : élève

sque : 0,'/é,è,e,a,ê/0,i,î/l,ll/è,é,e,ê,a/0,i,î/v,f,vr/e/

adr : 374 374 374 375 375 375 376 377 0

Fichiers param :

- musique : booléen
- bruitage : booléen
- code_se : chaîne de 15 car

Sécurité : La mise à jour des différents fichiers se fait au fur et à mesure du travail. Pour les fichiers "texte, phrases et feed back" , à la fin de la session, il y a création en même temps que les index, de fichiers *.asc qui peuvent servir à restaurer les fichiers PASCAL . Quant au fichier "mots", son volume n'est pas trop important . Il peut donc être sauvé sur disquette ainsi que les autres fichiers .

1.3.2.2 Fichiers Elèves

Fichier fel :

- nom et prénom, nom : chaîne de 15 car
- nom du fichier personnel, nf : chaîne de 8 car
  (nf = les 4 premières lettres du nom + les 4 premières du prénom) .
- nombre d'exercice(s) effectué(s), num : word;
- dernière révision effectuée, rev : byte;

Fichier nf.rev : liste des erreurs commises sur 5 séries de phrases (0 à 50): array [1..50] de 50 booléen .

Fichier nf.per :

- date et heure de début du travail, date_heure : long entier
- temps pour effectuer le travail, te : long entier
- points obtenus, points : byte
- début des fautes dans le fichier ere, deb : word
- nombre de fautes d'orthographe et de sens pour cet exercice, nbre : byte

Fichier nf.ere : fautes d'orthographe et de sens : chaîne de 18 car

Sécurité : La mise à jour des differents fichiers se fait à la fin de l'exercice pendant un temps relativement court . Donc, pas besoin de précautions particulières, si ce n'est un backup régulier de la disquette élève .

Chapitre 2

2.1 Programme VBR

```
program  VBR;

  uses
     ovr,
     tport,
     def,
     proc,
     eleve,
     ecr1,
     maitre;


  {$O eleve}
  {$O ecr1}
  {$O maitre}

procedure PDEBUT;

  const
       menu : tmenu    = ('V.B.R.','Elève','Maitre','Fin',
                          '','','','','');

  var
       r  : byte;

   begin
     PECRAN1;
     videodirecte:=true;
     repeat
       r:=0;
       PMENU(menu,8,5,71,20,3,r);
       curscache;
       case r of
         1 : PELEVES;
         2 : PMAITRE;
       end;
     until r = 3;
     clrscr;
   end;
```

```
begin
  contrlneige :=true;
  attrtexte:=7;
  PDEBUT;
  attrtexte:=7;
  cursnormal;
end.
```

2.2 Unit DEF

```
unit def;

  interface

{*****************************************************************************}
{                                                                       Types
}
{*****************************************************************************}

        { strings différentes longueurs utiles }

  type
        string10=  string[10];
        string15=  string[15];
        string30=  string[30];
        string50=  string[50];
        string60=  string[60];
        string80=  string[80];

        terreur =  string[78];
        tecran  =  array [1..4000]     of char;
        tmenu   =  array [1..9]        of string30;
        ttex    =  array [1..20]       of string[75];

        tmots   =  record
                     mots : string15;                  {mots       }
                     sque : string60;                  {squelette  }
                     adr  : array [1..15] of word;  {Adresse F.B}
                     end;

        trmom   =  record                          { du mot en maj}
                     mom : string15;
                     pl  : byte;
                   end;

        tmo     =  array [1..10]       of tmots;
        tmom    =  array [1..10]       of trmom;
        tmot    =  array [1..10]       of string15;
        tpo     =  array [1..10,1..2]  of byte;
        tlmo    =  array [1..10]       of word;
        tloer   =  array [1..15]       of boolean;
        ttabmot =  array [1..15,1..10] of string[3];
```

```
  ttrav   = record
              mo  : tmo;
              mom : tmom;
              mot : tmot;
              po  : tpo;
              tex : ttex;
              max : byte;
            end;


    { Fichier ELEVES }

  tfel    = record
              nom : string30;
              nf  : string[8];
              num : word;
              rev : byte;
            end;


  tere  = string[20];


   {record des fichiers '.per}

  tper  = record
              date_heure  : longint;
              te          : longint;
              points      : byte;
              deb         : word;
              nbre        : byte;
            end;


{Record des fichiers '.rev }

  trev  = record
              rev : array [1..50] of boolean;
              p10 : byte;
            end;


{Record du Fichier indF_B.dat}

  tind    = record
              deb  : word;
              nbre : byte;
            end;
```

```
        {Record des Fichiers indt.dat et indp.dat}

         tind2  = record
                    deb  : word;
                    nbre : byte;
                    lmo  : tlmo;
                  end;

       { record du fichier indm.dat }

        tindm    =  string15;

        {record des fichiers textes.dat, phrases.dat et f_b.dat }

        tf       = file of string60;

        tfi      = file of tind2;

       {record du fichier param.dat}


        toptions = record
                     musique : boolean;
                     bruitage : boolean;
                     code_se : string15;
                   end;

{*****************************************************************************}
{                                                                        Var
}
{*****************************************************************************}

            {def des fichiers}

      var
           ftextes,                { def du fichier textes              }
           fphrases,               { def du fichier phrases             }
           ff_b     : tf;          { def du fichier feed back           }

           fmots    : file of tmots; { def du fichier des mots         }

           ffel     : file of tfel;    { def du fichier eleves         }

           findm    : file of tindm;   { def du fichier index mots     }

           flmot,
           flmop    : file of tlmo;
```

```
        findt,                          { def du fichier index textes  }
        findp     : tfi;                { def du fichier index phrases }

        findf_b   : file of tind;{ def du fichier index feed_back}

        fper      : file of tper;    { def des fichiers personnels}

        fere      : file of tere;     { def des fichiers erreurs  }

        frev      : file of trev;     { def des fichiers revisions}

        foptions  : file of toptions;

        roptions  : toptions;

        ecran     : ^tecran;

        tabmot    : ttabmot;

        rech      : tfel;

        nom,
        prenom    : string15;
        c         : char;


implementation

      begin

    { définition des noms logiques et physiques des fichiers }

        assign(ftextes ,'DAT\TEXTES   .DAT');
        assign(fphrases,'DAT\PHRASES  .DAT');
        assign(ff_b    ,'DAT\F_B      .DAT');
        assign(fmots   ,'DAT\MOTS     .DAT');
        assign(findt   ,'DAT\INDT     .DAT');
        assign(findp   ,'DAT\INDP     .DAT');
        assign(findm   ,'DAT\INDM     .DAT');
        assign(flmot   ,'DAT\LMOT     .DAT');
        assign(flmop   ,'DAT\LMOP     .DAT');
        assign(findf_b ,'DAT\INDF_B   .DAT');
        assign(foptions,'OPTIONS      .PAR');
        {$i-}
        reset(foptions);
        {$i+}
```

```
  if ioresult<>0 then begin
    with roptions do begin
      musique:=true;
      bruitage:=true;
      code_se:='MAGISTER';
    end;
    rewrite(foptions);
    write(foptions,roptions);
  end
  else begin
    seek(foptions,0);
    read(foptions,roptions);
  end;
  close(foptions);
end.
```

2.3 Unit OVR

```
unit OVR;


    interface

      uses
           tport,
           overlay;

    implementation



    begin

{ init overlay }

      ovrinit('VBR.OVR');

      if ovrok<>0 then
        if ovrnotfound=-2 then
          ovrinit('C:\LANGAGES\PASCAL\EXE&TPU\VBR.OVR')
        else begin
          clrscr;
          gotoxy(12,30);
          write('OVRRESULT : ',ovrresult);
          halt(1);
        end;

    end.
```

2.4 Unit ECR1

```
{F+,O+}

    unit  ecr1;

       interface

*         uses
              tport,
              def,
              graph,
              bgifont,
              bgidriv;

    procedure PECRAN1;

       implementation

    procedure PECRAN1;

      var
           a,b,mx,my : integer;
           rx,ry     : real;


     PROCEDURE Abort(Msg : STRING);
       BEGIN
         Writeln(Msg, ': ', GraphErrorMsg(GraphResult));
         Halt(1);
       END;


       begin

 { chargement de dif driver graph }

         videodirecte:=false;

         IF RegisterBGIdriver(chr(64)CGADriverProc)     < 0 THEN
           Abort('CGA');
         IF RegisterBGIdriver(chr(64)EGAVGADriverProc) < 0 THEN
           Abort('EGA/VGA');
         IF RegisterBGIdriver(chr(64)HercDriverProc)    < 0 THEN
           Abort('Herc');

     {   IF RegisterBGIdriver(chr(64)ATTDriverProc)     < 0 THEN
           Abort('AT&T');
         IF RegisterBGIdriver(chr(64)PC3270DriverProc) < 0 THEN
           Abort('PC 3270');       }
```

```
{ Enregistrement des polices de caracteres graphiques. }
IF RegisterBGIfont(chr(64)SansSerifFontProc) < 0 THEN
  Abort('SansSerif');

  {$ifdef use8514  }
      a:=ibm8514;
      b:=ibm8514hi;
  {$else}
      a:=detect;
  {$endif}

initgraph(a,b,'');
if graphresult <> grok then halt(1);

mx:=getmaxx;
my:=getmaxy;
rx:=720/mx;
ry:=348/my;

setcolor(15);

a:=0;
repeat
  rectangle(round(a/rx),round(a/ry),round(mx-(a/rx)),
  round(my-(a/ry)));
  inc(a,5);
until a>25;

setfillstyle(1,15);

floodfill(round(26/rx),round(26/ry),15);

setusercharsize(round(7/rx),1,round(7/ry),1);
settextstyle(3,0,0);
setcolor(0);
outtextxy(round(100/rx),0,'V·B·R·');

setfillstyle(1,0);
floodfill(round(26/rx),round(26/ry),0);
floodfill(round(350/rx),round(200/ry),0);
floodfill(round(350/rx),round(100/ry),0);
floodfill(round(500/rx),round(200/ry),0);
floodfill(round(520/rx),round(100/ry),0);

setcolor(15);
settextstyle(0,0,0);

outtextxy(round(470/rx),round(280/ry),'VOCABULAIRE
          BASE RIVIERE');
outtextxy(round(470/rx),round(300/ry),'       (500 mots)');

outtextxy(round(50/rx),round(280/ry),'                R.
           LAURENT');
```

```
      repeat
        setcolor(15);
        outtextxy(round(300/rx),round(260/ry),
                  'Pressez une touche');
        delay(400);
        setcolor(0);
        outtextxy(round(300/rx),round(260/ry),
                  'Pressez une touche');
        delay(200);
      until keypressed;

      c:=readkey;
      closegraph;
      clrscr;

    end;

end.
```

2.5 Unit PROC

```
{F-,O-}

 unit PROC;

{*********************}
INTERFACE          {*}
{*********************}


        uses
             dos,
             tpcrt,
             tpedit,
             tpchaine,
             tpint24,
             def;

  ( teste si l'imprimante est prête )

     function PRINTERREADY : boolean;

  ( utilisé quand on veut introduire plus de 20 élèves par disquette )

     function FERREUR_MAX_20_ELEVES:char;

  ( utilisé a la fin de chaque exercices " texte, phrases, révisions "
)

     function  FFIN : boolean;

  ( la liste des noms des élèves )

     procedure PLIST_NOM(var rech    : tfel;  (élève sélectionné )
                         var pl      : word;  (place dans le fichier
                                              ffel)
                         var trouver : boolean; (élève trouvé o/n)
                         var esc     : boolean);(sortir de plist_nom
                                                 o/n)

  ( ouverture ou création du fichier ffel )

     procedure PIOFEL;
```

```
( vide le buffer du clavier )

    procedure  PBUFFC;

( idm mais après un certain delay d )

    procedure  PVBCD(d : word);

( lecture d'une chaîne de car + différentes choses )

    procedure  PSAISIE(var entree : string15;
                           bs,x,y : byte;
                           cht    : boolean;
                       var esc    : boolean);

( recherche dichotomique dans le fichier élève )

    procedure PDICOEL(var x  : tfel;
                      var bi : word;
                      var tr : boolean);

( sauvegarde ou restitution d'un écran );

    procedure PPRENDECR(pre    : boolean);

( affichage d'une chaine de car en mode inverse )

    procedure P1NVERSE (inv    : string80);

( affichage d'un message d'erreur )

    procedure PERREUR (erreur : terreur;
                       x,y    : byte;
                       s      : word);

( affichage d'un cadre en simple ou double ligne avec ou sans titre )

    procedure PCADRE (x1,y1,x2,y2 : byte;
                      titre       : string50;
                      coul        : byte;
                      sd          : boolean);

( affichage d'un menu )

    procedure PMENU (    choix  : tmenu; ( liste des choix );
                         x1,y1  : byte;  ( coin sup gauche )
                         x2,y2  : byte;  ( coin inf droit  )
```

```
    nbre   : byte;  ( nbre de choix   )
var choisi : byte); ( choix retenu    )
```

```
{**********************}
IMPLEMENTATION      {*}
{**********************}
```

2.5.1 Function PRINTERREADY

```
function PRINTERREADY : boolean;
  var
       reg : registers;

  begin
    reg.ax:=$0200;
    reg.dx:=0;
    intr($17,reg);
    if reg.ah=144 then PRINTERREADY:=true
    else PRINTERREADY:=false;
  end;
```

2.5.2 Procedure PIOFEL

```
procedure PIOFEL;

  var
       rech    : tfel;
       a,b     : byte;

   begin
     assign(ffel,'a:\fel.per');
     repeat

       (*$I-*)
       reset(ffel);
       (*$I+*)

       a:=hi(resultint24);
       b:=lo(resultint24);

       if a+b<>0 then begin
         clrscr;
         curscache;
         if (a=$02) then
           perreur(' LECTEUR [A:] NON PRET ',28,12,1000);
         PVBCD(10);
       end

       else begin

         int24off(true);

         (*$I-*)
         reset(ffel);
         (*$I+*)

         if (ioresult<>0) then begin

           rewrite(ffel);

           rech.num:=0;
           rech.rev:=0;
           fillchar(rech.nom,30,#0);
           rech.nf :=chainecar(#32,8);

           write(ffel,rech);

           mkdir('a:\ere');
           mkdir('a:\nom');
           mkdir('a:\rev');

         end
       end;

     until (a+b=0);
```

```
end;
```

2.5.3 Function FERREUR_MAX_20_ELEVES

```
function  FERREUR_MAX_20_ELEVES:char;

  begin

    clrscr;
    PCADRE(5,6,75,16,'',7,false);

    ecritvite('MAXIMUM : 20 ELEVES PAR DISQUETTE. SI VOUS
               VOULEZ ENTRER',10,14,7);

    ecritvite('    UN NOUVEL ELEVE PRENEZ UNE NOUVELLE
              DISQUETTE',12,15,7);

    ecritvite(#204+chainecar(#205,69)+#185,14,5,7);
    changeattr(67,15,7,112);
    ecritvite('Liste des élèves : [ESC]',15,9,112);
    ecritvite('Menu principal : '+#17+#196+#217,15,50,112);
    affichecarlu:=false;
    cachecursdanslitcar:=true;
    litcar('',1,1,$00,[#27,#13],c);

    FERREUR_MAX_20_ELEVES:=c

  end;
```

2.5.4 Procedure PLIST_NOM

```
procedure PLIST_NOM(var rech    : tfel;
                    var pl      : word;
                    var trouver : boolean;
                    var esc     : boolean);


  type
       ttab = array [1..20] of tfel;

  var
       i   : integer;
       tab : ttab;
       che : string60;
       mai : boolean;


procedure PECRAN1(var tab : ttab;
                      esc : boolean);

  var
       x,i,ec2 : byte;
       aff     : string[31];

   begin
     if esc then ec2:=3 else ec2:=0;
     curscache;
     PCADRE(4,2+ec2,76,19+ec2,'Liste des eleves',0,false);
     if not esc then begin
       PCADRE(14,21,66,23,'',7,false);
       ecritvite(' Presser [A] si votre nom n''est pas dans la
                 liste ',22,16,15);
     end;
     PCADRE(6,5+ec2,74,16+ec2,'',7,true);
     ecritvite(#194,5+ec2,40,7);
     for i:=1 to 10 do ecritvite('|',5+i+ec2,40,7);
     ecritvite(#193,16+ec2,40,7);
     ecritvite(#204+chainecar(#205,71)+#185,17+ec2,4,7);
     changeattr(69,18+ec2,6,112);
     ecritvite('BOUGER  :   '     +#27+#23 +#26 ,18+ec2, 8,112);
     ecritvite('CHOISIR :   '     +#17+#196+#217,18+ec2,34,112);
     ecritvite('QUITTER : [ESC]'                ,18+ec2,58,112);

     for i:=1 to filesize(ffel)-1 do read(ffel,tab[i]);
     for i:=1 to filesize(ffel)-1 do begin
       aff:=copy(tab[i].nom,1,15)+' '+copy(tab[i].nom,16,30);
       if i>10 then ecritvite(aff,i-5+ec2,42,7)
       else ecritvite(aff,5+i+ec2,8,7);
     end;
     changeattr(33,6+ec2,7,112);

   end;
```

```
procedure GESTION_CLAVIER(var i   : integer;
                          var pl  : word;
                          var esc : boolean);

  var
       j : byte;
       aj : boolean;


procedure CHANGEMENT(j,i : byte;
                     tab : ttab);

  var
       x,ec2,ec2b : byte;


   begin
     if not aj then begin
       ec2:=8;
       ec2b:=2;
     end
     else begin
      ec2:=5;
      ec2b:=5;
     end;

     if j>10 then changeattr(33,j-ec2b,41,7)
       else changeattr(33,j+ec2,7,7);

     if i>10 then changeattr(33,i-ec2b,41,112)
       else changeattr(33,i+ec2,7,112);

   end;


   begin

     if esc then aj:=false else aj:=true;
     esc:=false;
     repeat

       c:=readkey;

       case ord(c) of

              13 : pl:=i;

              27 : begin
                     esc:=true;
                     exit;
                   end;

           65,97 : if aj then begin
                     pl:=0;
                     exit;
```

```
      end;

0   : begin
        c:=readkey;

        case ord(c) of

          72 : if i=1 then begin
                 j:=1;
                 i:=filesize(ffel)-1;
               end
               else begin
                 j:=i;
                 dec(i);
               end;

          80 : if i=filesize(ffel)-1 then begin
                 j:=i;
                 i:=1;
               end
               else begin
                 j:=i;
                 inc(i);
               end;

          77 : if (i+10<=filesize(ffel)-1)
                  and (i<11) then begin
                 j:=i;
                 inc(i,10);
               end
               else if i-10>=1 then begin
                 j:=i;
                 dec(i,10);
               end;

          75 : if (i+10<=filesize(ffel)-1)
                  and (i<11) then begin
                 j:=i;
                 inc(i,10);
               end
               else if i-10>=1 then begin
                 j:=i;
                 dec(i,10);
               end;

          71 : if i<>2 then begin
                 j:=i;
                 i:=1;
               end;

          79 : if i<>filesize(ffel)-1 then begin
                 j:=i;
                 i:=filesize(ffel)-1;
               end;
```

```
                    end;

                    CHANGEMENT(j,i,tab);
                end;
            end;

    until c=#13;


end;


begin
  curscache;

  seek(ffel,1);



  PECRAN1(tab,esc);
  if esc then mai:=true else mai:=false;
  i:=1;
  repeat

    GESTION_CLAVIER(i,pl,esc);

    if esc then exit;

    if (pl=0) and (filesize(ffel)>20) then begin
      PPRENDECR(true);
      c:=FERREUR_MAX_20_ELEVES;
      if c=#13 then begin
        esc:=true;
        exit;
      end
      else BEGIN
        esc:=false;
        PPRENDECR(false);
      end;
    end;


  until   ((filesize(ffel)<=20) and (pl=0))
       or (pl<>0);

    if pl=0 then begin
      trouver:=false;
      rech:=tab[1];
    end
    else begin
      rech:=tab[pl];

      assign(fper,'A:\NOM\'+tab[pl].nf+'.NOM');
      assign(fere,'A:\ERE\'+tab[pl].nf+'.ERE');
      assign(frev,'A:\REV\'+tab[pl].nf+'.REV');
```

```
        trouver:=true;
      end;
end;
```

2.5.5 Function FFIN

```
function  FFIN;

  type
       tnote =  array [1..40] of integer;
       cn    =  string;

  const
        part : array [1..4] of string =

        ('^]^[^Y^[^V^X^T^[^X^Y]T]Y]T]V[S[X[S[U[Q[U[Q[VYQYVY
          QYRXOXTXOXQVMVRVMVOULUQULUM',
         'QJQMQJQEQJQMQJQLQLQQQLQEQLQQQLQM',
         ']V]Y]V]Q]V]Y]V]X]X][]X]Q]X][]X]Y',
         ']V]X]U]V');


  var
       i,j,code,temp,long : integer;
       note               : tnote;
       c,ca               : char;
       boucle             : boolean;



procedure PLAY(CHAIN:cn);

  var i : integer;

   begin
     i:=1;
     repeat
       ca:=(chain[i]);
       boucle:=true;
       if ca='*' then begin
         long:=(byte(chain[i+1])-65)*7;
         boucle:=false;
       end;
       if ca='+' then begin
         temp:=(byte(chain[I+1])-65)*2;
         boucle:=false;
       end;
       if ca=' 'then begin
         dec(i);
         boucle:=false;
       end;
       if boucle=false then inc(i);
       code:=byte(ca);
       if code=65+32 then code:=65+31;
       dec(code,64);
```

```
        sound(note[code]);
        delay(long);
        if (ca<>'+') and (ca<>'*') then begin
          sound(0);
          sound(65535);
        end;
        delay(temp);
        inc(i);
      until (i>length(chain)) or keypressed;
    end;

PROCEDURE INIT;

    begin
      long:=100;temp:=100;

      note[1] :=174;
      note[2] :=186;
      note[3] :=196;
      note[4] :=208;
      note[5] :=220;
      note[6] :=234;
      note[7] :=246;
      note[8] :=261; (* do *)
      note[9] :=280;
      note[10]:=293;
      note[11]:=312;
      note[12]:=329;
      note[13]:=349;
      note[14]:=370;
      note[15]:=392;
      note[16]:=420;
      note[17]:=440;
      note[18]:=470;
      note[19]:=493;
      note[20]:=523; (* do *)
      note[21]:=555;
      note[22]:=587;
      note[23]:=630;
      note[24]:=659;
      note[25]:=698;
      note[26]:=744;
      note[27]:=784;
      note[28]:=846;
      note[29]:=880;
      note[30]:=940;
      note[31]:=987;
      note[32]:=1046; (* do dernière note*)
      note[33]:=1120;
      note[34]:=1174;

    END;
```

```
  begin
    contrlbreak:=false;

    PVBCD(1);
    clrscr;

    PCADRE(24,12,55,14,'',7,false);

    cursnormal;

    gotoxy(27,13);
    write('Continuer (');
    highvideo;
    write('O');
    lowvideo;
    write('ui ou ');
    highvideo;
    write('N');
    lowvideo;
    write('on) : ');

    if roptions.musique then begin

      init;
      play(' ');
      play('*T');
      i:=1;

      repeat
        play(part[i]);
        inc(i);
      until (i>2) or keypressed;
    end;

    cursnormal;
    cachecursdanslitcar:=false;
    affichecarlu:=true;
    FFIN:= not ouiounon(' ',13,51,$07,#0);
    curscache;
    contrlbreak:=true;

  end; {'-- END pfin --'}


procedure PBUFFC;

  VAR
      reg    : registers;
      touche : boolean;

  begin
    touche:=true;
    while touche do begin
      reg.ah:= 01;
      intr($16,reg);
```

```
        if  boolean (reg.flags and.$40) then touche:=false
        else begin
          reg.ah:=0;
          intr($16,reg);
          touche:=true;
        end;
      end;
    end;
```

2.5.6 Procedure PVBCD

```
procedure  PVBCD( d : word);
  begin
    delay(d);
    PBUFFC;
  end;
```

2.5.7 Procedure PSAISIE

```
procedure  PSAISIE(var entree : string15;
                       bs,x,y : byte;
                       cht    : boolean;
                   var esc    : boolean);

  var
        bi : byte;
        sor: boolean;

   begin
     bi:=x;
     sor:=esc;
     esc:=false;
     ecritvite(entree,y,x,7);
     cursbloc;

     repeat
       gotoxy(x,y);
       c:=readkey;
       if (bi+bs>=x) and (bi<=x) then begin

       case c of

     #65..#90,#97..#122 : if bi+bs>x then begin
                            inc(x);
                            insert(c,entree,x-bi);
                            ecritvite(entree,y,bi,7);
                            cursbloc;
                          end;

            #45,#39 : if bi+bs>x then begin
                            inc(x);
                            insert(c,entree,x-bi);
                            ecritvite(entree,y,bi,7);
                            cursbloc;
                          end;

          #128..#154 : if bi+bs>x then begin
                            inc(x);
                            insert(c,entree,x-bi);
                            ecritvite(entree,y,bi,7);
                            cursbloc;
                          end;

                #27 : if sor then begin
                            esc:=true;
                            c:=#13;
                          end;

              #48..#57 : if bi+bs>x then begin
                            inc(x);
                            insert(c,entree,x-bi);
```

```
                ecritvite(entree,y,bi,7);
                cursbloc;
              end;

            #8 : if x>=bi then begin

                delete(entree,x-bi,1);
                gotoxy(length(entree)+bi,y);

                if cht then
                  ecritvite(#32,coordyabs,coordxabs,7)
                else
                  ecritvite(#95,coordyabs,
                                coordxabs,7);

                if x>bi then dec(x);
                ecritvite(entree,y,bi,7);
                cursbloc;

              end;

            #0 : begin
                c:=readkey;
                case c of

                 #75 : if bi<x then dec(x)
                       else if bi=x then begin
                         sound(1000);
                         delay(100);
                         nosound;
                       end;

                 #77 : if (bi+bs>x) and
                       (bi+length(entree)>x)    then
inc(x);

                 #83 : if (bi<=x) then begin
                         delete(entree,x-bi+1,1);
                         gotoxy(bi,y);
                         if cht then
                         ecritvite(entree+#32,coordyabs
                                              coordxabs
                         else
                           ecritvite(entree+'_',
                           coordyabs,coordxabs,7);
                       end;

                 #79 : x:=length(entree)+bi;

                 #71 : x:=bi;

                end;
              end;
```

```
            end;
        end;

    until c=#13;
end;
```

2.5.8 Procedure PDICOEL

```
(* recherche dichotomique dans le fichier élève *)

procedure PDICOEL(var x  : tfel;        (* élève à rechercher *)
                  var bi : word;(* nr où il devrait se trouver*)
                  var tr : boolean);    (* true si trouvé *)

  var
       bs,y : word;
       z    : tfel;


   begin
     bi:=1;
     bs:=filesize(ffel)-1;
     while bi<bs do begin
       y:=(bi+bs) div 2;
       seek(ffel,y);
       read(ffel,z);
       case comparechaine(x.nom,z.nom) of

         Inferieur : bs:=y-1;

         Superieur : bi:=y+1;

               Egal : begin
                        bi:=y;
                        bs:=y;
                      end

       end;

     end;

     seek(ffel,bi);
     read(ffel,z);

     case comparechaine(x.nom,z.nom) of

       Inferieur : begin
                     tr:=false;
                   end;

       Superieur : begin
                     tr:=false;
                     inc(bi);
                   end;


            Egal : begin
                     tr:=true;
                     x:=z;
```

```
                    end;

  end;
end;
```

2.5.9 Procedure PPRENDECR

```
procedure PPRENDECR(pre    : boolean);  (* pre = true s'il faut
                                          enregistrer l'ecran et
                                             a false s'il faut
                                           rendre l'ecran     *)


procedure CGA_EGA_VGA;

  var
       ecr     : tecran absolute $B800:0000;

   begin
     if pre then begin
       getmem(ecran,sizeof(ecran));
       ecran^:=ecr;
     end
     else ecr:=ecran^;
   end;


procedure HERCULES;

  var
       ecr     : tecran absolute $B000:0000;

   begin
     if pre then begin
       getmem(ecran,sizeof(ecran));
       ecran^:=ecr;
     end
     else ecr:=ecran^;
   end;


   begin
    if HerculesPresent then HERCULES
     else CGA_EGA_VGA;
   end;
```

2.5.10 Procedure PINVERSE

```
procedure PINVERSE(inv : string80);
   begin
     attrtexte:=112;
     write(inv);
     attrtexte:=7;
   end;
```

2.5.11 Procedure PERREUR

```
(*************************************************************)
(* message d'erreur max 74 car et x,y position du message *)
(*************************************************************)


procedure PERREUR (erreur : terreur;    (* mess d'erreur        *)
                   x,y    : byte;       (* cood sup gauche      *)
                   s      : word);      (* valeur en Hz du son *)

  var
       k   : byte;


   begin

     curscache;

     ecritvite('╔'+chainecar(#205,length(erreur)+4)+'╗',y,x,135);

     ecritvite('╚'+chainecar(#205,length(erreur)+4)+'╝',y+2,x,135);

     ecritvite('║',y+1,x,135);

     ecritvite(' '+erreur+' ',y+1,x+2,112);

     ecritvite('║',y+1,x+length(erreur)+5,135);

     PVBCD(1);

     contrlbreak:=false;
     if roptions.bruitage then
       for k:=1 to 3 do begin
         sound(s);
         delay(200);
         nosound;
         delay(150);
       end;

     contrlbreak:=true;

     k:=0;
     repeat
       delay(334);
       inc(k);
     until keypressed or (k=20);

     cursnormal;
   end;
```

2.5.12 Procedure PCADRE

```
procedure PCADRE(x1,y1,x2,y2 : byte;  (* cood inf d et sup g *)
                 titre       : string50;(* titre du cadre    *)
                 coul        : byte;  (* couleur du cadre    *)
                 sd          : boolean );

  type
       tsod = array [1..8] of char;

  const
       csl : tsod = (#218,#196,#191,#179,#192,#217,#180,#195);
       cdl : tsod = (#201,#205,#187,#186,#200,#188,#185,#204);


  var   k  : byte;
        ti : boolean;
        sod      : tsod;


(*************** affichage du titre ***************************)

  procedure PTITRE;

    var
        i,lt,lgt : byte;

     begin
       lgt:=length(titre);
       lt:=(((x2-x1)-(lgt+6))div 2)+x1+1;
       ecritvite(sod[1]+chainecar(sod[2],lgt+4)+sod[3],y1-1,lt,coul);
       ecritvite(sod[7],y1,lt,coul);
       ecritvite('  '+titre+'  ',y1,lt+1,15);
       ecritvite(sod[8],y1,lt+lgt+5,coul);
       ecritvite(sod[5]+chainecar(#205,lgt+4)+sod[6],y1+1,lt,coul);
     end;



     begin
       if titre<>'' then ti:=true else ti:=false;
       if ti then coul:=7;
       curscache;
       if sd then sod:=csl else sod:=cdl;
       ecritvite(sod[1]+chainecar(sod[2],x2-(x1+1))+sod[3],y1,x1,coul
       for k:=y1+1 to y2-1 do begin
         ecritvite(sod[4],k,x1,coul);
         ecritvite(sod[4],k,x2,coul);
       end;
       ecritvite(sod[5]+chainecar(sod[2],x2-(x1+1))+sod[6],y2,x1,coul
       if ti then PTITRE;
     end;
```

2.5.13 Procedure PMENU

```
procedure PMENU(     choix   : tmenu;      (* titre + choix    *)
                     x1,y1   : byte;       (* cood sup gauche *)
                     x2,y2   : byte;       (* cood inf droit  *)
                     nbre    : byte;       (* nbre de  choix  *)
                 var choisi  : byte);      (* choix retenu     *)


   type
        tl = array [1..9] of record
                               c : char;
                               p : byte;
                             end;

   var
        gr,debut : byte;
        l        : tl;
        esc      : boolean;


procedure PG(var gr : byte;
             var l  : tl);

   var
        i,j,k    : byte;
        t        : boolean;

    begin
      gr:=length(choix[2]);
      l[1].c:=choix[2,1];
      l[1].p:=2;
      for i:=3 to nbre+1 do begin
        if gr<length(choix[i]) then gr:=length(choix[i]);
        j:=0;
        repeat
          inc(j);
          l[i-1].c:=choix[i,j];
          t:=false;
          for k:=1 to i-2 do
            if not t then
            if l[i-1].c=l[k].c then t:=true
            else t:=false;
        until not t;

        l[i-1].p:=pos(l[i-1].c,choix[i])+1;
      end;
      for i:=2 to nbre+1 do
        choix[i]:=complete(' '+choix[i],gr+2);
      gr:=40-((gr+2) div 2);
    end;
```

```
procedure PECRAN1(var debut : byte);

  var
       i : byte;
       x : string[85];


   begin
     debut:=((18-((nbre*2)+1))div 2)+5;

     delete(choix[2],l[1].p,1);
     ecritvite( copy(choix[2],l[1].p-1,l[1].p-1)
               +l[1].c
               +copy(choix[2],l[1].p,gr),debut,gr,112);

     for i:=3 to nbre+1 do begin
       delete(choix[i],l[i-1].p,1);
       gotoxy(gr,debut+((i-2)*2));
       write(copy(choix[i],1,l[i-1].p-1));
       highvideo;
       write(l[i-1].c);
       lowvideo;
       write(copy(choix[i],l[i-1].p,gr));
     end;

     ecritvite(#204+chainecar(#205,x2-x1-1)+#185,y2-2,x1,7);

     changeattr(x2-x1-3,y2-1,x1+2,112);

     ecritvite('BOUGER  :    '+#24+#25,y2-1,x1+4 ,112);

     if esc then begin
       ecritvite('CHOISIR :   '+#17+#196+#217,y2-1,x1+((x2-x1)
                 div 2 )-7,112);
       ecritvite('QUITTER :  [ESC]',y2-1,x2-19,112);
     end
     else ecritvite('CHOISIR :   '+#17+#196+#217,y2-1,x2-17,112);

   end;


procedure PGESTION_CLAVIER(    debut  : byte;
                           var choisi : byte);

  var
       bi,i,j : byte;
       c      : char;
       FL     : BOOLEAN ;
       ch     : string[9];


procedure PCHANGEMENT(j,i    : byte;
                      debut  : byte);
```

```
      begin
        gotoxy(gr,debut+((j-2)*2));
        write(copy(choix[j],1,l[j-1].p-1));
        highvideo;
        write(l[j-1].c);
        lowvideo;
        write(copy(choix[j],l[j-1].p,gr));

        changeattr(length(choix[2])+1,debut+((i-2)*2),gr,112);

      end;


      begin
        ch:='';
        for i:=1 to nbre do ch:=ch+upcase(l[i].c);
        i:=2;
        repeat
          c:=readkey;

          case c of

                #13 : begin
                        choisi:=i-1;
                        exit;
                      end;

                #27 : if esc then begin
                        choisi:=nbre;
                        exit;
                      end;


#65..#90,#97..#122 : BEGIN
                       c:=upcase(c);
                       choisi:=pos(c,ch);
                       if (choisi<=nbre) and (choisi>0) then
                         c:=#13
                     END;

                #0  : begin
                        c:=readkey;
                        case c of

                         #72 : begin
                                 if i=2 then begin
                                   j:=2;
                                   i:=nbre+1;
                                 end
                                 else begin
                                   j:=i;
                                   dec(i);
                                 end;
                                 PCHANGEMENT(j,i,debut);
                               end;
```

```
                        #80 : begin
                                if i=nbre+1 then begin
                                  j:=i;
                                  i:=2;
                                end
                                else begin
                                  j:=i;
                                  inc(i);
                                end;
                                PCHANGEMENT(j,i,debut);
                              end;

                        #71 : IF i<>2 THEN begin
                                j:=i;
                                i:=2;
                                PCHANGEMENT(j,i,debut);
                              end;

                        #79 : IF i<>nbre+1 THEN begin
                                j:=i;
                                i:=nbre+1;
                                PCHANGEMENT(j,i,debut);
                              end;
                    end;
                end;
            end;
        until c=#13;
    end;


    begin
      clrscr;
      if choisi=1 then esc:=true else esc:=false;
      if x1+x2+y1+y2=0 then begin
        x1:=5;
        y1:=2;
        x2:=74;
        y2:=24;
      end;
      PCADRE(x1,y1,x2,y2,choix[1],0,false);
      PG(gr,l);
      PECRAN1(debut);
      PVBCD(10);
      curscache;
      PGESTION_CLAVIER(debut,choisi);
      cursnormal;
    end;


{******************}
BEGIN
END.
```

2.6 Unit ELEVE

```
{F+,O+}
    unit eleve;


     interface


     uses
            dos,
            tpcrt,
            tpchaine,
            tpedit,
            tpint24,
            proc,
            def,
            proc2;


     procedure  PELEVES;


     implementation
```

2.6.1 Procedure PIDENTIF

(choix d'un élève dans la liste ou ajout dans la liste)

```
     procedure  PIDENTIF(var rech        : tfel;
                         var pl          : word;
                         var esc         : boolean);


       var
            trouver,esc1,esc2,tr : boolean;
            x                    : byte;
            p,y                  : word;
```

```
{ intro. du nom  }

  procedure PINTRO_NOM(var nom,prenom : string15;
                       var esc        : boolean);

    var
         esc2 : boolean;
         i    : byte ;
         n,p  : string;


  procedure PMAJUSCULE(var rep : string15);

   var
         j : byte;

    begin
      for j:=1 to length(rep) do
        case ord(rep[j]) of

         97..119 : rep[j]:=upcase(rep[j]);

130,136,137,138 : rep[j]:='E';

        131..133 : rep[j]:='A';

         139,140 : rep[j]:='I';

             135 : rep[j]:='C';

     129,150,151 : rep[j]:='U';

        147..149 : rep[j]:='O'

         end;
    end;



    begin
      cursnormal;
      forcemajusc:=true;

      repeat
        clrscr;
        PCADRE(2,2,79,24,'Bonjour . Comment t''appelles-tu
            ?',0,false);
        PCADRE(10,7,70,19,'',7,true);
        ecritvite('Ecris ton prénom (ex. Jean ou Marie) :'
                  ,9,15,7);
        ecritvite('Continuer : '+#17+#196+#217,12,50,7);
        ecritvite(#195+chainecar(#196,59)+#180,13,10,7);
        ecritvite('Ecris ton nom (celui de ta famille) : '
```

```
            ,15,15,7);
    ecritvite('Continuer : '+#17+#196+#217,18,50,7);

    repeat;
      ecritvite(complete(prenom,16),11,20,112);
      ecritvite(complete(nom,16),17,20,112);
      attrtexte:=7;

      repeat
        p:=prenom;
        litchaine('',11,20,15,$00,$70,$00,esc,p);
        prenom:=copy(p,1,15);
        PMAJUSCULE(prenom);
        p:='';
        if esc then begin
          attrtexte:=7;
          exit;
        end;
      until prenom<>'';


      repeat
        n:=nom;
        litchaine('',17,20,15,$00,$70,$00,esc,n);
        nom:=copy(n,1,15);
        PMAJUSCULE(nom);
      until (nom<>'') or esc;

    until not esc;

    if not esc then begin
      clrscr;
      PCADRE(18,7,62,16,'Vérification',0,false);
      ecritvite('PRENOM       : ',10,24,7);
      ecritvite(prenom,10,40,15);
      ecritvite('NOM          : ',12,24,7);
      ecritvite(nom,12,40,15);
      ecritvite('Est-ce correct (',14,23,7);
      ecritvite('O',14,39,15);
      ecritvite('ui ou ',14,40,7);
      ecritvite('N',14,46,15);
      ecritvite('on) ? : ',14,47,7);
      cursnormal;
      affichecarlu:=true;
      cachecursdanslitcar:=false;
      litcar(' ',14,54,$07,['O','N'],c);
      curscache;
    end
    else c:='N';

  until (c='O');

end;
```

```
  procedure PFEL(     rech : tfel;
                      pl   : word);

    var
         i      : integer;
         tr     : boolean;
         rech1  : tfel;


{ création de fel.dat et des diff répertoires }

  procedure PECRI(    rech : tfel;
                      pl   : word);
    var
         i    : byte;
         per  : tper;
         ere  : tere;
         rev  : trev;
         che  : string[50];

     begin

       seek(ffel,pl);
       write(ffel,rech);

       che:='a:\nom\'+rech.nf+'.NOM';
       assign(fper,che);
       rewrite(fper);
       fillchar(per,sizeof(tper),0);
       write(fper,per);
       close (fper);


       assign(fere,'a:\ere\'+rech.nf+'.ERE');
       rewrite(fere);
       fillchar(ere,sizeof(tere),0);
       write(fere,ere);
       close (fere);

       assign(frev,'a:\rev\'+rech.nf+'.REV');
       rewrite(frev);
       for i:=1 to 50 DO REV.rev[i]:=FALSE;
       write(frev,rev);
       close (frev);

     end;



     begin
       if filesize(ffel)>1 then begin
         for i:=filesize(ffel)-1 downto pl do begin
           seek (ffel,i    );
           read (ffel,rech1);
           seek (ffel,i+1  );
```

```
      write(ffel,rech1);
    end;
  end;
  PECRI(rech,pl);
end;


begin

  curscache;
  PIOFEL;

  repeat
    esc2:=false;
    rech.nom:='';
    esc1:=false;
    if (filesize(ffel)>1) then begin
      clrscr;
      PLIST_NOM(rech,pl,trouver,esc1)
    end
    else begin
      trouver:=false;
      esc1:=false;
    end;

    if esc1 then begin
      esc:=true;
      exit;
    end;

    if not trouver then begin
      rech.nom:='';
      nom:='';
      prenom:='';
      PINTRO_NOM(nom,prenom,esc2);
      if not esc2 then begin
        if filesize(ffel)>1 then begin
          rech.nom:=complete(nom,15)+complete(prenom,15);
          PDICOEL(rech,pl,tr);
        end
        else begin
          tr:=false;
          pl:=1;
        end;
        if not tr then begin
          if not esc2 then begin
            rech.nom:=complete(nom,15)+complete(prenom,15);
            rech.nf :=copy(nom,1,4)+copy(prenom,1,4);
            rech.num:=0;
            rech.rev:=0;
            PFEL(rech,pl);
          end
        end
        else begin
```

```
            assign(fper,'A:\NOM\'+rech.nf+'.NOM');
            assign(fere,'A:\ERE\'+rech.nf+'.ERE');
            assign(frev,'A:\REV\'+rech.nf+'.REV');
          end;
        end;
      end;

    until   ((filesize(ffel)=1) and (esc2))
         or ((rech.nom<>'')and not esc2);

    if ((filesize(ffel)=1) and (esc2)) then begin
      esc:=true;
      pl:=0;
    end
    else esc:=false;

    close(ffel);

  end;
```

2.6.2 Procedure PELEVES

```
{ gestion des exercices }

  procedure  PELEVES;


    var
         pl,i,y              : word;
         esc,esc1,esc2       : boolean;
         tr,fin,exf          : boolean;
         tabmot              : ttabmot;
         dis2                : integer;



     begin

       repeat
         pidentif(rech,pl,esc1);
         fin:=false;

         if esc1 then begin
           while (''=FSearch('VBR.EXE','')) and
                 (''=FSearch('C:\LANGAGES\PASCAL\EXE&TPU\VBR.EXE','')
                  do begin
             clrscr;
             ecritvite('Introduire la disquette n° 1',12,25,112);
             c:=readkey;
           end;
           exit;
         end;

         esc2:=false;
         if (rech.num<100) or (rech.rev<10) then begin
           while (''=FSearch('DAT\TEXTES.DAT','')) do
                  begin
             clrscr;
             ecritvite('Introduire la disquette n° 2',12,25,112);
             c:=readkey;
           end;

           repeat

             if (rech.rev<(rech.num div 10)) then begin
               prevision(rech,pl,esc2,fin,exf);
               if not fin  and (rech.rev<10) and exf then
                 fin:=FFIN;
             end
             else
             if not esc2 then
```

```
            if not odd(rech.num) then begin

              i:=(rech.num div 2);

              ptextes(i+1,esc2);

              if not esc2 then begin
                reset(ffel);
                inc(rech.num);
                seek(ffel,pl);
                write(ffel,rech);
                reset(ffel);
                inc(i);
                fin:=FFIN;
              end;

            end
            else begin

              i:=(rech.num div 2);

              pphrases(i+1,esc2);

              if not esc2 then begin
                reset(ffel);
                inc(rech.num);
                seek(ffel,pl);
                write(ffel,rech);
                reset(ffel);
                inc(i);
                fin:=FFIN;
              end;

            end;

        until ((rech.num=100) and (rech.rev=10)) or esc2 or
          fin;
      end;

    until esc1;


end;


begin
end.
```

2.7 Unite proc2

```
unit  proc2;

   {   ensemble des exercices et de leurs procédures  }

interface

   uses
           tport,
           dos,
           tpdos,
           tpedit,
           printer,
           tpchaine,
           proc,
           def;


 ( exercice sur les texte )

     procedure PTEXTES(     num : byte; ( numero du texte )
                        var esc : boolean ); ( esc o/n )


 ( exercice sur les phrases )

     procedure PPHRASES(     num : byte; ( numero de la serie de
                                           phrases )
                         var esc : boolean); ( esc o/n )


 ( exercice de revision )

     Procedure PREVISION(var rech          : tfel; ( nom de l'élève )
                             pl            : word; ( place dans le
                                                     fichier élèves)
                         var esc,fin,exf : boolean); ( esc, fin des
                                                       exercices exf
                                                       exercice fini
                                                       o/n)
```

implementation

2.7.1 Procedure PANA

```
      procedure PANA(    rep    : string15; ( réponse de l'élève )
                         mot    : tmots;  ( mot , sque, adr f_b  )
                    var frape   : boolean; ( faute de frappe o/n  )
                    var ortho   : boolean; ( faute ortho    o/n  )
                    var loer    :  tloer;  ( découpage des diff  zones
fautives )
                    var tabmot  : ttabmot; ( découpage du sque )
                    var max     : byte);   ( nombre de découpages)


        type
             tmaxp   = array [1..15] of byte;

        var
             maxp : tmaxp;

(*-------------------------------------------------------------------------
(*----------P A N A M O T (ANALYSE DU SQUE)   ---------------------*)
(*-------------------------------------------------------------------------


      procedure PANAMOT(    amot   : string60; ( squelette )
                        var tabmot: ttabmot; ( découpage du squelette)
                        var max   : byte;    ( nbre de découpages )
                        var maxp  : tmaxp);  ( nbre de possibilités
                                               par découpage)

        var
             i,j : byte;

         begin

           for i:=1 to 15 do begin
             maxp[i]:=1;
             for j:=1 to 10 do tabmot[i,j]:='';
           end;

           max:=1;
           for i:=1 to length(amot) do
             case amot[i] of

               #39,
               #97..#122,
               #130..#140,
               #147..#151   : tabmot[max,maxp[max]]:=
                                tabmot[max,maxp[max]]+amot[i];
```

```
                        ',' : maxp[max]:=maxp[max]+1;

                        '/' : inc(max);

            end;

          maxp[max]:=0;
          dec(max);

        end; (*-- END panamot --*)


(*-------------------------------------------------------------------------
(*----------------------------P  A  N   A  R   E  P     (ANALYSE   REP)
------------------*)
(*-------------------------------------------------------------------------


      procedure PANAREP(   rep,mot: string15;

                           tabmot: ttabmot;
                           max   : byte;
                           maxpe : tmaxp;
                       var ortho : boolean;
                       var frape : boolean;
                       var loer  : tloer);

        type
             tfr = array [1..19] of char;

        const
             fr : tfr =(#97,#101,#105,#111,#117,#121,#129,#131,#132,
                        #133,#135,#136,#137,#139,#140,#147,#148,#150,#151

        var
             j   : array [1..15] of byte;
             i,k : byte;


         begin

           frape:=false;
           ortho:=false;


 ( cherche un  triplement d'une  lettre; si  oui, faute  de frappe  et
sortie )

           for i:=1 to 26 do
             if pos(chr(96+i)+chr(96+i)+chr(96+i),rep)>0 then begin
               frape:=true;
               exit;
             end;

           mot:='';
```

```
    for i:=1 to 19 do
      if (pos(fr[i]+fr[i],rep)>0) and
         (pos(fr[i]+fr[i],mot)=0)      then begin
        frape:=true;
        exit;
      end;


    i:=length(mot);
    k:=length(rep);


{ recherche si rep est trop long / à la rep }

    if ((i=1) and  (k>i+2)) or
       ((i=2) and  (k>i+2)) or
       ((i=3) and  (k>i+2)) or

       ((i=4) and  (k>i+3)) or
       ((i=5) and  (k>i+3)) or
       ((i=6) and  (k>i+3)) or

       ((i>=7) and  (k>i+4))  then begin

      ortho:=false;
      exit;
    end;


    for i:=1 to 15 do j[i]:=1;

    repeat
      repeat
        repeat
          repeat
            repeat
              repeat
                repeat
                  repeat
                    repeat
                      repeat
                        repeat
                          repeat
                            repeat
                              repeat
                                repeat
                                  repeat

                                    mot:='';

                                  { crea de la combi }

                                    for i:=1 to max do
                                      mot:=mot+tabmot[i,j[i]];

                                  { si combi et dans rep }
```

```
                    if (pos(mot,rep)>0) and
                       (pos(mot,rep)<=2) then
                       begin

{ regarde les diff parties fautives du mot }

                       for i:=1 to max do
                         if j[i]>1 then begin
                           ortho:=true;
                           loer[i]:=true
                         end
                         else loer[i]:=false;

                       if length(mot) <
                          length(rep) then
                         if pos(mot,rep)>1 then
                            begin
                           loer[1]:=true;
                           ortho:=true;
                           if
                        pos(mot,rep)+length(mot)
                              <length(rep)+1 then
                              loer[max]:=true;
                         end
                         else begin
                           loer[max]:=true;
                           ortho:=true
                         end;

                     end;

                     if ortho then exit;

                     inc(j[15]);

                   until (j[15]>maxp[15]);
                   j[15]:=1;
                   inc(j[14]);
                 until (j[14]>maxp[14]);
                 j[14]:=1;
                 inc(j[13]);
               until (j[13]>maxp[13]);
               j[13]:=1;
               inc(j[12]);
             until (j[12]>maxp[12]);
             j[12]:=1;
             inc(j[11]);
           until (j[11]>maxp[11]);
           j[11]:=1;
           inc(j[10]);
         until (j[10]>maxp[10]);
         j[10]:=1;
         inc(j[9]);
       until (j[9]>maxp[9]);
       j[9]:=1;
```

```
                    inc(j[8]);
                  until (j[8]>maxp[8]);
                  j[8]:=1;
                  inc(j[7]);
                until (j[7]>maxp[7]);
                j[7]:=1;
                inc(j[6]);
              until (j[6]>maxp[6]);
              j[6]:=1;
              inc(j[5]);
            until (j[5]>maxp[5]);
            j[5]:=1;
            inc(j[4]);
          until (j[4]>maxp[4]);
          j[4]:=1;
          inc(j[3]);
        until (j[3]>maxp[3]);
        j[3]:=1;
        inc(j[2]);
      until (j[2]>maxp[2]);
      j[2]:=1;
      inc(j[1]);
    until (j[1]>maxp[1]);

  end;   (*-- END  panarep --*)


  begin

    PANAMOT(mot.sque,tabmot,max,maxp);
    PANAREP(rep,tabmot,max,maxp,ortho,frape,loer);

  end;  (*-- de pana --*)
```

2.7.2 Procedure PPOINTS

```
{  affichage des points de l'elève }

procedure PPOINTS(j : real;
                  i : byte);

  begin

    if i=0 then PCADRE(60,3,76,5,'',7,false);
    gotoxy(64,4);
    highvideo;
    write(j:3:1,' / ',i);
    lowvideo;

  end; (* fin ppoints *)
```

2.7.3 Function FDATE_HEURE_TRAV

```
{  enreg de la date et de l'heure de début du travail }

   function FDATE_HEURE_TRAV : longint;
     var
          dh : datetime;
          x  : word;
          y  : longint;

      begin
        with dh do begin
          getdate(year,month,day,x);
          gettime(hour,min,sec,x);
        end;
        packtime(dh,y);
        FDATE_HEURE_TRAV:=y;
      end;  (* fin pdate heure trav *)
```

2.7.4 Procedure PF_B

```
{  gestion des feed back }

  procedure PF_B(     mot    : tmots;
                      loer   : tloer;
                      max,i  : byte);


    type tph     = array [1..3] of string60;
         tmo     = string[5];
         tpo     = array [1..2,1..2] of byte;

    var  j,nber,k,cp: byte;
         ind        : tind;
         bfb,esc,hv : boolean;
         mo         : tmo;
         po         : tpo;
         ph         : tph;
         rep        : string15;


{ préparation du feed back }

   procedure PFREP(var ph  : tph;
                   var mo  : tmo;
                   var po  : tpo;
                   var ind : tind);


     var
          i,j,k : byte;
          t     : string60;
          fl    : boolean;


      begin
        fl:=false;
        i:=0;
        seek(ff_b,ind.deb);
        for k:=1 to ind.nbre do read(ff_b,ph[k]);
        for k:=1 to ind.nbre do begin
          if pos('(',ph[K])<>0 then begin
            repeat
              if not fl then begin
                inc(i);
                j:=pos('(',ph[K]);
                delete(ph[K],j,1);
                po[i,2]:=k;
                po[i,1]:=j;
                repeat
                  mo:=mo+ph[k,j];
                  inc(j);
                until ph[K,j]=')';
```

```
            delete(ph[k],j,1);
            fl:=true;
          end
          else begin
            inc(i);
            j:=pos('(',ph[K]);
            delete(ph[K],j,1);
            po[2,2]:=k;
            po[2,1]:=j;
            repeat
              delete(ph[k],j,1);
            until ph[K,j]=')';
            delete(ph[k],j,1);
          end;
        until pos(')',ph[k])=0;
      end;
    end;
    insert(chainecar('_',length(mo)),ph[po[2,2]],po[2,1]);
    delete(ph[ind.nbre],pos('*',ph[ind.nbre]),10);
  end;  {'-- de pprep  --'}



  begin
    if i>2 then hv:=true else hv:=false;
    nber:=i;
    for i:=1 to 5 do begin
      gotoxy(12,18+i);
      clreol;
    end;
    PCADRE(5,18,75,24,'',112,false);
    ecritvite('╣ COMPLETER LE MOT       ENSUITE :  '
              +#17+#196+#217+' ╠',24,20,112);
    reset(findf_b);
    reset(ff_b);
    i:=0;
    repeat
      bfb:=false;
      k:=0;
      repeat
        inc(i);
        if loer[i] then bfb:=true;
      until (i=max) or (mot.adr[i]<>mot.adr[i+1]);
      curscache;
      if bfb then begin
        seek(findf_b,mot.adr[i]-1);
        read(findf_b,ind);
        mo:='';
        PPREP(ph,mo,po,ind);
        for j:=1 to 2 do
          ecritvite(chainecar(#32,60),20+j,10,7);
        for j:=1 to ind.nbre do ecritvite(ph[j],20+j,10,7);
        repeat
          if hv then ecritvite(mo,20+po[1,2],9+po[1,1],112)
```

```
            else ecritvite(mo,20+po[1,2],9+po[1,1],7);
            cursbloc;
            rep:='';
            PSAISIE(rep,length(mo),9+po[2,1],20+po[2,2],false,esc);
            rep:=copy(chnminusc(rep),1,15);
            inc(k);
            if rep<>mo then
              ecritvite(chainecar(#95,length(mo)),20+po[2,2],9+po[2,
          until (rep=mo) or (k=4);
        end;
       if k=4 then begin
          ecritvite(mo,20+po[2,2],9+po[2,1],7);
          curscache;
          cp:=0;
          if nber<4 then
            repeat
              delay(333);
              inc(cp);
            until (cp=10) or keypressed;
        end;
      until (i=max);
      attrtexte:=7;
      close(ff_b);
      close(findf_b);
   end; (* fin p_fb *)
```

2.7.5 Procedure PMAJ

```
{ liste en majuscules des mots étudiés }

   procedure PMAJ(var mom : tmom);

     var
          i,j : byte;


   procedure QUICKSORT(var a      : tmom;
                           Lo,Hi  : integer);

    procedure TRIE(l,r: integer);

     var
          i,j : integer;
          x,y : trmom;

      begin
        i:=l;
        j:=r;
        x:=a[(l+r) div 2];
        repeat
          while a[i].mom < x.mom    do inc(i);
          while x.mom    < a[j].mom do dec(j);
```

```
      if i<=j then begin
        y:=a[i];
        a[i]:=a[j];
        a[j]:=y;
        inc(i);
        dec(j);
      end;
    until i>j;
    if l<j then TRIE(l,j);
    if i<r then TRIE(i,r);
end;

begin {quicksort};

  TRIE(Lo,Hi);

end; (* fin quicksort *)



begin
  for i:=1 to 10 do
    for j:=1 to length(mom[i].mom) do
      case mom[i].mom[j] of

                #97..#122 :
          mom[i].mom[j]:=upcase(mom[i].mom[j]);

          #130,#136..#138 : mom[i].mom[j]:='E';

               #131..#133 : mom[i].mom[j]:='A';

                #139,#140 : mom[i].mom[j]:='I';

                     #135 : mom[i].mom[j]:='C';

           #129,#150,#151 : mom[i].mom[j]:='U';

               #147..#149 : mom[i].mom[j]:='O';

        end;

  QUICKSORT(mom,1,10);

end; (* fin maj *)
```

2.7.6 Procedure PERRSENS

```
{ message d'erreur de sens  selon le nb de fautes }

procedure PERRSENS(i : byte);

  var
       erreur : string[40];

  begin
    case i of

      1,2 : erreur:='CE N''EST PAS LE MOT ATTENDU';

      3,4 : erreur:='CE N''EST PAS ENCORE LE MOT ATTENDU';

    5,6,7 : erreur:='CE N''EST TOUJOURS PAS LE MOT ATTENDU';

    end;
    PERREUR(erreur,40-(length(erreur) div 2),20,131);
  end;
```

2.7.7 Procedure PERREURORTHO

```
{  message erreur d'orthographe  selon le nb de fautes }

  procedure PERRORTHO(i : byte);

    var
         erreur : string[40];

    begin
      case i of

        1,2 : erreur:='CE MOT EST MAL ORTHOGRAPHIE';

          3 : erreur:='CE MOT EST ENCORE MAL ORTHOGRAPHIE';

          4 : erreur:='CE MOT EST TOUJOURS MAL ORTHOGRAPHIE';

      end;
      PERREUR(erreur,40-(length(erreur) div 2),20,131);
    end;
```

2.7.8 Procedure PSENSAFF

```
{ aff du mess dans le cas d'une erreur de sens }

  procedure PSENSAFF(i : byte);

    begin
```

```
  PVBCD(10);
  clrscr;
  case i of

    1 : begin
          PCADRE(27,12,54,14,'',7,false);
          curscache;
          ecritvite('Ecris un autre mot.',13,32,7);
        end;

    2 : begin
          PCADRE(12,12,69,14,'',7,false);
          curscache;
          ecritvite('Tu as 10 secondes pour lire la LISTE'+
                    ' qui va suivre.',13,16,7);
         end;

    3 : begin
          PCADRE(12,12,69,14,'',7,false);
          curscache;
          ecritvite('Tu as 20 secondes pour lire la LISTE'+
                    ' qui va suivre.',13,16,7);
        end;

    4 : begin
          PCADRE(17,12,64,14,'',7,false);
          curscache;
          ecritvite('Tu as 5 secondes pour choisir entre 2
          mots.',13,20,7);
        end;

    5 : begin
          PCADRE(21,12,58,14,'',7,false);
          curscache;
          ecritvite('Recopie le mot qui va suivre.'
                    ,13,25,7);
        end;
  end;
  PVBCD(1);
  i:=0;
  repeat
    inc(i);
    delay(333);
  until (i=20) or keypressed;
end;
```

2.7.9 Procedure PAEFLISTE

```
{ aff de la liste des mots attendus }

  procedure PAFFLISTE(i,k : byte;
                      mom : tmom);

    var
        j : byte;


     begin
       clrscr;
       if (i=2) or (i=3) then begin
         PCADRE(25,6,55,22,'Liste des mots',7,false);
         curscache;
         for j:=1 to 10 do ecritvite(mom[j].mom,8+j,35,7);
         ecritvite('╠'+chainecar(#205,29)+'╣',20,25,7);
         ecritvite(chainecar(#32,27),21,27,112);
         ecritvite('CONTINUER :  '+#17+#196+#217,21,37,112);
         PVBCD(10);
         j:=1;
         c:=#65;
         case i of

           2 : repeat
                 delay(350);
                 inc(j);
                 if keypressed then c:=readkey;
               until (c=#13) or (j=15);

           3 : repeat
                 delay(350);
                 inc(j);
                 if keypressed then c:=readkey;
               until (c=#13) or (j=29);
         end;

       end
       else begin
         if i<>5 then PCADRE(25,9,55,18,'Liste des mots',7,false)
           else PCADRE(25,9,55,18,'',7,false);
         ecritvite('╠'+chainecar(#205,29)+'╣',16,25,7);
         ecritvite(chainecar(#32,27),17,27,112);
         ecritvite('CONTINUER :  '+#17+#196+#217,17,37,112);
         curscache;
         j:=0;
         repeat
           inc(j)
         until mom[j].pl=k;
         ecritvite(mom[j].mom,12,35,7);
         if i=4 then begin
           if (j=1) or (j=2) then j:=10;
           ecritvite(mom[(j div 2)+1].mom,14,35,7);
```

```
    end;
    j:=0;
    c:=#65;
    PVBCD(10);

    case i of

      4 : repeat
            delay(350);
            if keypressed then c:=readkey;
            inc(j);
          until (j>10) or (c=#13);

      5 : repeat
            delay(333);
            if keypressed then c:=readkey;
            inc(j);
          until(j=10) or (c=#13);

    end;
  end;
end;
```

2.7.10 Procedure PBONTEXTE

```
{ aff du mess quand une bonne rep est donnée dans un texte }

    procedure PBONTEXTE(pec : real);
      var
           i : byte;

       begin
         case round(pec*10) of

           20 : ecritvite('Très bien,
                    '+effespfin(copy(rech.nom,16,30))+
                          '.                ',24,6,240);

           15 : ecritvite('Bien, '+effespfin(copy(rech.nom,16,30))+
                          '.              ',24,6,240);

           10 : ecritvite('C''est bon.             ',24,6,240);

            5 : ecritvite('Enfin, tu as trouvé.     ',24,6,240);

         end;
         PVBCD(1);
         i:=0;
         repeat
           inc(i);
           delay(333);
         until (i=10) or keypressed;
         ecritvite('ECRIS TA REPONSE            ',24,6,112);
       end;
```

2.7.11 Procedure PBONPHRASE

```
{ idm mais pour les phrases }

    procedure PBONPHRASE(pec : real);
      var
           i : byte;

       begin
         case round(pec*10) of

           20 : ecritvite('Très bien,
                 '+effespfin(copy(rech.nom,16,30))+
                          '.             ',15,9,240);

           15 : ecritvite('Bien, '+effespfin(copy(rech.nom,16,30))+
                          '.          ',15,9,240);

           10 : ecritvite('C''est bon.             ',15,9,240);

            5 : ecritvite('Enfin, tu as trouvé.   ',15,9,240);
```

```
  end;
  PVBCD(1);
  i:=0;
  repeat
    inc(i);
    delay(333);
  until (i=10) or keypressed;
  ecritvite('ECRIS TA REPONSE          ',15,9,112);
end;
```

2.7.12 Procedure PAIDE

```
{ écran d'aide pour les premiers textes et phrases }

  procedure PAIDE(var esc : boolean);
    var
         x   : byte;
         pre : string[15];

     begin
       clrscr;
       PCADRE(10,5,70,22,'',7,false);
       pre:=effespfin(copy(rech.nom,16,30));
       x:=length(pre);

       if esc then begin
         ecritvite(pre,10,23-(x div 2),15);
         ecritvite(', lis bien le texte qui va suivre',10,
                     23+(x div 2)+(x mod 2),7);
       end
       else begin
         ecritvite(pre,10,21-(x div 2),15);
         ecritvite(', lis bien les phrases qui vont suivre',10,
                       21+(x div 2)+(x mod 2),7);
       end;
       ecritvite('car 10 mots vont disparaître',13,25,7);
       ecritvite('et tu devras les retrouver.',16,26,7);
       ecritvite('╠'+chainecar(#205,59)+'╣',20,10,7);
       ecritvite(chainecar(#32,57),21,12,112);
       ecritvite('CONTINUER :  '+#17+#196+#217,21,51,112);
       ecritvite('QUITTER : [ESC]',21,14,112);
       curscache;
       PVBCD(10);
       cachecursdanslitcar:=true;
       affichecarlu:=false;
       litcar('',1,1,$00,[#13,#27],c);
       if c=#27 then esc:=true else esc:=false;
     end;


  {$I ITEX}

  {$I IPHR}

  {$I IREV}


 begin
 end.
```

2.8 Include ITEX

```
(*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*)
(* ========================   TEXTES   ============================== *)
(*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*)

    procedure ptextes(     num : byte;
                       var esc : boolean );


      var
           lectu                  : tind2;
           trav                   : ttrav;
```

2.8.1 Procedure PTRAV

```
    procedure PTRAV(var trav  : ttrav);


      var
           i,j   : byte;
           test  : string[60];
```

2.8.1.1 Procedure PPREP

```
  { préparation du texte }

    procedure PPREP(var tex  : ttex;
                    var mo   : tmo;
                    var po   : tpo;
                        ind  : tind2);


      var
           i,j,k              : byte;
           t                  : string60;

       begin

         reset(ftextes);
         seek(ftextes,ind.deb-1);

         for k:=1 to ind.nbre do begin
           read(ftextes,t);
           tex[k]:=t;
         end;
```

```
  close(ftextes);

  for i:=1 to 10 do mo[i].mots:='';

  i:=0;
  for k:=2 to ind.nbre do begin
    if pos('(',tex[K])<>0 then begin
      repeat
        inc(i);
        j:=pos('(',tex[K]);
        delete(tex[K],j,1);
        po[i,2]:=k;
        po[i,1]:=j;
        repeat
          mo[i].mots:=mo[i].mots+tex[k,j];
          inc(j);
        until tex[K,j]=')';
        delete(tex[k],j,1);
      until pos('(',tex[k])=0;
    end;
  end;


    for i:=1 to 10 do begin
      trav.mot[i]:=mo[i].mots;
      trav.mom[i].mom:=mo[i].mots;
      trav.mom[i].pl:=i;
    end;


    reset(fmots);

    for i:=1 to 10 do begin
      seek(fmots,ind.lmo[i]);
      read(fmots,mo[i]);
    end;

    close(fmots);

    PMAJ(trav.mom);


end;  (* fin pprep *)
```

```
begin

  with trav do begin

    pprep(tex,mo,po,lectu);

    max:=length(mo[1].mots);
    for i:=2 to 10 do
      if max<length(mo[i].mots) then max:=length(mo[i].mots);

    inc(max,3);

    delete(tex[1],1,1);

  end;

end; (* fin ptrav *)
```

```
(*====================  travail de l'el tex. ====================*)
```

2.8.2 Procedure PELEV

```
{ partie affichage et exercice }

   procedure PELEV(    trav  : ttrav;
                   var esc   : boolean);


     var
          i,j,k,maxm,m,dbl : byte;
          rep,tro,saisie   : string15;
          ere              : tere;
          fautes           : array [1..10,1..6] of tere;
          ortho,frape      : boolean;
          loer             : tloer;
          l                : real;
          eleve            : tper;
```

2.8.2.1 Procedure PTEST

```
   procedure PTEST;

     var
          i,j,m : byte;
          pec   : real;

(*-------------------------------------------------------------------------*)

   procedure pfrappe;

      begin

        PERREUR('ERREUR DE FRAPPE',32,20,1000);
        PPRENDECR(false);
        cursbloc;

      end;


(*-------------------------------------------------------------------------*)

   procedure portho(var pec : real);

     var
          m,x              : byte;
          modeorigine,sup  : integer;
```

```
procedure mise_en_evidence(mo : tmots);

  var
       i,j,k : byte;
       evid  : boolean;

   begin

     k:=0;
     i:=0;
     write(' ');
     repeat
       evid:=false;
       repeat
         inc(i);
         if loer[i] then evid:=true;
       until (i=maxm) or (mo.adr[i]<>mo.adr[i+1]);

       for j:=k+1 to i do
         if evid then pinverse(tabmot[j,1])
         else write(tabmot[j,1]);

       k:=i
     until i=maxm;

   end;


   begin
     inc(j);
     if j=5 then exit;
     if pec>0 then pec:=pec - 0.5;
     with trav do begin
       if j<4 then fautes[i,j]:=complete(chr(i)+rep,20);
       perrortho(j);
       PVBCD(10);
       clrscr;
       case j of

         1 : begin
               PCADRE(15,12,65,14,'',7,false);
               curscache;
               ecritvite('Essaie encore une fois, '+
                         effespfin(copy(rech.nom,15,30))+'.',13,24,

             end;

         2 : begin
               PCADRE(15,12,65,14,'',7,false);
               curscache;
               ecritvite('Regarde bien le mot qui va
                          suivre.',13,24,7);
             end;

         3 : begin
```

```
            PCADRE(15,12,65,14,'',7,false);
            curscache;
            ecritvite('Regarde encore le mot qui va
                       suivre.',13,24,7);
          end;

      4 : begin
            PCADRE(15,10,65,16,'',7,false);
            curscache;
            ecritvite('Regarde pour la dernière
                       fois,',12,24,7);
            ecritvite('et aussi longtemps que tu
                      veux',13,24,7);
            ecritvite('le mot qui va suivre.'
                      ,14,24,7);
          end;
    end;

    x:=0;
    repeat
    delay(350);
    inc(x)
    until (x=14) or keypressed;


    if (j>1) and (j<5) then begin

      modeorigine:=lo(derniermode);
      modetexte(0);
      if herculespresent then sup:=20 else sup:=0;

      clrscr;
      mo[i].mots:=effblancs(mo[i].mots);

      x:=length(mo[i].mots);
      PCADRE(18+sup-(x div 2),12,21+sup+(x div 2)+(x mod
      2),14,'',7,false);

      curscache;
      gotoxy(19+sup-(x div 2),13);
      if (j=2) or (j=4) then write(' '+mo[i].mots+' ')
      else mise_en_evidence(mo[i]);

      if j<>4 then delay(2000) else begin
        PVBCD(10);
        c:=readkey;
      end;

      modetexte(modeorigine);
    end;

    PPRENDECR(false);
    cursbloc;

end;
```

```
        end;

(*-------------------------------------------------------------------------

      procedure psens(var k   : byte;
                      var pec : real);

        var
            m : byte;


         begin

           with trav do begin

             inc(k);

             if k<4 then fautes[i,3+k]:=complete(chr(i)+'*'+rep,20);

             if (k=2) and (pec>0) then pec:=pec-0.5
             else if (k=4) and (pec>0) then pec:=pec-0.5
               else if (k=5) and (pec>0) then pec:=pec-0.5;

             perrsens(k);

             if k=6 then begin
               PPRENDECR(false);
               exit;
             end;

             psensaff(k);

             if (k>1) then begin

               paffliste(k,i,mom);

               ppoints(0,0);
               ppoints(1,i);
               cursbloc;
             end;

             PPRENDECR(false);

             cursbloc;

           end;
         end;

(*-------------------------------------------------------------------------

         begin
```

```
with trav do begin
  for i:=1 to 10 do begin
    j:=0;
    k:=0;
    pec:=2;
    repeat
      esc:=false;

      repeat
        saisie:='';
        psaisie(saisie,max,4+po[i,1],5+po[i,2]+dbl,false,esc);
      until saisie<>'';

      rep:=copy(chnminusc(saisie),1,15);

      if rep<>mo[i].mots then begin

        PPRENDECR(true);
        ecritvite(chainecar(#32,20),24,6,112);
        ortho:=false;

        curscache;

        PANA(rep,mo[i],frape,ortho,loer,tabmot,maxm);

        if frape then pfrappe
                 else if ortho then portho(pec)
                               else psens(k,pec);

        ecritvite(tro,5+po[i,2]+dbl,4+po[i,1],7);

      end
      else begin
        if roptions.bruitage then begin
          sound(220);
          delay(150);
          nosound;
          delay(100);
          sound(175);
          delay(150);
          nosound;
        end;
        l:=l+pec;
        curscache;
        ppoints(l,i*2);
        ecritvite(centre(mot[i],max),5+po[i,2]+dbl,4+po[i,1],1
        if pec>0 then pbontexte(pec);
      end;

      cursbloc;

    until (j=5) or (k=6) or (rep=mo[i].mots);

    ppoints(l,i*2);
    ecritvite(centre(mot[i],max),5+po[i,2]+dbl,4+po[i,1],112)
```

```
      end;
    end;
    eleve.points:=round(l*10);
  end;


(*-----------------------------------------------------------------------

  begin
    if num < 6 then begin
      esc:=true;
      PAIDE(esc);
      if esc then exit;
    end;

    eleve.date_heure:=FDATE_HEURE_TRAV;

    with trav do begin

      clrscr;
      PCADRE(2,2,79,25,tex[1],0,false);
      curscache;

      ecritvite('╠'+chainecar(#205,76)+'╣',23,2,7);

      changeattr(74,24,4,112);

      ecritvite('QUITTER :  [ESC]',24,6,112);

      ecritvite('CONTINUER :  '+#17+#196+#217,24,60,112);

      affichecarlu:=false;
      cachecursdanslitcar:=true;
      i:=2;
      PVBCD(1);
      c:=readkey;
      dbl:=(13-lectu.nbre) div 2;
      while (c<>#27) and (i<=lectu.nbre) do begin
        ecritvite(tex[i],5+i+dbl,5,7);
        PVBCD(100);
        litcar('',1,1,$00,[#13,#27],c);
        inc(i);
      end;

      if c=#27 then begin
        esc:=true;
        exit;
      end;

      tro:=chainecar(#95,max);
      for i:=10 downto 1 do begin
        delete(tex[po[i,2]],po[i,1],length(mo[i].mots));
        insert(tro,tex[po[i,2]],po[i,1]);
```

```
    end;

    i:=0;
    repeat
      while (i<10) and (po[i,2]=po[i+1,2]) do begin
        inc(i);
        po[i,1]:=pos(tro,copy(tex[po[i,2]],po[i,1],
        length(tex[po[i,2]])))+po[i,1]-1;
      end;
      inc(i);
    until i>10;

    for i:=2 to lectu.nbre do ecritvite(tex[i],5+i+dbl,5,7);

    l:=0;
    ppoints(0,0);
    cursbloc;

    ecritvite('ECRIS TA REPONSE          ',24,6,112);

    for i:=1 to 10 do
      for j:=1 to 6 do fautes[i,j]:='';


    ptest;

    ecritvite('                          ',24,6,112);

    curscache;

    eleve.te:=FDATE_HEURE_TRAV;

    reset(fper);
    seek(fper,filesize(fper));

    reset(fere);
    seek(fere,filesize(fere));

    eleve.nbre:=0;
    eleve.deb:=filesize(fere);
    for i:=1 to 10 do
      for j:=1 to 6 do
        if fautes[i,j]<>'' then begin
          write(fere,fautes[i,j]);
          inc(eleve.nbre);
        end;

    if eleve.nbre=0 then eleve.deb:=0;

    write(fper,eleve);

    close(fper);
    close(fere);

end;
```

```
        end; (* fin pelev *)

(*=================== FIN TRAVAIL DE L'EL TEX. ======================*)


        begin

          reset(findt);
          seek(findt,num-1);
          read(findt,lectu);
          close(findt);

          ptrav(trav);

          pelev(trav,esc);

        end;(* FIN DE LA PROC. TEXTES *)
```

2.9 Include IPHR

```
(*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*)
(* ========================  PHRASES  ============================*)
(*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*)


    procedure pphrases(    num : byte;
                       var esc : boolean);


      type

           tlist  = array [1..10] of record
                      phr : array[1..2] of string[65];
                      nbr : byte;                      { nbr de phr }
                      po1 : byte;                      { pos en x   }
                      po2 : byte;                      { pos en y   }
                      mo  : tmots;
                      mot : string15;           { mot dans la phr }
                    end;

           ttitre = string[60];


      var
           lectu     : tind2;
           list      : tlist;
           mom       : tmom;
           titre     : ttitre;
           maxm      : byte;
```

2.9.1 Procedure PTRAV

```
    procedure PTRAV(var list  : tlist;
                    var mom   : tmom;
                    var titre : ttitre;
                    var maxm  : byte);


      var
           i,j,k,l,m,n : byte;
           debut       : word;
           tex         : array [1..20] of string60;


       begin
         reset(fphrases);
         seek(fphrases,lectu.deb-1);
         for k:=1 to lectu.nbre do read(fphrases,tex[k]);
         close(fphrases);
```

```
  delete(tex[1],1,1);
  titre:=tex[1];
  j:=1;
  l:=1;
  for i:=2 to lectu.nbre do
    if pos('*',tex[i])<>0 then begin
      delete(tex[i],pos('*',tex[i]),10);
      list[j].phr[l]:=tex[i];
      list[J].nbr:=l;
      inc(j);
      l:=1;
    end
    else begin
      list[j].phr[l]:=tex[i];
      l:=l+1;
    end;
  maxm:=0;
  for i:=1 to 10 do
    with list[i] do begin
      for j:=1 to nbr do begin
        if (pos('(',phr[j])>0) then begin
          po1:=pos('(',phr[j]);
          delete(phr[j],po1,1);
          po2:=j;
          mo.mots:='';
          k:=po1;
          repeat
            mo.mots:=mo.mots+phr[j,k];
            inc(k);
          until phr[j,k]=')';
          delete(phr[j],k,1);
          mot:=mo.mots;
          mom[i].mom:=mo.mots;
          mom[i].pl:=i;
          if length(mot)>maxm then maxm:=length(mot);
        end;
      end;
    end;
  inc(maxm,3);
  reset(fmots);
  for i:=1 to 10 do begin
    seek(fmots,lectu.lmo[i]);
    read(fmots,list[i].mo);
  end;
  close(fmots);
  PMAJ(mom);
end;  (*  fin ptrav *)
```

2.9.2 Procedure PELEV

```
(*=================================================================*)
(*=========================  travail de l'el PH. ==================*)
(*=================================================================*)

      procedure PELEV(     list  : tlist;
                           mom   : tmom;
                           titre : ttitre;
                           max   : byte;
                       var esc   : boolean);

        type
             tpo2 = array [1..10] of byte;


        var
             c                 : char;
             i,j,k,m,maxm      : byte;
             rep,tro,saisie    : string15;
             ortho,frape,hv    : boolean;
             loer              : tloer;
             po2               : tpo2;
             eleve             : tper;
             fautes            : array [1..10,1..6] of tere;
             rev               : trev;
             lit_list          : tlmo;
```

2.9.2.1 Procedure PAFFICHE

```
      procedure paffiche(j     : byte;
                         list  : tlist);

        var   i : byte;

         begin

           for i:=1 to 2 do ecritvite(chainecar(#32,65),10+i,10,7);

           with list[j] do

             for i:=1 to nbr do ecritvite(phr[i],10+i,10,7);

         end;  (* fin paffiche *)
```

2.9.2.2 Procedure PTEST

```
      procedure PTEST;

        var
```

```
        i,j,k : byte;
        l,pec : real;

procedure pfrappe;

   begin
     PERREUR('ERREUR DE FRAPPE',32,20,1000);
     PPRENDECR(false);
     cursbloc;
   end;


procedure portho(var j   : byte;
                 var pec : real;
                     k   : byte);

  var
       m : byte;

   begin
     inc(j);
     with list[i] do begin
       rev.rev[(10*((num-1) mod 5))+i]:=true;
       if j<4 then fautes[i,j]:=complete(chr(i)+rep,20);

       if pec>0 then pec:=pec-0.5;

       if j<4 then PPRENDECR(true);

       if j<5 then perrortho(j);

       ecritvite(tro,10+po2,9+po1,7);

       if j=1 then begin
         clrscr;
         PCADRE(15,12,65,14,'',7,false);
         curscache;
         ecritvite('Essaie encore une fois, '+
                   copy(rech.nom,15,30),13,24,7);
         PVBCD(10);
         m:=0;
         repeat
           delay(333);
           inc(m);
         until (m=10) or keypressed;
         PPRENDECR(false);
         exit;
       end;


       if j=5 then begin
         PPRENDECR(false);
         curscache;
         ecritvite(centre(mot,max),10+po2,9+po1,112);
```

```
          ecritvite('                              ',15,9,112);
          PVBCD(10);
          m:=0;
          repeat
            delay(333);
            inc(m);
          until (m=10) or keypressed;
          cursbloc;
          exit;
        end;

        if j>1 then pf_b(mo,loer,maxm,j);

        if j<4 then PPRENDECR(false);
        cursbloc;
      end;
    end;


procedure psens(var m : byte;
                var pec : real);

    begin

      with list[i] do begin

        inc(m);
        if m<4 then fautes[i,3+m]:=complete(chr(i)+'*'+rep,20);

        if m=6 then begin
          PPRENDECR(false);
          curscache;
          ecritvite(centre(mot,max),10+po2,9+po1,112);
          c:=readkey;
          cursbloc;
          pec:=0;
          exit;
        end;

        if (m=2) and (pec>0) then pec:=pec-0.5
        else if (m=4) and (pec>0) then pec:=pec-0.5
          else if (m=5) and (pec>0) then pec:=pec-0.5;

        PPRENDECR(true);

        perrsens(m);

        psensaff(m);

        curscache;

        if (m>1) then paffliste(m,i,mom);

        PPRENDECR(false);
```

```
    cursbloc;

  end;

end;


begin

  if num=5 then rev.p10:=0;
  l:=0;
  for i:=1 to 10 do begin
    with list[i] do begin
      curscache;
      paffiche(i,list);
      cursbloc;
      j:=0;
      m:=0;
      pec:=2;
      repeat
        ecritvite('ECRIS TA REPONSE            ',15,9,112);
        repeat
          saisie:='';
          psaisie(saisie,max,9+po1,10+po2,false,esc);
        until saisie<>'';

        rep:=copy(chnminusc(saisie),1,15);

        if rep<>mo.mots then begin

          if (j<4) then PPRENDECR(true);

          ecritvite(chainecar(#32,20),15,7,112);

          ortho:=true;

          pana(rep,mo,frape,ortho,loer,tabmot,maxm);

          if j=4 then begin
            frape:=false;
            ortho:=true;
          end;

          if frape then pfrappe
          else
            if ortho then portho(j,pec,k) else psens(m,pec);

          ecritvite(tro,10+po2,9+po1,7);
        end
        else begin
          if roptions.bruitage then begin
            sound(220);
            delay(150);
            nosound;
```

```
            delay(100);
            sound(175);
            delay(150);
            nosound;
          end;
          l:=l+pec;
          curscache;
          ppoints(l,i*2);
          if pec>0 then begin
            ecritvite(centre(mot,length(tro)),10+po2,9+po1,112);
            pbonphrase(pec);
          end;
        end;

      until (j=5) or (m=6) or (rep=mo.mots);

      if j>0 then PPRENDECR(false);

      ppoints(l,i*2);

      if i=10 then ecritvite(centre(mot,length(tro))
                             ,10+po2,9+po1,112);

    end;
  end;
  eleve.points:=round(l*10);
end;


begin

  if num<6 then begin
    esc:=false;
    PAIDE(esc);
    if esc then exit;
  end;

  reset(fper);
  seek(fper,filesize(fper));

  eleve.date_heure:=FDATE_HEURE_TRAV;

  reset(fmots);

  for i:=1 to 10 do begin
    seek(fmots,lectu.lmo[i]);
    read(fmots,list[i].mo)
  end;

  close(fmots);

  clrscr;
  PCADRE(5,7,75,16,titre,0,false);
  curscache;
```

```
ecritvite('╠'+chainecar(#205,69)+'╣',14,5,7);
changeattr(67,15,7,112);
ecritvite('QUITTER :  [ESC]',15,9,112);
ecritvite('CONTINUER :  '+#17+#196+#217,15,56,112);


i:=1;
PVBCD(1);
c:=readkey;
while (i<=10) and (c<>#27) do begin
  paffiche(i,list);
  PVBCD(100);
  c:=readkey;
  inc(i);
end;

if c=#27 then begin
  esc:=true;
  close(fper);
  exit;
end;

tro:=chainecar(#95,max);

for i:=10 downto 1 do begin
  with list[i] do begin
    delete(phr[po2],po1,length(mo.mots));
    insert(tro,phr[po2],po1);
  end;
end;

ppoints(0,0);
curscache;
k:=2;

reset(frev);
seek(frev,0);
read(frev,rev);
close(frev);

for i:=1 to 10 do
  for j:=1 to 6 do fautes[i,j]:='';

cursbloc;

ptest;

ecritvite('                    ',15,9,112);

curscache;

eleve.te:=FDATE_HEURE_TRAV;

reset(frev);
seek(frev,0);
```

```
    write(frev,rev);
    close(frev);

    reset(fere);
    seek(fere,filesize(fere));
    eleve.nbre:=0;
    eleve.deb:=filesize(fere);
    for i:=1 to 10 do
      for j:=1 to 6 do
        if fautes[i,j]<>'' then begin
          write(fere,fautes[i,j]);
          inc(eleve.nbre)
        end;

    if eleve.nbre=0 then eleve.deb:=0;

    reset(fper);
    seek(fper,filesize(fper));

    write(fper,eleve);

    close(fper);
    close(fere);

  end;


(*============================================================*)
(*============== FIN TRAVAIL DE L'EL PHR. ================*)
(*============================================================*)



  begin

    reset(findp);
    seek(findp,num-1);
    read(findp,lectu);
    close(findp);

    ptrav(list,mom,titre,maxm);
    pelev(list,mom,titre,maxm,esc);

  end;(*-- FIN DE LA PROC. PHRASES --*)
```

2.10 Include IREV

```
(*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*)
(* ======================  REVISIONS  ============================*)
(*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*)


    Procedure PREVISION(var rech         : tfel;
                            pl           : word;
                        var esc,fin,exf  : boolean);

      type

           tlist = array [1..50] of record
                     phr : array[1..2] of string[65];
                     nbr : byte;                          { nbr de phr }
                     po1 : byte;                          { pos en x   }
                     po2 : byte;                          { pos en y   }
                     mo  : tmots;
                     mot : string15;              { mot dans la phr }
                   end;

           tmomr  = array [1..5] of tmom; { liste des mots en maj }


      var
           rev                                : trev;
           titre                              : string[3];
           saisie,rep                         : string15;
           tro                                : string[20];
           i,j,k,y,m,maxm                     : byte;
           l,pec                              : real;
           list                               : tlist;
           ortho,frape,hv,debut_de_rev        : boolean;
           loer                               : tloer;
           mom                                : tmomr;
```

2.10.1 Function FREV_FIN

```
    function FREV_FIN(rev : trev) : boolean;

      var i : byte;

       begin

         i:=0;
         repeat
           inc(i);
           if rev.rev[I] then begin
             FREV_FIN:=true;
```

```
          exit;
        end
        else FREV_FIN:=false;
      until i=50;

    end;
```

2.10.2 Procedure PPREP

```
procedure PPREP(var list : tlist;
                var mom  : tmomr;
                var maxm : byte;
                    num  : byte);

  var
      i,j,k,l,m,n : byte;

      ind     : tind2;
      debut   : word;
      lmo     : array [1..5] of tlmo;
      tex     : array [1..100] of string60;


   begin
     reset(findp);
     seek(findp,num*5);
     read(findp,ind);
     lmo[1]:=ind.lmo;
     debut:=ind.deb;
     k:=ind.nbre;
     for i:=1 to 4 do begin
       read(findp,ind);
       lmo[i+1]:=ind.lmo;
       inc(k,ind.nbre);
     end;
     close(findp);
     reset(fphrases);
     seek(fphrases,debut);
     for i:=1 to k do read(fphrases,tex[I]);
     close(fphrases);
     j:=1;
     l:=1;
     for i:=1 to k do
       if pos('#',tex[i])=0 then begin
         if pos('*',tex[i])<>0 then begin
           delete(tex[i],pos('*',tex[i]),10);
           list[j].phr[l]:=tex[i];
           list[J].nbr:=l;
           inc(j);
           l:=1;
         end
         else begin
           list[j].phr[l]:=tex[i];
           inc(l);
```

```
            end;
          end;
        maxm:=0;
        for i:=1 to 50 do
          with list[i] do begin
            for j:=1 to nbr do begin
              if (pos('(',phr[j])>0) then begin
                po1:=pos('(',phr[j]);
                delete(phr[j],po1,1);
                po2:=j;
                mo.mots:='';
                k:=po1;
                repeat
                  mo.mots:=mo.mots+phr[j,k];
                  inc(k);
                until phr[j,k]=')';
                delete(phr[j],k,1);
                mot:=mo.mots;
                m:=((i-1) div 10)+1;
                n:=((i-1) mod 10)+1;
                mom[m,n].mom:=mo.mots;
                mom[m,n].pl:=n;
                if length(mot)>maxm then maxm:=length(mot);
              end;
            end;
          end;
        inc(maxm,3);
        reset(fmots);
        for i:=0 to 4 do
          for j:=1 to 10 do
          with list[(i*10)+j] do if rev.rev[(i*10)+j] then begin
            seek(fmots,lmo[i+1,j]);
            read(fmots,mo);
          end;
        close(fmots);
        for i:=1 to 5 do PMAJ(mom[i]);
      end;
```

2.10.3 Procedure PAFFICHE

```
    procedure PAFFICHE(    j : byte);

      var   i : byte;

       begin
         for i:=1 to 2 do ecritvite(chainecar(#32,65),10+i,10,7);
         for          i:=1          to          list[j].nbr          do
ecritvite(list[j].phr[i],10+i,10,7);
       end;
```

2.10.4 Procedure TEST

```
    procedure TEST;
```

```
var
     i,j,k,m : byte;
```

2.10.4.1 Procedure PFRAPPE

```
procedure PFRAPPE(i:byte);

   begin
     curscache;
     with list[i] do ecritvite(tro,10+po2,9+po1,7);
     if j<4 then PPRENDECR(TRUE);
     PERREUR('ERREUR DE FRAPPE',32,20,1000);
     PPRENDECR(false);
     cursbloc;
   end;
```

2.10.4.2 Procedure PORTHO

```
procedure portho( var j    : byte;
                      i    : byte;
                  var pec  : real);

  var
       m,k : byte;


   begin
     inc(j);
     rev.rev[i]:=true;
     with list[i] do begin
     if pec>0 then pec:=pec-0.5;
       if j<4 then PPRENDECR(true);
       if j<5 then perrortho(j);
       ecritvite(tro,10+po2,9+po1,7);
       if j=1 then begin
         clrscr;
         PCADRE(15,12,65,14,'',7,false);
         curscache;
         ecritvite('Essaie encore une fois, '+
                   copy(rech.nom,15,30),13,24,7);
         PVBCD(10);
         m:=0;
         repeat
           delay(333);
           inc(m);
         until (m=10) or keypressed;
         PPRENDECR(false);
         cursbloc;
         exit;
       end;
       if j=5 then begin
         PPRENDECR(false);
         curscache;
```

```
            ecritvite(centre(mot,length(tro)),10+po2,9+po1,112);
            ecritvite('                                ',15,10,112);
            PVBCD(10);
            m:=0;
            repeat
              delay(333);
              inc(m);
            until (m=10) or keypressed;
            cursbloc;
            exit;
          end;
          pf_b(mo,loer,maxm,j);
          if j<4 then PPRENDECR(false);
          cursbloc;
        end;
      end;
```

2.10.4.3 Procedure PSENS

```
procedure PSENS( var m   : byte;
                     i   : byte;
                 var pec : real);

  var
      n : byte;


  begin
    with list[i] do begin
      inc(m);
      if m=6 then begin
        PPRENDECR(false);
        curscache;
        ecritvite(centre(mot,length(tro)),10+po2,9+po1,112);
        c:=readkey;
        cursbloc;
        exit;
      end;
      if (m=2) and (pec>0) then pec:=pec-0.5
      else if (m=4) and (pec>0) then pec:=pec-0.5
        else if (m=5) and (pec>0) then pec:=pec-0.5;
      PPRENDECR(true);
      perrsens(m);
      psensaff(m);
      if (m>1) then
        paffliste(m,((i-1) mod 10)+1 ,mom[((i-1) div 10)+1]);
      PPRENDECR(false);
      cursbloc;
    end;
  end;



  begin
```

```
k:=rev.p10;
i:=0;
repeat
  inc(i);
  pec:=2;
  repeat
    inc(k);
  until (k=50) or rev.rev[k];
  if not (k>50) and rev.rev[k] then
    with list[k] do begin
      curscache;
      PAFFICHE(k);
      cursbloc;
      j:=0;
      m:=0;
      repeat
        ecritvite('ECRIS TA REPONSE',15,9,112);
        repeat
          saisie:='';
          PSAISIE(saisie,length(tro),9+po1,
                  10+po2,false,esc);
        until saisie<>'';
        rep:=copy(saisie,1,15);
        rep:=copy(chnminusc(rep),1,15);
        if rep<>mo.mots then begin
          if (j<4) then PPRENDECR(true);
          ecritvite(chainecar(#32,20),15,7,112);
          ortho:=true;
          PANA(rep,mo,frape,ortho,loer,tabmot,maxm);
          if j=4 then begin
            frape:=false;
            ortho:=true;
          end;
          if frape then PFRAPPE(k)
          else if ortho then PORTHO(j,k,pec)
            else PSENS(m,k,pec);
          ecritvite(tro,10+po2,9+po1,7);
        end
        else begin
          if roptions.bruitage then begin
            sound(220);
            delay(150);
            nosound;
            delay(100);
            sound(175);
            delay(150);
            nosound;
          end;
          l:=l+pec;
          curscache;
          ppoints(l,i*2);
          if pec>0 then begin
            ecritvite(centre(mot,length(tro)),10+po2,9+po1,11
            pbonphrase(pec);
          end;
```

```
          end;
        until (j=5) or (m=6) or (rep=mo.mots);
        if (j>0) then PPRENDECR(false);
        ppoints(l,i*2);
        ecritvite(centre(mot,length(tro)),10+po2,9+po1,112);
        if (m<7) and (j=0) then rev.rev[k]:=false;
      end;
    until (i=10) or (k=50);
    rev.p10:=k;
  end;


  begin
    debut_de_rev:=true;
    repeat

      clrscr;

      reset(frev);
      seek(frev,0);
      read(frev,rev);
      close(frev);

      if not FREV_FIN(rev) then begin
        reset(ffel);
        inc(rech.rev);
        seek(ffel,p1);
        write(ffel,rech);
        exf:=not debut_de_rev;
        exit;
      end
      else if rev.p10=50 then rev.p10:=0;

      if not debut_de_rev then fin:=FFIN;

      if fin then exit;

      debut_de_rev:=false;

      pprep(list,mom,maxm,rech.rev);

      clrscr;
      str(rech.rev+1,titre);
      PCADRE(5,7,75,16,'REVISION N° '+titre,0,false);
      curscache;

      ecritvite('╠'+chainecar(#205,69)+'╣',14,5,7);

      changeattr(67,15,7,112);

      ecritvite('QUITTER :  [ESC]',15,9,112);

      ecritvite('CONTINUER :  '+#17+#196+#217,15,55,112);
```

```
      tro:=chainecar(#95,maxm);

      i:=0;
      j:=rev.p10;
      PVBCD(1);
      c:=readkey;
      while (i<10) and (j<50) and (c<>#27) do begin
        inc(i);
        repeat
          inc(j);
          if rev.rev[j] then begin
            PAFFICHE(j);
            PVBCD(100);
            c:=readkey;
            if c<>#27 then
              with list[j] do begin
                delete(phr[po2],po1,length(mo.mots));
                insert(tro,phr[po2],po1);
              end;
          end;
        until rev.rev[j] or (j=50) or (c=#27)
      end;

      if c=#27 then begin
        esc:=true;
        exit;
      end;

      ppoints(0,0);
      cursbloc;
      l:=0;

      test;

      ecritvite('                                    ',15,9,112);

      reset(frev);
      seek(frev,0);
      write(frev,rev);
      close(frev);



    until fin;

  end;(*-- de REVISION --*)
```

2.11 Unit MAITRE

```
{F+,O+}

unit MAITRE;

interface

   uses
          tpcrt,
          dos,
          proc,
          def,
          printer,
          tpchaine,
          tpedit,
          tpint24;

 procedure pmaitre;

implementation


   procedure PMAITRE;


     const
          menu : tmenu = ('Menu Maître',
                          'Consul. des Fichiers',
                          'Suppression',
                          'Init. date et heure',
                          'Options',
                          'Fin','','','');

     var
          nbre : byte;
```

2.11.1 Function FCODE

```
{ code secret }

  function fcode : boolean;

    var
         j,lcode : byte;
         code2   : string[10];


     begin

       code2:='';
       lcode:=length(roptions.code_se);
       clrscr;
       PCADRE(20,10,60,15,'Code Secret',0,false);
       ecritvite(chainecar('_',lcode),13,40-(lcode div 2),112);
       for j:=1 to lcode do begin
         gotoxy(39-(lcode div 2)+j,13);
         c:=readkey;
         if c=#27 then begin
           fcode:=false;
           exit;
         end;
         code2:=code2+c;
         ecritvite('X',13,39-(lcode div 2)+j,112);
       end;

       contrlbreak:=false;

       if code2=roptions.code_se then fcode:=true
       else begin
         fcode:=false;
         for j:=1 to 10 do begin
           sound(1000);
           delay(500);
           nosound;
           delay(125);
         end;
       end;
       contrlbreak:=true;
     end;
```

2.11.2 Procedure PCONSULT

```
procedure PCONSULT;

  type
       tmoi = array [1..12] of string[9];

  const
       moi : tmoi =('janvier','février','mars','avril',
                    'mai','juin','juillet','août','septembre'
                   ,'octobre','novembre','décembre');

       menu : tmenu = ('Consultation',
                       'Ponctuelle',
                       'Impression',
                       'Bilan',
                       'Fin','','','','');

  var
       choix   : byte;
       li_per  : tper;
       mo      : tindm;
       ind     : tind2;
       dt1     : datetime;
       dt2     : datetime;
       ere     : tere;
       tabere  : array [1..10,1..2,1..3] of string[18];
       cte     : longint;
```

2.11.2.1 Procedure IMPRIMER

```
procedure IMPRIMER( fiche : word;
                    rech  : tfel);
  var
       i,j,k : byte;

   begin

     if not PRINTERREADY then exit;

     writeln(lst,#13);
     writeln(lst,#15);

     writeln(lst,complete('',25)+#218+chainecar(#196,32)+#194
                                     +chainecar(#196,19)+#194
                                     +chainecar(#196,10)+#194
                                     +chainecar(#196,7) +#194
                                     +chainecar(#196,9) +#191);

     write(lst,complete('',25)+#179);
     write(lst,' '+centre('NOM DE L''ELEVE',31));
     write(lst,#179);
```

```
write(lst,' DATE DE L''EXERCICE');
write(lst,#179);
write(lst,centre('HEURE',10));
write(lst,#179);
write(lst,centre('SCORE',7));
write(lst,#179);
writeln(lst,centre('TEMPS',8)+' '+#179);

writeln(lst,complete('',25)+#195+chainecar(#196,32)+#197+
                                 chainecar(#196,19)+#197+
                                 chainecar(#196,10)+#197+
                                 chainecar(#196,7) +#197+
                                 chainecar(#196,9) +#180);

seek(fper,fiche);
read(fper,li_per);

write(lst,complete('',25)+#179);
write(lst,' ',copy(rech.nom,1,15)+'
       '+copy(rech.nom,16,30));
write(lst,#179);

with li_per do begin


   write(lst,' ',centre(formate('chr(64)#',dt1.day)+'
          '+moi[dt1.month]+' '+
              formate('####',dt1.year),17),' ');

   write(lst,#179);

    write(lst,' ',formate('chr(64)#',dt1.hour)+':'+
              formate('chr(64)#',dt1.min)+':'+
              formate('chr(64)#',dt1.sec),' ');

    write(lst,#179);

    write(lst,' ',formate('##.#',(points/10)),'  ');
    write(lst,#179);


    writeln(lst,' ',formate('#',cte div 3600)+':'+
         formate('chr(64)#',(cte mod 3600) div 60)+':'+
              formate('chr(64)#',cte mod 60),' ',#179);


    writeln(lst,complete('',25)+#192+chainecar(#196,32)+#193
                                    +chainecar(#196,19)+#193
                                    +chainecar(#196,10)+#193
                                    +chainecar(#196,7)+#193
                                    +chainecar(#196,9)+#217);


    writeln(lst,#18);
```

```
        writeln(lst,'');
        writeln(lst,'');
        writeln(lst,'');

        writeln(lst,#201+chainecar(#205,24)+#209
                        +chainecar(#205,28)+#209
                        +chainecar(#205,24)+#187);

        write(lst,#186+centre('Erreur de sens',24)+#179);


        if not odd(fiche) then begin
          write(lst,centre('Phrases de  '+
                           longenchn(((fiche div 2)-1)*10+1)+
                           ' à '+
                       longenchn(((fiche div 2)-1)*10+10),28));
        end
        else begin
write(lst,centre('Texte  n°'+longenchn((fiche div 2)+1),28));
        end;

        writeln(lst,#179+'  Erreur d''orthographe'+'  '+#186);



        for i:=0 to 9 do begin
          seek(findm,ind.lmo[i+1]);
          read(findm,mo);

          writeln(lst,#199+chainecar(#196,24)+#197
                        +chainecar(#196,28)+#197
                        +chainecar(#196,24)+#182);

      write(lst,#186+'    '+complete(tabere[i+1,1,1],21)+#179);
          write(lst,'       '+centre(mo,15),'        ');
          writeln(lst,#179,'     ',tabere[i+1,2,1]+'  '+#186);

          for k:=2 to 3 do begin
      write(lst,#186+'    ',complete(tabere[i+1,1,k],21),#179);
      writeln(lst,complete('',28),#179,'     ',tabere[i+1,2,k]+
                        '  '+#186);
          end;

        end;

        writeln(lst,#200+chainecar(#205,24)+#207
                       +chainecar(#205,28)+#207
                       +chainecar(#205,24)+#188);

    end;
    for i:=1 to 12 do writeln(lst,'');

end;
```

2.11.2.2 Procedure PLIS

```
{ consultation ponctuelle }

   procedure PLIS;

     var
          rech          : tfel;
          pl            : word;
          trouver,esc   : boolean;


   procedure AFFRES(rech:tfel);


     var
          esc,ok  : boolean;
          i,k,l   : byte;
          j       : word;


      begin

        seek(fper,1);
        clrscr;
        attrtexte:=7;
        changeattr(80,4,1,112);
        changeattr(80,25,1,112);

        ecritvite('Erreur de sens'          , 4, 5,112);
        ecritvite('Erreur d''orthographe', 4,58,112);

        ecritvite('[S]uivant'   ,25,3 ,112);
        ecritvite('[P]récédent',25,17,112);
        ecritvite('[D]ébut'     ,25,33,112);
        ecritvite('[F]in'       ,25,44,112);
        ecritvite('[I]mprimer'  ,25,54,112);
        ecritvite('[Q]uitter'   ,25,70,112);

        ecritvite(chainecar(#196,80),3,1,7);
        changeattr(80,1,1,112);
        ecritvite('NOM DE L''ELEVE',1,8,112);
        ecritvite(copy(rech.nom,1,15)+' '
                  +copy(rech.nom,16,30),2,1,15);
        ecritvite(#179,1,32,112);
        ecritvite(#179,2,32,7);
        ecritvite(#193,3,32,7);

        j:=0;
        repeat

          inc(j);
          seek(fper,j);
          read(fper,li_per);
```

```
with li_per do begin

  unpacktime(date_heure,dt1);

  ecritvite('DATE DE L''EXERCICE',1,34,112);
  ecritvite(centre(formate('chr(64)#',dt1.day)+' '
            +moi[dt1.month]+' '+
            formate('####',dt1.year),17),2,34,15);
  ecritvite(#179       ,1,52,112);
  ecritvite(#179       ,2,52,  7);
  ecritvite(#193       ,3,52,  7);

  ecritvite('HEURE'    ,1,55,112);
  ecritvite(formate('chr(64)#',dt1.hour)+
            ':'+formate('chr(64)#',dt1.min)+':'+
            formate('chr(64)#',dt1.sec),2,54,15);
  ecritvite(#179       ,1,63,112);
  ecritvite(#179       ,2,63,  7);
  ecritvite(#193       ,3,63,  7);

  ecritvite('SCORE'    ,1,65,112);
  ecritvite(formate('##.#',(points/10)),2,65,15);
  ecritvite(#179       ,1,71,112);
  ecritvite(#179       ,2,71,  7);
  ecritvite(#193       ,3,71,  7);

  for i:=1 to 20 do begin
    ecritvite(#179,4+i,25,7);
    ecritvite(#179,4+i,55,7);
  end;

  cte:=(dt1.hour*3600)+(dt1.min*60)+dt1.sec;
  unpacktime(te,dt2);
  cte:=((dt2.hour*3600)+(dt2.min*60)+dt2.sec)-cte;

  ecritvite('TEMPS'    ,1,74,112);
  ecritvite(formate('#',cte div 3600)+':'+
      formate('chr(64)#',(cte mod 3600) div 60)+':'+
            formate('chr(64)#',cte mod 60),2,73,15);

  if not odd(j) then begin
    ecritvite(centre('Phrases de  '+
                     longenchn(((j div 2)-1)*10+1)+
                     ' à '+
                     longenchn(((j div 2)-1)*10+10),28),
                     4,27,112);
    seek(findp,(j div 2)-1);
    read(findp,ind);
  end
  else begin
ecritvite(centre('Texte  n°'+longenchn((j div 2)+1),28),
                     4,27,112);
    seek(findt,(j div 2));
    read(findt,ind);
  end;
```

```
    for i:=0 to 9 do begin
      seek(findm,ind.lmo[i+1]);
      read(findm,mo);
      ecritvite(centre(mo,15),5+(i*2),33,7);
    end;

    for i:=1 to 10 do
      for k:=1 to 2 do
        for l:=1 to 3 do
          tabere[i,k,l]:=complete('',18);

    seek(fere,deb);

    for i:=1 to nbre do begin
      read(fere,ere);
      if ere[2]='*' then begin
        if tabere[ord(ere[1]),1,1,1]=' ' then
          tabere[ord(ere[1]),1,1]:=copy(ere,3,20)
        else if tabere[ord(ere[1]),1,2,1]=' ' then
          tabere[ord(ere[1]),1,2]:=copy(ere,3,20)
          else if tabere[ord(ere[1]),1,3,1]=' ' then
            tabere[ord(ere[1]),1,3]:=copy(ere,3,20)
      end
      else begin
        if tabere[ord(ere[1]),2,1,1]=' ' then
          tabere[ord(ere[1]),2,1]:=copy(ere,2,20)
        else if tabere[ord(ere[1]),2,2,1]=' ' then
          tabere[ord(ere[1]),2,2]:=copy(ere,2,20)
          else if tabere[ord(ere[1]),1,3,1]=' ' then
            tabere[ord(ere[1]),2,3]:=copy(ere,2,20)
      end;
    end;


    l:=0;
    for i:=1 to 10 do
      for k:=0 to 1 do begin
        inc(l);
        ecritvite(tabere[i,1,k+1],4+l,3,7);
        ecritvite(tabere[i,2,k+1],4+l,59,7);
      end;

    for i:=1 to 5 do begin
      changeattr(80,2+(i*4)+1,1,15);
      changeattr(80,2+(i*4)+2,1,15);
    end;

  end;

  repeat
    ok:=false;
    c:=readkey;
    c:=upcase(c);
    case c of
```

```
      #81,#27   : exit;

          #83   : if j<filesize(fper)-1 then ok:=true;

          #80   : if j>1 then begin
                    dec(j,2);
                    ok:=true;
                  end;

          #68   : if j>1 then begin
                    j:=0;
                    ok:=true;
                  end;

          #70   : if j<filesize(fper)-1 then begin
                    j:=filesize(fper)-2;
                    ok:=true;
                  end;

          #73   : IMPRIMER(j,rech);

          end;

      until ok;
    until not ok;
  end;


  begin
    curscache;
    piofel;
    if filesize(ffel)=1 then exit;
    repeat
      esc:=true;
      clrscr;
      PCADRE(26,1,55,3,'',112,false);
      ecritvite('  CONSULTATION  PONCTUELLE  ',2,27,112);
      PLIST_NOM(rech,pl,trouver,esc);
      if esc then exit;
      reset(fere);
      reset(fper);
      if filesize(fper)>1 then affres(rech);
      close(fere);
      close(fper);
    until esc;
    close(fper);
    close(fere);
  end;
```

2.11.2.3 Procedure PIMP

```
{  impression des résultats  }

procedure PIMP;
  var
       debut,fin,pl : word;
       esc,trouver  : boolean;

procedure PIMPB;
  var
       j,i,k,l : byte;

   begin
     seek(fper,1);
     j:=0;
     repeat
       inc(j);
       seek(fper,j);
       read(fper,li_per);

       with li_per do begin

         unpacktime(date_heure,dt1);

         cte:=(dt1.hour*3600)+(dt1.min*60)+dt1.sec;

         unpacktime(te,dt2);

         cte:=((dt2.hour*3600)+(dt2.min*60)+dt2.sec)-cte;

         if not odd(j) then begin
           seek(findp,(j div 2)-1);
           read(findp,ind);
         end
         else begin
           seek(findt,(j div 2));
           read(findt,ind);
         end;

         for i:=1 to 10 do
           for k:=1 to 2 do
             for l:=1 to 3 do
               tabere[i,k,l]:=complete('',18);

         seek(fere,deb);

         for i:=1 to nbre do begin
           read(fere,ere);
           if ere[2]='*' then begin
             if tabere[ord(ere[1]),1,1,1]=' ' then
               tabere[ord(ere[1]),1,1]:=copy(ere,3,20)
             else if tabere[ord(ere[1]),1,2,1]=' ' then
               tabere[ord(ere[1]),1,2]:=copy(ere,3,20)
               else if tabere[ord(ere[1]),1,3,1]=' ' then
```

```
                tabere[ord(ere[1]),1,3]:=copy(ere,3,20)
            end
            else begin
              if tabere[ord(ere[1]),2,1,1]=' ' then
                tabere[ord(ere[1]),2,1]:=copy(ere,2,20)
              else if tabere[ord(ere[1]),2,2,1]=' ' then
                tabere[ord(ere[1]),2,2]:=copy(ere,2,20)
                else if tabere[ord(ere[1]),1,3,1]=' ' then
                  tabere[ord(ere[1]),2,3]:=copy(ere,2,20)
            end;
          end;

      end;

      (*imprimer(j,rech);*)

      PCADRE(10,10,70,15,'',7,false);
      ecritvite('                    ',12,12,7);
      for i:=1 to 4 do changeattr(59,10+i,11,112);
      ecritvite('Pressez [ESC] pour stoper l'
                'impression',12,25,112);
      ecritvite('ou RETURN pour la fiche suivante.',13,22,112);
      PVBCD(10);
      c:=readkey;

    until (j>=filesize(fper)-1) or (c=#27);

end;

begin
  curscache;
  piofel;
  if filesize(ffel)=1 then exit;
  repeat
    clrscr;
    esc:=true;
    PCADRE(33,1,48,3,'',112,false);
    ecritvite('  IMPRESSION  ',2,34,112);
    PLIST_NOM(rech,pl,trouver,esc);
    if esc then exit;
    reset(fere);
    reset(fper);
    if (filesize(fper)>10) and not esc then begin
      debut:=1;
      fin:=1;
      clrscr;
      PCADRE(2,2,78,25,'Impression',0,false);
      litmot('De la serie  ( 1 à 10 ) : ',10,10,3,7,112,
             1,10,esc,debut);
      litmot(' à la serie  ( 1 à 10 ) : ',12,15,3,7,112,
             debut,10,esc,fin);
      pimpb;
    end;
    close(fper);
    close(fere);
```

```
  until esc;
end;
```

2.11.2.4 Procedure PBILAN

```
procedure PBILAN;
  var
       lffel : tfel;
       i     : byte;

   begin
     curscache;
     piofel;
     if filesize(ffel)=1 then exit;
     clrscr;
     PCADRE(2,2,79,25,'Bilan',7,false);
     ecritvite('    Quitter  :  [ESC] '
                +chainecar(#32,48),25,6,112);
     seek(ffel,1);
     for i:=1 to filesize(ffel)-1 do begin
       read(ffel,lffel);
       with lffel do begin
         ecritvite(completecar(effespfin(copy(nom,1,15))+'  '+
         copy(effespfin(copy(nom,16,30))+'  ',1,15),#196,30)+
                    chainecar(#196,10)+#16,4+i,7,7);
         if num = 0 then ecritvite('aucun exercice effectué',
                                   4+i,50,7)
         else
         if not odd(num) then
           ecritvite('Phrases de  '
                      +longenchn(((num div 2)-1)*10+1)+
                     ' à '
                      +longenchn(((num div 2)-1)*10+10),
                      4+i,50,7)
         else ecritvite('Texte  n°'
                        +longenchn((num div 2)+1),4+i,50,7);
       end;
     end;
     repeat
       c:=readkey;
     until c=#27;
   end;


   begin
     curscache;
     reset(findm);
     reset(findp);
     reset(findt);
     repeat
       choix:=1;
       PMENU(menu,0,0,0,0,4,choix);
       case choix of
         1 : PLIS;
         2 : PIMP;
         3 : PBILAN;
       end;
     until choix=4;
```

```
    close(findm);
    close(findp);
    close(findt);
end;
```

2.11.3 Procedure PSUPP

```
procedure PSUPP;
  var
       trouver,
       esc        : boolean;
       pl         : word;
       i          : byte;
       lfel       : tfel;

   begin
     curscache;
     piofel;
     repeat
       if filesize(ffel)=1 then exit;
       clrscr;
       PCADRE(32,1,49,3,'',112,false);
       ecritvite('  SUPPRESSION   ',2,33,112);
       esc:=true;
       PLIST_NOM(rech,pl,trouver,esc);
       if esc then exit;
       PCADRE(20,23,60,25,'',7,true);
       cachecursdanslitcar:=false;
       affichecarlu:=true;
    if ouiounon('En êtes-vous sûr (Oui/Non) ? : ',24,25,$07,#0)
      then
       begin
         erase(fere);
         erase(fper);
         for i:=pl to filesize(ffel)-2 do begin
           seek(ffel,i+1);
           read(ffel,lfel);
           seek(ffel,i);
           write(ffel,lfel);
         end;
         seek(ffel,filesize(ffel)-1);
         truncate(ffel);
         close(ffel);
         reset(ffel);
       end;
     until esc;
   end;
```

2.11.4 Procedure PPARAM

```
{  modif des options  }

procedure PPARAM;
  var
       s   : string;
       esc : boolean;

   begin
     clrscr;
     PCADRE(2,2,79,25,'Options',7,false);
     ecritvite(#204+chainecar(#205,76)+#185,23,2,7);
     changeattr(74,24,4,112);
     ecritvite('QUITTER    :  [ESC]',24,8,112);
     ecritvite('CONTINUER :  '+#17+#196+#217,24,60,112);
     with roptions do begin
       ecritvite('Musique      : ',10,20,7);
       if musique then begin
         ecritvite('Oui',10,45,112);
         ecritvite('Non',10,35,7);
       end
       else begin
         ecritvite('Oui',10,45,7);
         ecritvite('Non',10,35,112);
       end;
       ecritvite('Bruitage     : ',12,20,7);

       if bruitage then begin
         ecritvite('Oui',12,45,112);
         ecritvite('Non',12,35,7);
       end
       else begin
         ecritvite('Oui',12,45,7);
         ecritvite('Non',12,35,112);
       end;

       ecritvite('Code secret : ',14,20,7);
       ecritvite(complete(code_se,15),14,35,112);
       ecritvite(#17+#196+#196+#196+#16,10,39,7);

       repeat
         c:=readkey;
         if c=#27 then exit
         else
           if c=#0 then begin
             c:=readkey;
             case  c of
               #75 : if musique then begin
                       musique:=false;
                       changeattr(3,10,35,112);
                       changeattr(3,10,45,7);
                     end
                     else begin
```

```
                musique:=true;
                changeattr(3,10,45,112);
                changeattr(3,10,35,7);
              end;

        #77 :  if musique then begin
                musique:=false;
                changeattr(3,10,35,112);
                changeattr(3,10,45,7);
               end
               else begin
                musique:=true;
                changeattr(3,10,45,112);
                changeattr(3,10,35,7);
               end;
      end;
  end;

until c=#13;

ecritvite(#17+#196+#196+#196+#16,10,39,0);
ecritvite(#17+#196+#196+#196+#16,12,39,7);
ecritvite('     ',10,39,7);
repeat
  c:=readkey;
  if c=#27 then exit
  else
    if c=#0 then begin
      c:=readkey;
      case  c of
        #75 : if bruitage then begin
                bruitage:=false;
                changeattr(3,12,35,112);
                changeattr(3,12,45,7);
              end
              else begin
                bruitage:=true;
                changeattr(3,12,45,112);
                changeattr(3,12,35,7);
              end;

        #77 :  if bruitage then begin
                bruitage:=false;
                changeattr(3,12,35,112);
                changeattr(3,12,45,7);
               end
               else begin
                bruitage:=true;
                changeattr(3,12,45,112);
                changeattr(3,12,35,7);
               end;
      end;
  end;

until c=#13;
```

```
      ecritvite(#17+#196+#196+#196+#16,12,39,0);
      s:=code_se;
      litchaine('',14,35,15,0,112,0,esc,s);
      if esc then begin
        rewrite(foptions);
        write(foptions,roptions);
        close(foptions);
        exit
      end;
      code_se:=copy(s,1,15);
      rewrite(foptions);
      write(foptions,roptions);
      close(foptions);
    end;

end;
```

2.11.5 Procedure PDH

```
{  modif de la date et de l'heure }

 procedure PDH;

   var
        he,mi,se,dise,
        aa,mm,jj,js    : word;
        mise           : string[10];
        erre,erre1     : integer;


    begin


(*-------------------------------------------------------------------------*)
(* ----------------------- Mise à jour de l'HEURE   -----------------------*)
(*-------------------------------------------------------------------------*)


        clrscr;
        PCADRE(15,10,66,15,'Mise à jour de l''HEURE',0,false);

        repeat

          ecritvite(' Validation  (Oui/Non) :   ',15,26,112);

          repeat
            repeat
              gettime(he,mi,se,dise);
              ecritvite(' '+formate('##',he)+':'
                           +formate('##',mi)+':'
                           +formate('##',se)+' ',13,35,112);
              cursnormal;
              gotoxy(51,15);
              DELAY(100);
            UNTIL KEYPRESSED;

            c:=readkey;
            c:=upcase(c);
          until (c='O') or (c='N');


          if c='N' then begin
            repeat
              ecritvite(' Introduire l''heure  (HH:MM:SS) puis '+
                        ' '+chr(17)+chr(196)+chr(217)+' ',
                         15,20,112);

              ecritvite('   :  :   ',13,35,7);
```

```
            gotoxy(36,13);
            readln(mise);
            val(copy(mise,1,2),he,erre);
            if erre=0 then begin
              val(copy(mise,4,2),mi,erre);
              if erre=0 then val(copy(mise,7,2),se,erre);
            end;
          until erre=0;
          settime(he,mi,se,0);
          PCADRE(15,10,66,15,'Mise à jour de l''HEURE',0,false);
        end;
      until c='O';


(*-------------------------------------------------------------------*)
(* ---------------------- Mise à jour de la DATE--------------------*)
(*-------------------------------------------------------------------*)


      repeat
        PCADRE(15,10,66,15,'Mise à jour de la DATE',0,false);
        getdate(aa,mm,jj,js);

        ecritvite(' '+formate('##',jj)+':'
                     +formate('##',mm)+':'
                     +formate('####',aa)+' ',13,35,112);

        ecritvite(' Validation  (Oui/Non) :    ',15,26,112);

        cursnormal;

        gotoxy(51,15);

        repeat
          c:=readkey;
          c:=upcase(c);
        until (c='O') or (c='N');


        if c='N' then begin
          repeat
            ecritvite( ' Introduire la date (jj:MM:AAAA) puis  '
                      +chr(17)+chr(196)+chr(217)+' ',15,20,112);

            ecritvite('    :   :      ',13,35,7);

            gotoxy(36,13);
            readln(mise);
            val(copy(mise,1,2),jj,erre);
             if erre=0 then begin
              val(copy(mise,4,2),mm,erre);
              if erre=0 then val(copy(mise,7,4),aa,erre);
```

```
        end;

      until erre=0;
      setdate(aa,mm,jj);
    end;
  until c='O';
end;


begin
  if FCODE then begin
    repeat
      curscache;
      nbre:=1;
      PMENU(menu,0,0,0,0,5,nbre);
      case nbre of

        1 : PCONSULT;
        2 : PSUPP;
        4 : PPARAM;
        3 : PDH;

      end;
    until nbre=5;
  end;
end;



begin
end.
```

Chapitre 3

3.1 Program GESTION_DES_FICHIERS (éditeur)

```
program   gestion_des_fichiers;

uses
     tpcrt,
     tpedit,
     tpchaine,
     printer,
     dos,
     def,
     proc;


const
     choix : tmenu =('GESTION FICHIER','TEXTES','PHRASES',
                     'FEED-BACK','MOTS','FIN','','','');

var
     ch : byte;
```

3.1.1 Procedure PDICO

```
{ utilisée par mots }

procedure PDICO(var x : tmots;
                var p : word;
                var t : boolean);

  var
       bi,bs,y : integer;
       z       : tmots;

   begin
     bi:=0;
     bs:=filesize(fmots)-1;
     while bi<bs do begin
       y:=(bi+bs) div 2;
       seek(fmots,y);
       {$I-}
       read(fmots,z);
       {$I+}
       if ioresult<>0 then writeln(lst,x.mots,y);
       case comparechaine(x.mots,z.mots) of

         Inferieur : bs:=y-1;

         Superieur : bi:=y+1;

            Egal   : begin
                       bi:=y;
                       bs:=y;
                     end;
       end;
     end;

     seek(fmots,bi);
     read(fmots,z);
     t:=false;
     case comparechaine(x.mots,z.mots) of

       Inferieur : p:=bi;

       Superieur : p:=bi+1;

            Egal : begin
                     t:=true;
                     p:=bi;
                   end;

     end;
   end;
```

3.1.2 Procedure PDICO2

```
{ utilisée par pindex textes et phrases }

procedure PDICO2(    x : tindm;
                 var p : word);

  var
      bi,bs,y : integer;
      z       : tindm;

  begin
    bi:=0;
    bs:=filesize(findm)-1;
    while bi<bs do begin
      y:=(bi+bs) div 2;
      seek(findm,y);
      read(findm,z);
      case comparechaine(x,z) of

        Inferieur : bs:=y-1;

        Superieur : bi:=y+1;

           Egal   : begin
                      bi:=y;
                      bs:=y;
                    end;
      end;
    end;
    p:=bi;
  end;



  {$I IGTEXTES}

  {$I IGPHRASES}

  {$I IGF_B}

  {$I IGMOTS}


  begin
    videodirecte:=true;
    contrlneige:=true;
    effespaces:=false;
    attrtexte:=7;
    ch:=1;
    repeat
      PMENU(choix,0,0,0,0,5,ch);
      case ch of
```

```
        1 : ptextes;
        2 : pphrases;
        3 : pfeed_back;
        4 : pmots;
      end;
    until ch=5;

    videodirecte:=false;
    contrlneige:=false;
    clrscr;
end.
```

3.2 Include IGTEXTES

```
procedure PTEXTES;

  const
       choix : tmenu =('CONSULTATION TEXTES','TOTAL ECRAN',
                       'IMPRIMER','IMPORTER','FIN','','','','');

  var
       ch    : byte;
```

3.2.1 Procedure PIMPOR

{ importation du fichier textes.asc et transformation de celui-ci en .dat avec remise à jour de l'index }

```
procedure PIMPOR;

  var
       i,j,k,x : word;
       f       : string60;
       y       : byte;
       mo      : tmo;
       fie     : text;
       ind     : tind2;


   begin
     assign(fie,'dat\textes.asc');
     reset(fie);
     rewrite(ftextes);
     rewrite(findt);
     reset(findm);

     for k:=1 to 10 do mo[k].mots:='';
     k:=0;
     i:=0;
     y:=0;
     x:=0;
     while not eof(fie) do begin
       readln(fie,f);
       write(ftextes,f);
       if pos('#',f)<>0 then begin
         inc(i);
         ind.deb:=x;
         x:=filepos(ftextes);
         ind.nbre:=y;
         y:=1;
         if ind.nbre>0 then begin
```

```
          for k:=1 to 10 do
            pdico2(chnminusc(mo[k].mots),ind.lmo[k]);
          for k:=1 to 10 do mo[k].mots:='';
          k:=0;
          write(findt,ind);
        end;
      end
      else begin
        inc(y);
        if pos('(',f)<>0 then begin
          repeat
            inc(k);
            j:=pos('(',f);
            delete(f,j,1);
            repeat
              mo[k].mots:=mo[k].mots+f[j];
              inc(j);
            until f[j]=')';
            delete(f,j,1);
          until pos('(',f)=0;
        end;
      end;
    end;
    close(findt);
    close(findm);
    close(fie);

end;
```

3.2.2 Procedure LISTE_ECRAN

```
(*-----------------LISTE TOTALE ECRAN-----------------------------*)

      procedure  LISTE_ECRAN;

        var f               : string60;
            s               : string;
            y               : word;
            i,j             : integer;
            esc,modif       : boolean;
```

3.2.2.1 Procedure PAFFICHE

```
      procedure PAFFICHE( i,j : word);

         begin
           seek(ftextes,j);
           while (i-j<15) and (not eof(ftextes)) do begin
             read(ftextes,f);
             ecritvite(complete(f,60),6+i-j,10,7);
             inc(i);
           end;
         end;
```

3.2.2.2 Procedure PINDEX

```
      procedure PINDEX;

        var
            i,j,k,x     : word;
            f           : string60;
            y           : byte;
            mo          : tmo;
            ind         : tind2;
            fie         : text;


         begin

           assign(fie,'dat\textes.asc');
           rewrite(fie);
           rewrite(findt);
           reset(findm);
           seek(ftextes,0);

           for k:=1 to 10 do mo[k].mots:='';
           k:=0;
           i:=0;
           y:=0;
           x:=0;
```

```
  while not eof(ftextes) do begin
    read(ftextes,f);
    writeln(fie,f);
    if pos('#',f)<>0 then begin
      inc(i);
      ind.deb:=x;
      x:=filepos(ftextes);
      ind.nbre:=y;
      y:=1;
      if ind.nbre>0 then begin
        for k:=1 to 10 do
          pdico2(chnminusc(mo[k].mots),ind.lmo[k]);
        for k:=1 to 10 do mo[k].mots:='';
        k:=0;
        write(findt,ind);
      end;
    end
    else begin
      inc(y);
      if pos('(',f)<>0 then begin
        repeat
          inc(k);
          j:=pos('(',f);
          delete(f,j,1);
          repeat
            mo[k].mots:=mo[k].mots+f[j];
            inc(j);
          until f[j]=')';
          delete(f,j,1);
        until pos('(',f)=0;
      end;
    end;
  end;
  close(findt);
  close(fie);
  close(findm);

end;




begin
  modif:=false;
  clrscr;
  if filesize(ftextes)>15 then i:=filesize(ftextes)-15
  else i:=0;
  PCADRE(2,2,79,25,'EDITION textes',0,false);
  ecritvite(#204+chainecar(#205,76)+#185,22,2,7);
  changeattr(74,24,4,112);
  changeattr(74,23,4,112);
  ecritvite('BOUGER  :  '+#24+#25,24,6,112);
  ecritvite('INSER.  :  [F1]',24,30,112);
```

```
ecritvite('QUITTER :  [ESC]',24,55,112);
ecritvite('MODIF.  :  [M]',23,6,112);
ecritvite('AJOUTER :  [A]',23,30,112);
ecritvite('SUPP.   :  [SHIFT-F1]',23,55,112);
j:=i;
PAFFICHE(i,j);
i:=filesize(ftextes)-1;
gotoxy(10,20);

repeat
  cursbloc;
  c:=readkey;
  case c of


  #65,#97 : begin
              i:=filesize(ftextes)-14;
              j:=i;
              PAFFICHE(i,j);
              repeat
                i:=filesize(ftextes);
                seek(ftextes,i);
                s:='';
                litchaine('',20,10,60,$00,$70,$00,esc,s);
                f:=copy(s,1,60);
                if s<>'' then write(ftextes,f);
                if filesize(ftextes)<15 then i:=0
                else i:=filesize(ftextes)-14;
                j:=i;
                PAFFICHE(i,j);
              until (s='');
              i:=filesize(ftextes)-15;
              j:=i;
              PAFFICHE(i,j);
              modif:=true;
              inc(i,14);
              gotoxy(10,20);
            end;


 #77,#109 : begin
              seek(ftextes,i);
              read(ftextes,f);
              s:=f;
              litchaine('',6+i-j,10,60,$00,$70,$00,esc,s);
              f:=copy(s,1,60);
              ecritvite(complete(f,60),6+i-j,10,7);
              seek(ftextes,i);
              write(ftextes,f);
              inc(i);
              gotoxy(10,6+i-j);
              modif:=true;
            end;

    #0    : begin
```

```
c:=readkey;
case c of

 #59 : begin
         repeat
           for y:=filesize(ftextes)-1 downto
              i do begin
             seek(ftextes,y);
             read(ftextes,f);
             seek(ftextes,y+1);
             write(ftextes,f);
           end;
           PAFFICHE(j,j);
           s:='';
 litchaine('',6+i-j,10,60,$00,$70,$00,esc,s);
           f:=copy(s,1,60);
           if f<>'' then begin
             seek(ftextes,i);
             write(ftextes,f);
             inc(i);
             if i-j>15 then j:=i-15;
           end;
           PAFFICHE(j,j);
         until s='';
         for y:=i+1 to filesize(ftextes)-1 do
           begin
           seek(ftextes,y);
           read(ftextes,f);
           seek(ftextes,y-1);
           write(ftextes,f);
         end;
         seek(ftextes,filesize(ftextes)-1);
         truncate(ftextes);
         y:=i;
         i:=j;
         PAFFICHE(i,j);
         i:=y;
         gotoxy(10,6+i-j);
         y:=0;
         modif:=true;
       end;


 #84 : begin
         ecritvite (chain ecar(# 32,60),6+i -
j,10,7);

         for y:=i+1 to filesize(ftextes)-1 do
begin
           seek(ftextes,y);
           read(ftextes,f);
           seek(ftextes,y-1);
           write(ftextes,f);
         end;
         seek(ftextes,filesize(ftextes)-1);
         truncate(ftextes);
```

```
          y:=i;
          i:=j;
          PAFFICHE(i,j);
          ecritvite(chainecar(#32,60),20,10,7);
          i:=y;
          if i=filesize(ftextes) then begin
            gotoxy(10,5+i-j);
            dec(i);
          end
          else gotoxy(10,6+i-j);
          y:=0;
          modif:=true;
        end;


#72 :   if i-1>=0 then begin
          dec(i);
          if i-j<0 then begin
            j:=i;
            PAFFICHE(i,j);
          end;
          gotoxy(10,6+i-j);
        end;


#80 :   if i+1<filesize(ftextes) then begin
          inc(i);
          if i-j>14 then begin
            j:=i-14;
            PAFFICHE(i-14,j);
            gotoxy(10,20);
          end
          else gotoxy(10,6+i-j);
        end;

#81 :   if i<filesize(ftextes)-1 then begin
          if i+15>filesize(ftextes)-1 then
            begin
            i:=filesize(ftextes)-15;
            j:=i;
            PAFFICHE(i,j);
            i:=filesize(ftextes)-1;
          end
          else begin
            j:=i;
            PAFFICHE(i,j);
            inc(i,14);
          end;
          gotoxy(10,20);
        end;

#73 :    if i>0 then begin
           dec(i,14);
           if i<0 then i:=0;
           j:=i;
```

```
                    PAFFICHE(i,j);
                    gotoxy(10,6);
                  end;

            #119 : if i>0 then begin
                    i:=0;
                    j:=0;
                    PAFFICHE(i,j);
                    gotoxy(10,6);
                  end;


            #117 : if i<filesize(ftextes)-1 then begin
                    i:=filesize(ftextes)-15;
                    j:=i;
                    PAFFICHE(i,j);
                    i:=filesize(ftextes)-1;
                    gotoxy(10,20);
                  end;

            end;
        end;
      end;

    until c=#27;

    if modif then begin
      curscache;
      PINDEX;
    end;

end;
```

3.2.3 Procedure LISTE_IMP

{ impression des textes }

```
(*---------------LISTE TOTALE IMPRIMEE-------------------------------*)

      procedure LISTE_IMP;

        var
             phrases      : string60;
             esc          : boolean;
             deb,fin,i,j  : word;
             ind          : tind2;


         begin
           reset(findt);
           clrscr;
           PCADRE(2,2,79,25,'IMPRESSION Textes',0,false);
           ecritvite(#204+chainecar(#205,76)+#185,23,2,7);
           changeattr(74,24,4,112);
           ecritvite('QUITTER :  [ESC]',24,10,112);

           litmot('Debut d''impression : ',10,20,7,7,112,1,
                  filesize(findt),esc,deb);
           if esc then begin
             close(findt);
             exit;
           end;

           litmot('Fin   d''impression : ',12,20,7,7,112,deb,
                  filesize(findt),esc,deb);
           if esc then begin
             close(findt);
             exit;
           end;

           while not PRINTERREADY do begin
             ecritvite('erreur d''imprimante',16,20,112);
             curscache;
             c:=readkey;
             if c=#27 then begin
               close(findt);
               exit;
             end;
           end;


           writeln(lst,'');
           for i:=deb to fin do begin
             seek(findt,deb);
             read(findt,ind);

             seek(fphrases,ind.deb);
```

```
            for j:=1 to ind.nbre do begin
              read(Ftextes,phrases);
              writeln(lst,phrases);
            end;
            writeln(lst,'');
            writeln(lst,'');
          end;

        end;

(*--------------- programme " texte " principal --------------------*)


        begin
          reset(ftextes);
          repeat
            PMENU(CHOIX,0,0,0,0,4,CH);
            case ch of
              1:LISTE_ECRAN;
              2:LISTE_IMP;
              3:begin
                  curscache;
                  PIMPOR;
                end;
            end;
          until (ch=4);
          close(ftextes);
        end;
```

3.3 Include IGPHRASES

```
procedure PPHRASES;

  const
        choix : tmenu =('CONSULTATION PHRASES','TOTAL ECRAN',
                        'IMPRIMER','IMPORTER','FIN','','','','');
```

3.3.1 Procedure PIMPOR

```
{  idm textes }

  procedure PIMPOR;


    var
         i,j,k,x : word;
         f       : string60;
         y       : byte;
         ind     : tind2;
         mo      : tmo;
         fie     : text;

     begin
       assign(fie,'dat\phrases.asc');
       reset(fie);
       reset(findm);
       rewrite(fphrases);
       rewrite(findp);

       for k:=1 to 10 do mo[k].mots:='';
       k:=0;
       i:=0;
       y:=0;
       x:=0;
       while not eof(fie) do begin
         readln(fie,f);
         write(fphrases,f);
         if pos('#',f)<>0 then begin
           inc(i);
           ind.deb:=x;
           x:=filepos(fphrases);
           ind.nbre:=y;
           y:=1;
           if ind.nbre>0 then begin
             for k:=1 to 10 do
               pdico2(chnminusc(mo[k].mots),ind.lmo[k]);
             for k:=1 to 10 do mo[k].mots:='';
             k:=0;
             write(findp,ind);
           end;
```

```
      end
      else begin
        inc(y);
        if pos('(',f)<>0 then begin
          repeat
            inc(k);
            j:=pos('(',f);
            delete(f,j,1);
            repeat
              mo[k].mots:=mo[k].mots+f[j];
              inc(j);
            until f[j]=')';
            delete(f,j,1);
          until pos('(',f)=0;
        end;
      end;
    end;
    close(findm);
    close(findp);
    close(fie);

end;
```

3.3.2 Procedure LISTE_ECRAN

```
(*----------------------LISTE TOTALE ECRAN-------------------*)

      procedure LISTE_ECRAN;

        var
             f              : string60;
             s              : string;
             y              : word;
             i,j            : integer;
             esc,modif      : boolean;
```

3.3.2.1 Procedure PAFFICHE

```
      procedure PAFFICHE(i,j : word);
         begin
           seek(fphrases,j);
            while (i-j<15) and (not eof(fphrases)) do begin
              read(fphrases,f);
              ecritvite(complete(f,60),6+i-j,10,7);
              inc(i);
            end;
          end;
```

3.3.2.2 Procedure PINDEX

```
      procedure PINDEX;


        var
             i,j,k,x : word;
             fo      : integer;
             f       : string60;
             y       : byte;
             ind     : tind2;
             mo      : tmo;
             fie     : text;


         begin
           assign(fie,'dat\phrases.asc');
           rewrite(fie);
           reset(findm);
           seek(fphrases,0);
           rewrite(findp);
           seek(fphrases,0);
           seek(findp,0);

           for k:=1 to 10 do mo[k].mots:='';
           k:=0;
```

```
  i:=0;
  y:=0;
  x:=0;
  while not eof(fphrases) do begin
    read(fphrases,f);
    writeln(fie,f);
    if pos('#',f)<>0 then begin
      inc(i);
      ind.deb:=x;
      x:=filepos(fphrases);
      ind.nbre:=y;
      y:=1;
      if ind.nbre>0 then begin
        for k:=1 to 10 do
          pdico2(chnminusc(mo[k].mots),ind.lmo[k]);
        for k:=1 to 10 do mo[k].mots:='';
        k:=0;
        write(findp,ind);
      end;
    end
    else begin
      inc(y);
      if pos('(',f)<>0 then begin
        repeat
          inc(k);
          j:=pos('(',f);
          delete(f,j,1);
          repeat
            mo[k].mots:=mo[k].mots+f[j];
            inc(j);
          until f[j]=')';
          delete(f,j,1);
        until pos('(',f)=0;
      end;
    end;
  end;
  close(findp);
  close(findm);
  close(fie);
end;



begin
  modif:=false;
  clrscr;
  if filesize(fphrases)>15 then i:=filesize(fphrases)-15
  else i:=0;
  PCADRE(2,2,79,25,'EDITION phrases',0,false);
  ecritvite(#204+chainecar(#205,76)+#185,22,2,7);
  changeattr(74,24,4,112);
  changeattr(74,23,4,112);
  ecritvite('BOUGER  :  '+#24+#25,24,6,112);
  ecritvite('INSER.  :  [F1]',24,30,112);
```

```
ecritvite('QUITTER :  [ESC]',24,55,112);
ecritvite('MODIF.  :  [M]',23,6,112);
ecritvite('AJOUTER :  [A]',23,30,112);
ecritvite('SUPP.   :  [SHIFT-F1]',23,55,112);
j:=i;
PAFFICHE(i,j);
i:=filesize(fphrases)-1;
gotoxy(10,20);
repeat
 cursbloc;
 c:=readkey;
 case c of

 #65,#97 : begin
             i:=filesize(fphrases)-14;
             j:=i;
             PAFFICHE(i,j);
             repeat
               i:=filesize(fphrases);
               seek(fphrases,i);
               s:='';
               litchaine('',20,10,60,$00,$70,$00,esc,s);
               f:=copy(s,1,60);
               if s<>'' then write(fphrases,f);
               if filesize(fphrases)<15 then i:=0
               else i:=filesize(fphrases)-14;
               j:=i;
               PAFFICHE(i,j);
             until (s='');
             i:=filesize(fphrases)-15;
             j:=i;
             PAFFICHE(i,j);
             inc(i,14);
             gotoxy(10,20);
             modif:=true;
           end;

#77,#109 : begin
             seek(fphrases,i);
             read(fphrases,f);
             s:=f;
             litchaine('',6+i-j,10,60,$00,$70,$00,esc,s);
             f:=copy(s,1,60);
             ecritvite(complete(f,60),6+i-j,10,7);
             seek(fphrases,i);
             write(fphrases,f);
             inc(i);
             gotoxy(10,6+i-j);
             modif:=true;
           end;

   #0    : begin
             c:=readkey;
             case c of
```

```
                    #59 : begin
                            repeat
                              for y:=filesize(fphrases)-1 downto
                                i do
                                begin
                                seek(fphrases,y);
                                read(fphrases,f);
                                seek(fphrases,y+1);
                                write(fphrases,f);
                              end;
                              PAFFICHE(j,j);
                              s:='';
                              l i t c h a i n e (  ' ' , 6 + i  -
j,10,60,$00,$70,$00,esc,s);
                              f:=copy(s,1,60);
                              if f<>'' then begin
                                seek(fphrases,i);
                                write(fphrases,f);
                                inc(i);
                                if i-j>15 then j:=i-15;
                              end;
                              PAFFICHE(j,j);
                            until s='';
                            for y:=i+1 to filesize(fphrases)-1 do
                              begin
                              seek(fphrases,y);
                              read(fphrases,f);
                              seek(fphrases,y-1);
                              write(fphrases,f);
                            end;
                            seek(fphrases,filesize(fphrases)-1);
                            truncate(fphrases);
                            y:=i;
                            i:=j;
                            PAFFICHE(i,j);
                            i:=y;
                            gotoxy(10,6+i-j);
                            y:=0;
                            modif:=true;
                          end;

                    #84 : begin
                            ecritv ite(ch ainecar (#32,6 0),6+i -
j,10,7);
                            for y:=i+1 to filesize(fphrases)-1 do
                             begin
                              seek(fphrases,y);
                              read(fphrases,f);
                              seek(fphrases,y-1);
                              write(fphrases,f);
                            end;
                            seek(fphrases,filesize(fphrases)-1);
                            truncate(fphrases);
                            y:=i;
                            i:=j;
```

```
        PAFFICHE(i,j);
        i:=y;
        if i=filesize(fphrases) then begin
          gotoxy(10,5+i-j);
          dec(i)
        end
        else gotoxy(10,6+i-j);
        y:=0;
        modif:=true;
      end;


#72 : if i-1>=0 then begin
        dec(i);
        if i-j<0 then begin
          j:=i;
          PAFFICHE(i,j);
        end;
        gotoxy(10,6+i-j);
     end;

#80 : if i+1<filesize(fphrases) then begin
        inc(i);
        if i-j>14 then begin
          j:=i-14;
          PAFFICHE(i-14,j);
          gotoxy(10,20);
        end
        else gotoxy(10,6+i-j);
      end;

#81 : if i<filesize(fphrases)-1 then begin
        if i+15>filesize(fphrases)-1 then
           begin
          i:=filesize(fphrases)-15;
          j:=i;
          PAFFICHE(i,j);
          i:=filesize(fphrases)-1;
        end
        else begin
          j:=i;
          PAFFICHE(i,j);
          inc(i,14);
        end;
        gotoxy(10,20);
      end;

#73 :  if i>0 then begin
         dec(i,14);
         if i<0 then i:=0;
         j:=i;
         PAFFICHE(i,j);
         gotoxy(10,6);
       end;
```

```
                #119 : if i>0 then begin
                         i:=0;
                         j:=0;
                         PAFFICHE(i,j);
                         gotoxy(10,6);
                       end;


               #117 : if i<filesize(fphrases)-1 then begin
                        i:=filesize(fphrases)-15;
                        j:=i;
                        PAFFICHE(i,j);
                        i:=filesize(fphrases)-1;
                        gotoxy(10,20);
                      end;
                end;
              end;
    end;

  until c=#27;

  if modif then PINDEX;


end;
```

3.3.3 Procedure LISTE_IMP

```
  { impression des phrases }

(*---------------LISTE TOTALE IMPRIMEE-----------------------*)

      procedure LISTE_IMP;

        var
             phrases : string60;
             deb,fin : word;
             i,j     : word;
             ind     : tind2;
             esc     : boolean;

         begin
           reset(findp);
           clrscr;
           PCADRE(2,2,79,25,'IMPRESSION Phrases',0,false);
           ecritvite(#204+chainecar(#205,76)+#185,23,2,7);
           changeattr(74,24,4,112);
           ecritvite('QUITTER :  [ESC]',24,10,112);

           litmot('Debut d''impression : ',10,20,7,7,112,1,
                  filesize(findp),esc,deb);
           if esc then begin
             close(findp);
             exit;
           end;

           litmot('Fin   d''impression : ',12,20,7,7,112,deb,
                  filesize(findp),esc,deb);
           if esc then begin
             close(findp);
             exit;
           end;

           while not PRINTERREADY do begin
             ecritvite('erreur d''imprimante',16,20,112);
             curscache;
             c:=readkey;
             if c=#27 then begin
               close(findp);
               exit;
             end;
           end;

           writeln(lst,'');
           for i:=deb to fin do begin
             seek(findp,deb);
             read(findp,ind);

             seek(fphrases,ind.deb);
             for j:=1 to ind.nbre do begin
```

```
            read(fphrases,phrases);.
            writeln(lst,phrases)
          end;
          writeln(lst,'');
          writeln(lst,'');
        end;
      end;

(*---------------programme "phrases" principal --------------------*)

      var
          ch : byte;

      begin
        reset(fphrases);

        repeat
          ch:=1;
          PMENU(choix,0,0,0,0,4,ch);
          case ch of
            1:liste_ecran;
            2:Liste_imp;
            3:begin
                curscache;
                PIMPOR;
              end;
          end;
        until (ch=4);
        close(fphrases);
      end;
```

3.4 Include IGF_B

```
procedure PFEED_BACK;


  const
       choix : tmenu =('CONSULTATION FEED BACK','TOTAL ECRAN',
                       'IMPRIMER','IMPORTER','FIN','','','','');
```

3.4.1 Procedure PIMPOR

```
{ idm textes }

  procedure PIMPOR;

    var
         i,x : word;
         y   : byte;
         ind : tind;
         f   : string60;
         fie : text;

     begin
       assign(fie,'dat\f_b.asc');
       reset(fie);
       rewrite(ff_b);
       rewrite(findf_b);

       i:=0;
       y:=1;
       x:=0;
       while not eof(fie) do begin
         readln(fie,f);
         write(ff_b,f);
         if pos(#42,f)<>0 then begin
           inc(i);
           ind.deb:=x;
           x:=filepos(ff_b);
           ind.nbre:=y;
           y:=1;
           write(findf_b,ind);
         end else inc(y);
       end;
       close(findf_b);
       close(fie);
     end;
```

3.4.2 Procedure LISTE_ECRAN

```
(*----------------------LISTE TOTALE ECRAN-------------------------*)

      procedure LISTE_ECRAN;

        var
             f              : string60;
             s              : string;
             y              : word;
             i,j            : integer;
             esc,modif      : boolean;
```

3.4.2.1 Procedure PAFFICHE

```
      procedure PAFFICHE(i,j : word);
         begin
           seek(ff_b,j);
            while (i-j<15) and (not eof(ff_b)) do begin
              read(ff_b,f);
              ecritvite(complete(f,60),6+i-j,10,7);
              inc(i);
            end;
          end;
```

3.4.2.2 Procedure PINDEX

```
      procedure PINDEX;

        var
             i,x : word;
             y   : byte;
             ind : tind;
             f   : string60;
             fie : text;

         begin
           assign(fie,'dat\f_b.asc');
           rewrite(fie);
           rewrite(findf_b);
           seek(ff_b,0);
           seek(findf_b,0);

           i:=0;
           y:=1;
           x:=0;
           while not eof(ff_b) do begin
             read(ff_b,f);
             writeln(fie,f);
             if pos(#42,f)<>0 then begin
               inc(i);
               ind.deb:=x;
```

```
      x:=filepos(ff_b);
      ind.nbre:=y;
      y:=1;
      write(findf_b,ind);
    end else inc(y);
  end;
  close(fie);
  close(findf_b);
end;


begin
  modif:=false;
  clrscr;
  if filesize(ff_b)>15 then i:=filesize(ff_b)-15
  else i:=0;
  PCADRE(2,2,79,25,'EDITION FEED BACK',0,false);
  ecritvite(#204+chainecar(#205,76)+#185,22,2,7);
  changeattr(74,24,4,112);
  changeattr(74,23,4,112);
  ecritvite('BOUGER  :  '+#24+#25,24,6,112);
  ecritvite('INSER.  :  [F1]',24,30,112);
  ecritvite('QUITTER :  [ESC]',24,55,112);
  ecritvite('MODIF.  :  [M]',23,6,112);
  ecritvite('AJOUTER :  [A]',23,30,112);
  ecritvite('SUPP.   :  [SHIFT-F1]',23,55,112);
  j:=i;
  PAFFICHE(i,j);
  i:=filesize(ff_b)-1;
  gotoxy(10,20);
  repeat
   cursbloc;
   c:=readkey;
   case c of

    #65,#97 : begin
                i:=filesize(ff_b)-14;
                j:=i;
                PAFFICHE(i,j);
                repeat
                  i:=filesize(ff_b);
                  seek(ff_b,i);
                  s:='';
                  litchaine('',20,10,60,$00,$70,$00,esc,s);
                  f:=copy(s,1,60);
                  if f<>'' then write(ff_b,f);
                  if filesize(ff_b)<15 then i:=0
                 else i:=filesize(ff_b)-14;
               until s='';
               j:=i;
               PAFFICHE(i,j);
               inc(i,14);
               modif:=true;
             end;
```

```
#77,#109 : begin
             seek(ff_b,i);
             read(ff_b,f);
             s:=f;
             litchaine('',6+i-j,10,60,$00,$70,$00,esc,s);
             f:=copy(s,1,60);
             ecritvite(complete(f,60),6+i-j,10,7);
             seek(ff_b,i);
             write(ff_b,f);
             inc(i);
             gotoxy(10,6+i-j);
             modif:=true;
           end;

  #0     : begin
             c:=readkey;
             case c of

               #59 : begin
                       repeat
                         for y:=filesize(ff_b)-1 downto i
                           do begin
                           seek(ff_b,y);
                           read(ff_b,f);
                           seek(ff_b,y+1);
                           write(ff_b,f);
                         end;
                         PAFFICHE(j,j);
                         s:='';
                         litchaine('',6+i-j,10,60,
                               $00,$70,$00,esc,s);
                         f:=copy(s,1,60);
                         if f<>'' then begin
                           seek(ff_b,i);
                           write(ff_b,f);
                           inc(i);
                           if i-j>15 then j:=i-15;
                         end;
                         PAFFICHE(j,j);
                       until s='';
                       for y:=i+1 to filesize(ff_b)-1 do
                        begin
                         seek(ff_b,y);
                         read(ff_b,f);
                         seek(ff_b,y-1);
                         write(ff_b,f);
                       end;
                       seek(ff_b,filesize(ff_b)-1);
                       truncate(ff_b);
                       y:=i;
                       i:=j;
                       PAFFICHE(i,j);
                       i:=y;
                       gotoxy(10,6+i-j);
                       y:=0;
```

```
          modif:=true;
        end;


#84 : begin
        ecritvite(chainecar(#32,60),20,10,7);
        for y:=i+1 to filesize(ff_b)-1 do
         begin
          seek(ff_b,y);
          read(ff_b,f);
          seek(ff_b,y-1);
          write(ff_b,f);
        end;
        seek(ff_b,filesize(ff_b)-1);
        truncate(ff_b);
        y:=i;
        i:=j;
        PAFFICHE(i,j);
        ecritvite(chainecar(#32,60),20,10,7);
        i:=y;
        if i=filesize(ff_b) then begin
          gotoxy(10,5+i-j);
          dec(i);
        end
        else gotoxy(10,6+i-j);
        y:=0;
        modif:=true;
      end;


#72 : begin
        if i-1>=0 then begin
          dec(i);
          if i-j<0 then begin
            j:=i;
            PAFFICHE(i,j);
            gotoxy(10,6);
          end;
          gotoxy(10,6+i-j);
       end;
     end;


#80 : if i+1<filesize(ff_b) then begin
        inc(i);
        if i-j>14 then begin
          j:=i-14;
          PAFFICHE(i-14,j);
          gotoxy(10,20);
        end else gotoxy(10,6+i-j);
      end;


#81 : if i<filesize(ff_b) then begin
```

```
                        if i+15>filesize(ff_b)-1 then begin
                          i:=(filesize(ff_b)-16);
                          j:=i;
                          PAFFICHE(i,j);
                          i:=filesize(ff_b)-1;
                        end
                        else begin
                          j:=i;
                          PAFFICHE(i,j);
                          inc(i,14);
                        end;
                        gotoxy(10,20);
                      end;


          #73 :   if i>0 then begin
                    dec(i,14);
                    if i<0 then i:=0;
                    j:=i;
                    PAFFICHE(i,j);
                    gotoxy(10,6);
                  end;


          #119 : if i>0 then begin
                   i:=0;
                   j:=0;
                   PAFFICHE(i,j);
                   gotoxy(10,6);
                 end;


          #117 : if i<filesize(ff_b) then begin
                   i:=filesize(ff_b)-15;
                   j:=i;
                   PAFFICHE(i,j);
                   i:=filesize(ff_b)-1;
                   gotoxy(10,20);
                 end;
        end;
      end;
    end;

  until c=#27;

  if modif then Pindex;

end;
```

3.4.3 Procedure LISTE_IMP

```
(*---------------LISTE TOTALE IMPRIMEE----------------------------*)

      procedure LISTE_IMP;

        var
             phrases      : string60;
             i,j,deb,fin  : word;
             ind          : tind;
             esc          : boolean;

         begin
           clrscr;
           reset(findf_b);
           PCADRE(2,2,79,25,'IMPRESSION Feed Back',0,false);
           ecritvite(#204+chainecar(#205,76)+#185,23,2,7);
           changeattr(74,24,4,112);
           ecritvite('QUITTER :  [ESC]',24,10,112);

           litmot('Debut d''impression : ',10,20,7,7,112,1,
                  filesize(findf_b),esc,deb);
           if esc then begin
             close(findf_b);
             exit;
           end;

           litmot('Fin   d''impression : ',12,20,7,7,112,deb,
                  filesize(findf_b),esc,deb);
           if esc then begin
             close(findf_b);
             exit;
           end;

           while not PRINTERREADY do begin
             ecritvite('erreur d''imprimante',16,20,112);
             curscache;
             c:=readkey;
             if c=#27 then begin
               close(findf_b);
               exit;
             end;
           end;

           write(lst,'');
           for i:=deb to fin do begin
             seek(findf_b,i-1);
             read(findf_b,ind);
             seek(ff_b,ind.deb);
             for j:=1 to ind.nbre do begin
               read(Ff_b,phrases);
               writeln(lst,i:5,' ',phrases)
             end;
             writeln(lst,'');
```

```
          end;
        end;

    (*---------------programme      "feed      back"      principal
--------------*)

        var
             ch : byte;

        begin
          reset(ff_b);
          ch:=1;
          repeat
            PMENU(choix,0,0,0,0,4,ch);
            case ch of
              1:liste_ecran;
              2:Liste_imp;
              3:begin
                  curscache;
                  PIMPOR;
                end;
            end;
          until (ch=4);
          close(ff_b);
        end;
```

3.5 Include IGMOTS

```
procedure pmots;

  const
       choix : tmenu =('CONSULTATION mots','TOTAL ECRAN',
                       'TOTAL IMPRIME','FIN','','','','','');
  var
       ch   : byte;
```

3.5.1 Procedure LISTE_ECRAN

```
procedure LISTE_ECRAN;

  var
       mot              : tmots;
       y,p              : word;
       i,j              : integer;
       ind              : tind;
       t,modif          : boolean;
```

3.5.1.1 Procedure INDEX

```
procedure INDEX;

  var
       i    : word;
       mo   : tmots;


   begin
     rewrite(findm);
     seek(fmots,0);
     for i:=0 to filesize(fmots)-1 do begin
       read(fmots,mo);
       write(findm,mo.mots);
     end;
     close(findm);
   end;
```

3.5.1.2 Procedure PAFFICHE

```
procedure PAFFICHE(mo:tmots);

  var
       y,p : byte;
```

```
begin

  ecritvite('MOT  : '+complete(mot.mots,16),10,5,7);

  ecritvite('SQUE : '+complete(mot.sque,60),12,5,7);

  ecritvite('ADR  : '+chainecar(#32,60),14,5,7);
  ecritvite(          chainecar(#32,60),16,10,7);

  y:=0;
  repeat
    inc(y);
    if y>9 then p:=1 else p:=0;
    ecritvite(longenchn(mot.adr[y]),14+(2*p),5+(7*(y-(9*p))),7);
  until (y=15) or (mot.adr[y]=0);
end;
```

3.5.1.3 Procedure PAJOUT

```
procedure PAJOUT;

  var
       i,y,pl    : word;
       esc,esc1  : boolean;
       ph        : string;
       mot,mot2  : tmots;

   begin
     repeat

       ecritvite('MOT  : '+chainecar(#32,16),10,5,7);
       ecritvite('SQUE : '+chainecar(#32,61),12,5,7);
       ecritvite('ADR  : '+chainecar(#32,60),14,5,7);
       ecritvite(          chainecar(#32,60),16,10,7);

       mot.mots:='';
       mot.sque:='';
       for i:=1 to 10 do mot.adr[i]:=0;

       repeat
         ph:=mot.mots;
         esc:=false;
         litchaine('MOT  : ',10,5,15,$07,$70,$00,esc,ph);

         if esc then exit;
         mot.mots:=copy(ph,1,15);

         pdico(mot,pl,t);
         if not t then begin

           repeat
             ph:=mot.sque;
             litchaine('SQUE : ',12,5,60,$07,$70,$00,esc,ph);
```

```
            if not esc then begin
              mot.sque:=copy(ph,1,60);

              ecritvite('adr  : '+chainecar(#32,60),14,5,7);

              i:=0;
              esc1:=false;
              repeat
                if esc1 and (i>1) then i:=i-2;
                inc(i);
                if i>9 then p:=1 else p:=0;

                litmot('',14+(2*p),5+(7*(i-(9*p))),5,
                $00,$70,0,30000,esc1,mot.adr[i]);

              until (((mot.adr[i]=0) or (i=15)) and not esc1)
              or  ((i=1) and esc1);
            end;

          until not esc1 or esc;
        end
        else begin
          pprendecr(true);
          perreur('deja dans la liste',20,10,1000);
          pprendecr(false);
          exit;
        end;

        until not esc;

        if pl<filesize(fmots) then
          for y:=filesize(fmots)-1 downto pl do begin
            seek(fmots,y);
            read(fmots,mot2);
            seek(fmots,y+1);
            write(fmots,mot2);
          end;
          seek(fmots,pl);
          write(fmots,mot);

    until ph='';
  end;
```

3.5.1.4 Procedure PMODIF

```
procedure PMODIF(var mot : tmots;
                     j   : word);


  var
       i,y,p       : word;
       esc,esc1,t  : boolean;
       ph          : string;
```

```
    mot2        : tmots;

begin

  mot2:=mot;

  repeat
    ph:=mot2.mots;
    litchaine('MOT   : ',10,5,15,$07,$70,$00,esc,ph);

    if esc then exit;
    mot.mots:=copy(ph,1,15);

    repeat
      ph:=mot2.sque;
      litchaine('SQUE  : ',12,5,60,$07,$70,$00,esc,ph);
      if not esc then begin
        mot.sque:=copy(ph,1,60);

        ecritvite('adr   : '+chainecar(#32,60),14,5,7);

        i:=0;
        repeat
          if esc1 and (i>1) then dec(i,2);
          inc(i);
          if i>9 then p:=1 else p:=0;

          litmot('',14+(2*p),5+(7*(i-(9*p))),5,
                 $00,$70,0,30000,esc1,mot.adr[i]);

        until (((mot.adr[i]=0) or (i=15)) and not esc1) or
          ((i=1) and esc1);
      end;

    until esc or (mot2.sque<>mot.sque) or (mot.adr[i]=0) or
             (i=15);

  until not esc;

  if mot.mots<>mot2.mots then begin
    for y:=j+1 to filesize(fmots)-1 do begin
      seek(fmots,y);
      read(fmots,mot2);
      seek(fmots,y-1);
      write(fmots,mot2);
    end;
    seek(fmots,filesize(fmots)-1);
    truncate(fmots);
    PDICO(mot,p,t);

    if p<filesize(fmots) then
      for y:=filesize(fmots)-1 downto p do begin
        seek(fmots,y);
        read(fmots,mot2);
```

```
        seek(fmots,y+1);
        write(fmots,mot2);
      end;
      seek(fmots,p);
      write(fmots,mot);
  end
  else begin
    seek(fmots,j);
    write(fmots,mot);
  end;
  PAFFICHE(mot);
  modif:=true;
end;


begin
  modif:=false;
  clrscr;
    PCADRE(2,2,79,25,'EDITION mots',0,false);
  ecritvite(#204+chainecar(#205,76)+#185,22,2,7);
  changeattr(74,24,4,112);
  changeattr(74,23,4,112);
  ecritvite('BOUGER  :  '+#24+#25,24,6,112);
  ecritvite('QUITTER :  [ESC]',24,55,112);
  ecritvite('MODIF.  :  [M]',23,6,112);
  ecritvite('AJOUTER :  [A]',23,30,112);
  ecritvite('SUPP.   :  [SHIFT-F1]',23,55,112);
  if filesize(fmots)>0 then begin
    i:=filesize(fmots)-1;
    seek(fmots,i);
    read(fmots,mot);
    PAFFICHE(mot);
     end
  else i:=0;
  repeat
   c:=readkey;
   case c of

  #65,#97 : begin
              PAJOUT;
              PAFFICHE(mot);
              modif:=true;
            end;

  #77,#109 : begin
              PMODIF(mot,i);
              modif:=true;
              PAFFICHE(mot);
            end;

     #0    : begin
               c:=readkey;
               case c of
```

```
                #84 : begin
                        for y:=i+1 to filesize(fmots)-1 do
                          begin
                          seek(fmots,y);
                          read(fmots,mot);
                          seek(fmots,y-1);
                          write(fmots,mot);
                        end;
                        seek(fmots,filesize(fmots)-1);
                        truncate(fmots);
                        seek(fmots,i);
                        read(fmots,mot);
                        PAFFICHE(mot);
                        modif:=true;
                      end;

                #72 : if i-1>=0 then begin
                        dec(i);
                        seek(fmots,i);
                        read(fmots,mot);
                        PAFFICHE(mot);
                     end;

               #80 : if i+1<filesize(fmots) then begin
                        inc(i);
                        seek(fmots,i);
                        read(fmots,mot);
                        PAFFICHE(mot);
                      end;


               #71 : if i>0 then begin
                        i:=0;
                        seek(fmots,i);
                        read(fmots,mot);
                        PAFFICHE(mot);
                      end;


               #79 : if i<filesize(fmots) then begin
                        i:=filesize(fmots)-1;
                        seek(fmots,i);
                        read(fmots,mot);
                        PAFFICHE(mot);
                      end;
             end;
           end;
    end;

  until c=#27;

  if modif then INDEX;

end;
```

3.5.2 Procedure LISTE_IMP

```
(*---------------    LISTE TOTALE IMPRIMEE  ----------------------*)

      procedure LISTE_IMP;
        var
             mot : tmots;
             j   : word;
             i   : byte;

         begin

           if not PRINTERREADY then exit;

           seek(fmots,0);
           for j:=1 to filesize(fmots) do begin
             read(fmots,mot);
             write(lst,#27,#71);
             write(lst,complete(mot.mots,15));
             write(lst,#27,#72);

             writeln(lst,mot.sque);
             write(lst,'                ');
             i:=1;
             while mot.adr[i]>0 do begin
               write(lst,mot.adr[i]:5,'  ');
               inc(i);
             end;
             writeln(lst);
             writeln(lst);
           end;
         end;

(*--------------- programme " mots "  principal  -------------------*)


         begin
           reset(fmots);
           ch:=1;
           repeat
             PMENU(choix,0,0,0,0,3,ch);
             case ch of
               1:liste_ecran;
               2:Liste_imp;
             end;
           until (ch=3);

           close(fmots);
         end;
```

Table des Matières