9/10.1.1

Gestion de compte Turbo

Vous avez certainement utilisé, avec intérêt, le programme de gestion de compte, réalisé en Basic, proposé en Partie 9 chapitre 10.1.

Nous vous proposons ici d'en améliorer le fonctionnement et d'en combler quelques lacunes.

LES PROBLÈMES POSÉS

Certains, parmi ceux qui ventilent beaucoup de chèques ou ceux qui ne ferment pas suffisamment tôt leur fichier, auront remarqué les limites de ce logiciel : non à cause de sa conception, nous mettrons plutôt en cause le Basic de l'AMSTRAD qui, bien que performant, se révèle lent pour des écritures conséquentes.

La lenteur du traitement serait encore acceptable, mais la gestion du fichier est bien plus gênante : le BASIC LOCOMOTIVE ne gère que les fichiers à accès séquentiel, c'est-à-dire que, lors de l'utilisation du fichier, le programme doit charger le fichier complet, ce qui demande du temps et de nombreux accès disquette pour un fichier proche de la fin de l'année.

De plus, cette fonctionnalité implique une taille du fichier limitée à la mémoire disponible après chargement du programme et initialisation du programme. Impossible donc de travailler sur un fichier de deux ans ou plus.

Le problème aurait été résolu si le Basic possédait la gestion des fichiers en mode dit aléatoire, encore appelé à accès direct, ou indexé, c'est-à-dire que l'on accède aux données directement sur la disquette.

La gestion devient alors plus rapide, puisqu'il suffit de lire ou d'enregistrer uniquement les données concernées par un travail.

La taille des fichiers peut aussi être plus importante, ainsi l'on pourrait, si la disquette est formattée au format DATA, et qu'elle ne contient que le fichier, atteindre la valeur de 178 k-OCTETS d'écritures, sur le lecteur de disquette d'origine. Un fichier de près de 700 k-Octets sera possible si vous utilisez un lecteur additionnel de capacité plus importante.

LA SOLUTION

Nous vous proposons de résoudre tous ces problèmes par l'utilisation du logiciel Turbo Pascal, qui nous permettra de créer un programme exécutable en langage machine, donc plus rapide, et possédant la gestion de fichier à accès direct qui nous manque sous Basic.

L'utilisation s'effectuant sur le système d'exploitation CP/M 2.2, il sera possible de procéder à un lancement automatique du programme de gestion de compte dès la frappe de la commande **ùCPM** (ou **¦CPM**).

PRÉSENTATION DU LOGICIEL

Nous ne nous étendrons pas longuement sur l'utilisation du logiciel, la présentation étant quasi identique à celle proposée dans le programme Basic.

Rappelons rapidement la procédure :

— D'abord ouvrir un compte par le choix 0, et introduire toutes les caractéristiques.

— Si le compte a déjà été ouvert, le sélectionner.

— Arrivent ensuite les choix de travail :

- — Lecture
- — Insertion
- — Recherche numéro
- — Mise à jour
- — Sauvegarde(fin)
- — Suite

Ce dernier choix permet de fermer le fichier en cours, et d'en ouvrir un nouveau, ou encore de travailler sur un fichier déjà fermé.

Pour les choix précédents, nous vous conseillons de vous reporter Partie 9, chapitre 10.1 pour des explications plus détaillées. De plus, le logiciel est lui-même suffisamment documenté pour une utilisation autonome rapide.

REMARQUE IMPORTANTE ET PRÉPARATIFS

Nous commencerons par une remarque importante, et qui va décevoir les utilisateurs de CP/M 3.0 : afin de contenter les utilisateurs de CPC 464, munis d'un lecteur de disquette, et de CPC 664, nous avons été obligés de créer ce logiciel sous CP/M 2.2.

Cette contrainte nous a tout de même apporté un avantage : nous avons utilisé les vecteurs de sauts à certaines routines de la ROM interne, qui ne sont pas présents sous CP/M 3.0, afin d'utiliser au mieux les performances graphiques de votre AMSTRAD CPC.

Vous pouvez maintenant préparer une disquette formattée au format système, sous CP/M 2.2, à l'aide de l'utilitaire DISCKIT2.COM.

Vous effectuerez ensuite une copie de la version CP/M 2.2 de Turbo Pascal. Seuls les fichiers TURBO.COM et TURBO.OVR seront dupliqués. Le fichier TURBO.MSG ne sera pas utile, car nous ne pourrons pas l'utiliser à cause de la taille des fichiers à compiler.

Lancez enfin TURBO.COM, et répondez No à la question :

Include Error Messages (Y/N) ?

Vous pouvez maintenant entrer les fichiers qui suivent à l'aide de l'éditeur.

LE PROGRAMME PRINCIPAL DE LANCEMENT : COMPTE.PAS

Voici le programme principal de gestion qui définit toutes les variables globales (variables chaînes, entières, réelles, booléennes, et caractères).

Ce programme permet de plus l'insertion des fichiers inclus :

— COULEUR.INC
— CADRE.INC
— ENREGISTR.INC
— MESSAGE.INC
— AFFICHE.INC
— INSERT.INC
— REPERT1.INC
— REPERT2.INC
— RECHERCH.INC
— MISEJOUR.INC
— SAUVFICH.INC
— SUITE.INC
— TRAVAIL.INC
— CHOIX.INC
— OUVCCP.INC
— MENU.INC

Il lance enfin la procédure de départ MENU_1.

En fin de procédure, le programme reviendra au clscr, pour effacer l'écran et rendra la main à CP/M par le **END.** final.

```
{===================================}
Program Gestion_d_un_compte_bancaire;
{===================================}

          {programme principal
           definition des variables
           inclusion des procedures
           et lancement de la gestion
                                    }

          {------------------------}
          {definition des variables}
          {------------------------}

var
```

```
      {------------------}
      {variables chaines}
      {------------------}

nom_du_fichier      : string[4];
nom_du_compte       : string[3];
chaine              : string[40];
date                : string[9];

      {-------------------}
      {variables entieres}
      {-------------------}

caractere           : integer;
champ               : integer;
erreur              : integer;
fin                 : integer;
indice_de_travail   : integer;
ligne_d_affichage   : integer;
nombre_de_pages     : integer;
page_a_lire         : integer;
variable            : integer;

      {------------------}
      {variables reelles}
      {------------------}

variable_reelle     : real;
dernier_avoir       : real;

      {---------------------}
      {variables booleennes}
      {---------------------}

numero_trouve       : boolean;
ouv_compte          : boolean;
tableau_rempli      : boolean;
variable_booleenne  : boolean;
fichier_principal   : boolean;

      {---------------------}
      {variables caracteres}
      {---------------------}

reponse             : char;

         {-----------------------------}
         {insertion des fichier inclus}
         {-----------------------------}

{$I couleur.inc}
         {couleur des encres}

{$I cadre.inc}
         {dessin du cadre}

{$I enregist.inc}
         {definition d'un enregistrement}

{$I message.inc}
         {messages a afficher}
```

```
{$I affiche.inc}
        {affichage page compte}

{$I insert.inc}
        {insertion d'une ligne}

{$I repert1.inc}
        {affichage des pages calcul}

{$I repert2.inc}
        {recherche des pages calcul}

{$I recherch.inc}
        {recherche d'un numero d'operation}

{$I misejour.inc}
        {mise a jour d'une operation}

{$I sauvfich.inc}
        {sauvegarde du fichier}
{$I suite.inc}
        {suite demandee au menu}

{$I travail.inc}
        {travail sur un fichier ferme}

{$I choix.inc}
        {selection de choix au menu}

{$I ouvccp.inc}
        {ouverture de compte}

{$I menu.inc}
        {affichage du menu}

        {--------------------}
        {programme principal}
        {--------------------}

begin
    menu_1;
    clrscr;
        {fin d'utilisation

             effacement ecran
             et retour a CPM
                              }
    end.
```

LES FICHIERS

COULEUR.INC

Ce fichier définit les procédures permettant le choix d'encres, de couleurs de bord, du stylo, du fond, et la gestion des fenêtres (ouverture et fermeture).

ink(n1,c1,c2) permet de définir l'encre **n1** avec les couleurs **c1** et **c2**.

border(c1,c2) définit les couleurs de bord comme étant de valeurs **c1** et **c2**.

papier(x1) définit la couleur du fond à la valeur de l'encre **x1**.

stylo(x) choisit l'encre x pour la couleur du stylo.

mode(x) permet de changer le mode écran.

window(x1,x2,y1,y2) définit la taille d'une fenêtre à l'aide des coordonnées x1, x2, y1 et y2.

window0 permet de retourner à la fenêtre initiale utilisant tout l'écran.

window_tableau définit la fenêtre active pour l'affichage dans la feuille de calcul de la gestion.

efface_ligne_1 possède un nom suffisamment explicite pour se passer de commentaires.

```
{===========}
{couleur.inc}
{===========}

   {procedures definies:
                          - ink
                          - border
                          - papier
                          - stylo
                          - mode
                          - window
                          - window0
                                    }

    {*il faut imperativement donner 2 couleurs pour l'encre ou
      pour le bord*}

        {definition de couleur d'encre}
     procedure ink(n1,c1,c2:integer);
          begin
          write(#28,chr(n1),chr(c1),chr(c2));
          end;

        {definition du bord ecran}
    procedure border(c1,c2:integer);
          begin
          write(#29,chr(c1),chr(c2));
          end;
```

```
      {definition couleur papier}
  procedure papier(x1:integer);
            begin
            write(#14,chr(x1));
        end;

      {definition du pen utilise}
  procedure stylo(x:integer);
        begin
        write(#15,chr(x));
        end;

      {definition du mode utilise}
 procedure mode(x:integer);
 begin
 if x<0 then x:=0;
 if x>2 then x:=2;
 write(#4,chr(x));
 clrscr;
 end;

      {definition d'une taille de fenetre}
procedure window(x1,x2,y1,y2:integer);
begin
write (#26,chr(x1-1),chr(x2-1),chr(y1-1),chr(y2-1));
end;

      {retour en fenetre initiale}
procedure window0;
begin
write(#26,chr(0),chr(79),chr(0),chr(24));
end;

      {definition de la fenetre de la feuille}
procedure window_tableau;
begin
window(2,79,5,24);
end;

      {effacement premiere ligne}
procedure efface_ligne_1;
begin
     window(1,80,1,1);
     clrscr;
     window0;
     write(chr(2));
     gotoxy(1,1);
 end;
```

CADRE.INC

Ce fichier contient les procédures qui permettront de dessiner le cadre contenant la feuille de calcul, ainsi que les messages de cette feuille.

Il utilise les vecteurs d'accès à la ROM du système d'exploitation en **&BBC0**, **&BBC3** et **&BBF9**.

move définit le vecteur situé en &BBC0 pour effectuer un déplacement vers une position absolue.

mover définit le vecteur situé en &BBC3 pour effectuer un déplacement vers une position relative par rapport à la position actuelle.

trace définit le vecteur situé en &BBF9 pour tracer une ligne depuis la position actuelle à une position relative définie par la distance.

ligne_horizontale trace une ligne horizontale à partir des valeurs chargées dans les registres à l'aide de la procédure prédéfinie **inline**.

ligne_verticale effectue de même pour les lignes verticales.

messages affiche les messages de la feuille (Date, Rentré le, Numéro, N.Avoir, Objet, Débit et Crédit) en ligne 3 de l'écran, pour le repérage des champs de la feuille de calcul.

lignes_centrales trace les lignes centrales de la feuille de calcul.

cadre trace le cadre de la feuille de calcul en utilisant les procédures précédentes.

```
{=========}
{cadre.inc}
{=========}

          {appel des vecteurs existants sous CPM 2.2}

procedure move ;external $bbc0;
          {vecteur move absolu}

procedure mover ; external $bbc3;
          {vecteur move relatif}

procedure trace; external $bbf9;
          {vecteur trace de ligne}

          {----------------------------------------}

procedure ligne_horizontale;
begin
     inline($11/>4);
     move;
     inline($21/>0/$11/630);
     trace;
end;

procedure ligne_verticale;
begin
     inline($21/>4);
     move;
```

```
      inline($21/368/$11/>0);
      trace;
      inline($21/-368/$11/>1);
      mover;
      inline($21/368/$11/>0);
      trace;
end;

procedure messages;
begin
      gotoxy(4,3);
      write('Date');
      gotoxy(71,3);
      write('Rentre le');
      gotoxy(13,3);
      write('Numero');
      gotoxy(61,3);
      write('N.Avoir');
      gotoxy(26,3);
      write('Objet');
      gotoxy(50,3);
      write('Debit');
      gotoxy(39,3);
      write('Credit');
end;
          {messages en ligne superieure}

procedure lignes_centrales;
begin
      inline($11/>163);
      ligne_verticale;
      inline($11/>459);
      ligne_verticale;
      inline($11/>283);
      ligne_verticale;
      inline($11/>371);
      ligne_verticale;
end;

procedure cadre;
begin
      inline($21/>4);
      ligne_horizontale;
      inline($11/>4);
      ligne_verticale;
      inline($21/372);
      ligne_horizontale;
      inline($11/635);
      ligne_verticale;
      inline($21/344);
      ligne_horizontale;
      inline($11/>83);
      ligne_verticale;
      inline($11/>555);
      ligne_verticale;
      lignes_centrales;
      messages;
end;
```

ENREGIST.INC

Ce programme définit le fichier sur lequel la gestion de compte va se faire.

Le type de **fichier_de_travail** est défini en tant que fichier à accès direct, puis le **fichier_de_travail_courant** est défini sous ce type de fichier ; les variables utilisées seront **fichier** et **rubrique**.

On effectue de même pour **intitule** et **fichier_intitule**. Les variables sont ensuite définies : **intitule_compte**, **int_compte**.

Un nouveau type de fichier est utilisé : **variable_compte** sur lequel on définit **fichier_variable_compte** ; et dont les variables sont : **compte_variable** et **varcompte**.

enregistrement_de_l_intitule permettra la sauvegarde de l'intitulé lors de l'ouverture du compte.

lecture_intitule lira l'intitulé pour une utilisation ultérieure.

enregistrement_variable est la procédure qui sauvegarde les variables du compte.

lire_fichier_des_variables lira la variable **varcompte**.

enregistrement(champ) permettra de sauvegarder le champ défini.

lecture(champ) réalise l'opération de lecture sur un champ défini lors d'un travail sur la feuille de calcul.

```
{============}
{enregist.inc}
{============}

{definition du fichier de travail}

type
        fichier_de_travail = record
          date          : string[9];
          numero        : string[9];
          objet         : string[14];
          operation     : real;
          nouvel_avoir  : real;
          rentre_le     : string[9];
        end;
          fichier_de_travail_courant = file of fichier_de_travail;

var
          fichier       : fichier_de_travail_courant;
          rubrique      : fichier_de_travail;

{.....................................}

{definition des variables pour l'intitule}

type
        intitule = record
          centre        : string[12];
          cc            : string[15];
          nom           : string[28];
```

```
          intitule1     : string[28];
          intitule2     : string[28];
          ville         : string[28];
        end;
          fichier_intitule = file of intitule;

var
          intitule_compte : fichier_intitule;
          int_compte      : intitule;

{.....................................}

{definition des variables pour le fichier contenant
                - l'index des operations sans numero
                - le nom du fichier courant
                - somme indiquee par le dernier releve de la banque
                - la date de mise a jour
                - par....
                                              }

type
        variable_compte = record
          index_osn             : integer;
          fichier_courant       : string[4];
          dif_rent              : real;
          date_de_mise_a_jour   : string[9];
          mise_a_jour_par       : string[15];
        end;
          fichier_variable_compte = file of variable_compte;

var
          compte_variable : fichier_variable_compte;
          varcompte       : variable_compte;

{.....................................}

procedure enregistrement_de_l_intitule;

begin
     rewrite(intitule_compte);
     reset(intitule_compte);
     write(intitule_compte,int_compte);
     close(intitule_compte);
end;

procedure lecture_intitule;

begin
     reset(intitule_compte);
     read(intitule_compte,int_compte);
     close(intitule_compte);
end;

procedure enregistrement_variables;

begin
     assign(compte_variable,'var'+nom_du_compte+'.dat');
     rewrite(compte_variable);
     reset(compte_variable);
     write(compte_variable,varcompte);
     close(compte_variable);
end;
```

```
procedure lire_fichier_des_variables;

begin
     reset(compte_variable);
     read(compte_variable,varcompte);
     close(compte_variable);
end;

procedure enregistrement(champ : integer);

begin
     seek(fichier,champ);
     write(fichier,rubrique);

end;

procedure lecture(champ : integer);

begin
     seek(fichier,champ);
     read(fichier,rubrique);

end;
```

MESSAGE.INC

Ce fichier définit les procédures de positionnement sur l'écran et d'acquisition d'un message au clavier.

print_using (colonne, ligne, longueur) effectue le positionnement en ligne-colonne pour définir une fenêtre à la largeur « longueur ».

entree_message (colonne, ligne, longueur) est une procédure permettant l'acquisition en position colonne-ligne d'un message de longueur prédéfinie.

entree_chaine (colonne, ligne, longueur) effectue de même pour une chaîne utilisée par la feuille de calcul.

entree_nombre (colonne, ligne, longueur) permettra l'acquisition d'un nombre représentant la longueur multipliée par un chiffre x.

entree_nombre sans formattage (colonne, ligne, longueur) effectue de même pour un nombre sans format précis.

```
{===========}
{message.inc}
{===========}

procedure print_using(colonne,ligne,longueur: integer);

begin
     window(colonne,colonne+longueur-1,ligne,ligne);
     str(variable_reelle,chaine);
     if chaine[16]='+' then chaine:=copy(chaine,17,2)
                       else chaine:=copy(chaine,16,3);
     val(chaine,variable,erreur);
     clrscr;
     window0;
     if variable_reelle<0 then erreur:=-1
                          else erreur:=0;
     if variable<1 then
        begin
             gotoxy(colonne+longueur+erreur-3,ligne);
             write(variable_reelle:1:2);
        end
                  else
        begin
             gotoxy(colonne+longueur+erreur-variable-3,ligne);
             write(variable_reelle:variable:2);
        end;
end;

procedure entree_message(colonne,ligne,longueur : integer);

begin
     window(colonne,colonne+longueur-1,ligne,ligne);
     lowvideo;
     write(chr(2));
     chaine:='';
     reponse:=chr($7f);
     for variable:=1 to longueur do chaine:=chaine+' ';
     variable:=0;
     repeat
        if reponse=chr($7f) then
                             begin
                               chaine:=' '+copy(chaine,1,longueur-1);
                               variable:=variable-1;
                               if variable<0 then variable:=0;

                             end
                            else
                             begin
                               chaine:=copy(chaine,2,longueur)
                                       +reponse;
                               variable:=variable+1;
                               if variable>longueur then
                                                  variable:=longueur;
                             end;
        gotoxy(1,1);
        write(chaine);
        read(kbd,reponse);
     until reponse=chr(13);
     write(chr(3));
```

```
     normvideo;
     chaine:=copy(chaine,longueur-variable+1,variable+1)

end;

{-------------------------------------------------------}
{procedure permettant l'entree d'une chaine de caracteres}
{-------------------------------------------------------}

procedure entree_chaine(colonne,ligne,longueur : integer);

begin
     entree_message(colonne,ligne,longueur);
     window(colonne,colonne+longueur,ligne,ligne);
     clrscr;
     write(chaine);
     window0;
     if ouv_compte=false then
                                 begin
                                      gotoxy(colonne+longueur,ligne);
                                      write(chr(149));
                                  end;
end;

{---------------------------------------}
{procedure permettant l'entree d'un nombre}
{---------------------------------------}

procedure entree_nombre(colonne,ligne,longueur : integer);

begin
   variable_reelle:=0;
   repeat
        repeat
             entree_message(colonne,ligne,longueur);
             val(chaine,variable_reelle,erreur);
        until erreur=0;
   until abs(variable_reelle)<1E+07;
end;

procedure entree_nombre_sans_formatage(colonne,ligne,longueur
                                                        : integer);

begin
     repeat
           entree_message(colonne,ligne,longueur);
           val(chaine,variable_reelle,erreur);
     until erreur=0;
     window(colonne,colonne+longueur,ligne,ligne);
     clrscr;
     gotoxy(1,1);
     write(variable_reelle:5:2);
end;
```

AFFICHE.INC

Ce fichier définit des procédures permettant d'afficher les champs du fichier de compte, en réécrivant les éventuelles lignes verticales effacées, ainsi que les scrolling lors d'un affichage au bas de la feuille de calcul.

caractere_149 (colonne) affiche le caractère **chr$** (149), composé d'un trait vertical, dans la colonne spécifiée par la variable locale.

traits_verticaux réaffiche les traits verticaux sur une ligne de la feuille de calcul.

efface ligne_d_affichage est la procédure qui permet d'effacer la ligne d'affichage en cours.

scrolling_haut effectue la montée d'une ligne dans la fenêtre de la feuille de calcul.

affichage_champ (champ) est la procédure qui lit le champ et permet son affichage dans le champ spécifié.

```
{===========}
{affiche.inc}
{===========}

procedure caractere_149(colonne : integer);

begin
     gotoxy(colonne,ligne_d_affichage);
     write(chr(149));
         {affichage trait vertical}
end;

procedure traits_verticaux;

begin
     caractere_149(11);
     caractere_149(70);
     caractere_149(21);
     caractere_149(58);
     caractere_149(36);
     caractere_149(47);
         {affichege trait vertical en colonne designee}
end;

procedure efface_ligne_d_affichage;

begin
     window(2,79,ligne_d_affichage,ligne_d_affichage);
     clrscr;
     window0;
     traits_verticaux;
end;

procedure scrolling_haut;

begin
     window_tableau;
     gotoxy(79,24);
```

```
      write;
      window0;
      traits_verticaux;
end;

procedure affichage_champ(champ : integer);

begin
      if (ligne_d_affichage=24) and (tableau_rempli=true) then
                                                    scrolling_haut;
      if ligne_d_affichage=24 then tableau_rempli:=true;
      lecture(champ);
      gotoxy(2,ligne_d_affichage);
      write(rubrique.date);
      gotoxy(12,ligne_d_affichage);
      write(rubrique.numero);
      gotoxy(22,ligne_d_affichage);
      write(rubrique.objet);
      variable_reelle:=rubrique.operation;
      if variable_reelle<0 then

                              begin


                                  variable_reelle:=-variable_reelle;
                                  print_using(48,ligne_d_affichage,9)
                              end
                           else print_using(37,ligne_d_affichage,9);
      variable_reelle:=rubrique.nouvel_avoir;
      print_using(59,ligne_d_affichage,10);
      gotoxy(71,ligne_d_affichage);
      write(rubrique.rentre_le);
      if ligne_d_affichage<>24 then
                              ligne_d_affichage:=ligne_d_affichage+1;
end;
```

INSERT.INC

Ce fichier contient la procédure insertion, qui permet l'ajout d'une nouvelle écriture dans le fichier de la gestion de compte.

```
{==========}
{insert.inc}
{==========}

procedure insertion;

begin
    efface_ligne_1;
    gotoxy(35,1);
    lowvideo;
    write(' Insertion ');
    normvideo;
    if (ligne_d_affichage=24) and (tableau_rempli=true) then
                                                  scrolling_haut;
```

```
entree_chaine(2,ligne_d_affichage,9);
rubrique.date:=chaine;
entree_chaine(12,ligne_d_affichage,9);
rubrique.numero:=chaine;
if chaine='' then
                  begin
                       varcompte.index_osn:=varcompte.index_osn+1;
                       str(varcompte.index_osn,chaine);
                       gotoxy(12,ligne_d_affichage);
                       write('OSN '+chaine);
                       rubrique.numero:='OSN '+chaine;
                  end;
entree_chaine(22,ligne_d_affichage,14);
rubrique.objet:=chaine;
entree_nombre(37,ligne_d_affichage,10);
if variable_reelle=0 then
                        begin
                            gotoxy(37,ligne_d_affichage);
                            write('           ');
                            entree_nombre
                                       (48,ligne_d_affichage,10);
                            print_using(48,ligne_d_affichage,9);
                            rubrique.operation:=-variable_reelle;
                        end
                       else
                        begin
                             rubrique.operation:=variable_reelle;
                             print_using(37,ligne_d_affichage,9);
                        end;
rubrique.nouvel_avoir:=dernier_avoir+rubrique.operation;
variable_reelle:=rubrique.nouvel_avoir;
print_using(59,ligne_d_affichage,10);
entree_chaine(71,ligne_d_affichage,9);
rubrique.rentre_le:=chaine;
efface_ligne_1;
write('Ok (O/N) ?'+chr(2));
read(kbd,reponse);
if upcase(reponse)='O' then
          begin
           if rubrique.rentre_le<>'' then
                            varcompte.dif_rent:=varcompte.dif_rent
                            +rubrique.operation;
           dernier_avoir:=rubrique.nouvel_avoir;
           enregistrement(filesize(fichier));
           if ligne_d_affichage<23 then
                    ligne_d_affichage:=ligne_d_affichage+1
                           else
                               begin
                                ligne_d_affichage:=24;
                                tableau_rempli:=true;
                               end;
end
                else
      begin
         str(varcompte.index_osn,chaine);
         if rubrique.numero='OSN '+chaine then
```

```
                        varcompte.index_osn:=varcompte.index_osn-1;
                    efface_ligne_d_affichage;
                    if ligne_d_affichage=24 then
                                                tableau_rempli:=false;
                end;
    write(chr(3));
end;
```

REPERT1.INC

On trouvera dans ce fichier les procédures permettant l'affichage des possibilités offertes par la lecture de pages du fichier de gestion en cours.

affichage_No_page affiche le numéro de la page consultée dans le fichier.

demande_No_page est la procédure qui interroge l'utilisateur sur le numéro de la page qu'il désire consulter.

affichage_de_la_page permet l'affichage du numéro de page demandée ainsi que de la page concernée.

page_suivante permet, lorsque l'utilisateur a appuyé sur la touche +, de consulter la page suivante.

page_precedente permet, par l'appui sur la touche −, d'afficher la page précédent la page en cours de consultation.

```
{==========}
{repert1.inc}
{==========}

procedure affichage_No_page;

begin
     window(66,80,1,1);
     clrscr;
     lowvideo;
     write('Page No: ');
     write(page_a_lire);
     normvideo;
     window0;
end;

procedure demande_No_page;

begin
     write(chr(3));
     window(20,65,1,1);
     clrscr;
     write('No de page :');
     window(32,65,1,1);
     repeat
```

```
            repeat
                 clrscr;
                 {$I-}
                 read(page_a_lire);
                 {$I+}
            until IOresult=0;
      until (page_a_lire<=nombre_de_pages) and (page_a_lire>0);
end;

procedure affichage_de_la_page;

begin
      affichage_No_page;
      if (page_a_lire*20-19<=filesize(fichier)) and
                                    (filesize(fichier)<=page_a_lire*20)
                                        then fin:=filesize(fichier)
                                        else fin:=page_a_lire*20;
      for champ:=20*(page_a_lire-1) to fin-1 do
                                                affichage_champ(champ);
end;

procedure autre_page;

begin
      demande_No_page;
      affichage_de_la_page;
end;

procedure page_suivante;

begin
      if page_a_lire<nombre_de_pages then
                                        begin
                                            page_a_lire:=page_a_lire+1;
                                            affichage_de_la_page;
                                        end
                                     else variable_booleenne:=true;
end;

procedure page_precedente;

begin
      if page_a_lire<>1 then
                              begin
                                   page_a_lire:=page_a_lire-1;
                                   affichage_de_la_page;
                              end
                         else variable_booleenne:=true;
end;
```

REPERT2.INC Ce fichier contient la procédure lecture_d_une_page qui permet la lecture d'une page complète dans le fichier de gestion ouvert, afin de l'afficher dans la feuille de calcul.

```
{===========}
{repert2.inc}
{===========}

procedure lecture_d_une_page;

begin
     efface_ligne_1;
     nombre_de_pages:=1+trunc(filesize(fichier)/20);
     if nombre_de_pages=1+filesize(fichier)/20 then
                                   nombre_de_pages:=nombre_de_pages-1;
     lowvideo;
     write(nombre_de_pages);
     if nombre_de_pages>1 then write(' pages ecrites')
                          else write(' page ecrite');
     normvideo;
     autre_page;
     variable_booleenne:=false;
     repeat
        if variable_booleenne=false then
                                      begin
                                         window(20,65,1,1);
                                         write('Page: ');
                                         lowvideo;
                                         write('S');
                                         normvideo;
                                         write('uivante - ');
                                         lowvideo;
                                         write('P');
                                         normvideo;
                                         write('recedente - ');
                                         lowvideo;
                                         write('A');
                                         normvideo;
                                         write('utre - ');
                                         lowvideo;
                                         write('F');
                                         normvideo;
                                         write('in');
                                      end
                                   else variable_booleenne:=false;
        write(chr(2));
        repeat
             read(kbd,reponse);
        until upcase(reponse) In ['A','S','P','F'];
        case upcase(reponse) of
                        'A' : autre_page;
                        'S' : page_suivante;
                        'P' : page_precedente;
                        'F' : window0;
        end;
      until upcase(reponse)='F';
      write(chr(3));
end;
```

RECHERCH.INC

Ce fichier contient les procédures permettant de rechercher un enregistrement dans le fichier.

Lorsque cet enregistrement n'est pas trouvé dans la feuille de calcul courante, il est proposé une recherche dans le fichier complet, avec cette fois un accès au lecteur de disquette.

recherche (de,a) permet de chercher un enregistrement depuis le numéro de jusque le numéro **a**.

recherche_numero permet la recherche du numéro de l'enregistrement, éventuellement sur tout le fichier, si l'utilisateur accepte la requête.

```
{============}
{recherch.inc}
{============}

procedure recherche(de,a : integer);

begin
     numero_trouve:=false;
     repeat
           de:=de-1;
           lecture(de);
           if rubrique.numero=chaine then
              begin
               numero_trouve:=true;
                if tableau_rempli=true then
                   begin
                     tableau_rempli:=false;
                     affichage_champ(de);
                     tableau_rempli:=true;
                   end
                                        else affichage_champ(de);
              end;
           champ:=de
     until (de=a) or (numero_trouve=true);
end;

procedure recherche_numero;

begin
    efface_ligne_1;
    if (ligne_d_affichage=24) and (tableau_rempli=true) then
                                                    scrolling_haut;
    entree_chaine(12,ligne_d_affichage,9);
    if filesize(fichier)>60 then recherche(filesize(fichier),60)
                            else recherche(filesize(fichier),0);
    if numero_trouve=false then
                          begin
                            gotoxy(12,ligne_d_affichage);
                            write('          ');
                            gotoxy(1,1);
                            write(chaine);
                            gotoxy(10,1);
                            write(' :ne fait pas partie du fichier');
                            write(chr(2));
                            repeat
```

```
 until keypressed;
 if filesize(fichier)>60 then
   begin
      gotoxy(12,1);
      write
('Recherche sur tout le fichier (O/N)?');
      repeat
          read(kbd,reponse);
      until upcase(reponse) In ['O','N'];
      if upcase(reponse)='N' then
        begin
          tableau_rempli:=false;
          window(1,80,1,1);
          clrscr;
          window0;
                                    end
                                                  else
                                    begin
                                      recherche(60,0);
                                      if numero_trouve=false then
                                         begin
                                           window(12,80,1,1);
                                           clrscr;
                                write('ne fait pas partie du fichier');
                                           repeat
                                           until keypressed;
                                           tableau_rempli:=false;
                                           window(1,80,1,1);
                                           clrscr;
                                           window0;
                                         end;
                                    end;
                               end;
                          end;
     write(chr(3));
end;
```

MISEJOUR.INC

Trois procédures composent ce fichier, afin d'entrer la date de mise à jour, ainsi que les différents choix.

entree_date permet la saisie d'une date pour la mise à jour.

mise_jour permet de remplir la colonne Rentré-le pour valider l'apparition d'une opération sur un relevé bancaire.

mise_a_jour est la procédure contenant tous les choix permettant la mise à jour, depuis la recherche d'un numéro à mettre à jour, la consultation du fichier, l'entrée d'une date de valeur, jusqu'à la mise à jour effective.

```
{============}
{misejour.inc}
{============}

procedure entree_date;

begin
     entree_chaine(71,1,9);
     date:=chaine;
end;
       {remplie la colonne -rentre le- si numero trouve}

procedure mise_jour;

begin
    if numero_trouve=true then
       begin
         if ligne_d_affichage=24 then gotoxy(71,24)
                                 else gotoxy(71,ligne_d_affichage-1);
         write(date);
         if rubrique.rentre_le='' then
                               varcompte.dif_rent:=varcompte.dif_rent
                               +rubrique.operation;
         rubrique.rentre_le:=date;
         enregistrement(champ);
         numero_trouve:=false;
    end;
end;

procedure mise_a_jour;

begin
     numero_trouve:=false;
     repeat
         efface_ligne_1;
         window(1,80,1,1);
         gotoxy(7,1);
         lowvideo;
         write('D');
         normvideo;
         write('ate - ');
         lowvideo;
         write('N');
         normvideo;
         write('umero - ');
         lowvideo;
         write('L');
         normvideo;
         write('ecture du fichier - ');
         lowvideo;
         write('M');
         normvideo;
         write('ise a jour - ');
         lowvideo;
         write('Q');
         normvideo;
         write('uitter');
         gotoxy(71,1);
         write(date);
         window0;
         write(chr(2));
```

```
        repeat
             read(kbd,reponse);
        until upcase(reponse) In ['D','N','L','Q','M'];
        case upcase(reponse) of
                    'D' : entree_date;
                    'N' : if date<>'' then recherche_numero;
                    'L' : lecture_d_une_page;
                    'M' : mise_jour;
                    'Q' : write(chr(3));
        end;
     until upcase(reponse)='Q';
end;
```

SAUVFICH.INC

Ce fichier contient la procédure de sauvegarde du fichier de gestion de compte bancaire : sauvegarde_fin.

Après l'indication du dernier avoir devant figurer sur le fichier, il demande confirmation, puis les date et nom de l'utilisateur.

Une dernière confirmation est demandée, sinon il est possible de retourner au fichier.

```
{============}
{sauvfich.inc}
{============}

procedure sauvegarde_fin;

begin
     ouv_compte:=true;
     efface_ligne_1;
     gotoxy(2,1);
     write('Avoir indique sur le dernier extrait de compte :');
     write(varcompte.dif_rent:5:2);
     gotoxy(71,1);
     write('Ok (O/N)?');
     write(chr(2));
     read(kbd,reponse);
     if upcase(reponse)='O' then
       begin
            efface_ligne_1;
            write(' Mise a jour le :');
            entree_chaine(18,1,9);
            varcompte.date_de_mise_a_jour:=chaine;
            gotoxy(40,1);
            write('Par :');
            entree_chaine(45,1,15);
```

```
            varcompte.mise_a_jour_par:=chaine;
            close(fichier);
            enregistrement_variables;
            efface_ligne_1;
            write('  ');
            lowvideo;
            write('R');
            normvideo;
            write('etour au fichier');
            lowvideo;
            gotoxy(60,1);
            write('F');
            normvideo;
            write('in...(Sauvegarde)');
            write(chr(2));
            read(kbd,reponse);
            if upcase(reponse)<>'F' then
                begin
                 assign(fichier,nom_du_compte+nom_du_fichier+'.dat');
                 reset(fichier);
                 reponse:='Q';
                end
                                        else reponse:='S';
        end;
        ouv_compte:=false;
end;
```

SUITE.INC

Ce fichier est composé de trois procédures permettant de clôturer le fichier en cours, pour en ouvrir un nouveau, celui-ci reprenant les mêmes caractéristiques, et reportant le dernier avoir. Il est aussi possible de travailler sur un ancien fichier, si une mise à jour ultérieure est à effectuer.

Fermeture_ouverture est la procédure permettant de fermer le fichier en cours, pour en ouvrir un nouveau.

autre_fichier permet de s'intéresser à un fichier déjà clôturé.

suite est la procédure du menu permettant la sélection d'ouverture-fermeture, et le travail sur un fichier fermé.

```
{==========}
{suite.inc}
{==========}

procedure Fermeture_ouverture;

begin
     repeat
          efface_ligne_1;
          write('Date :');
          entree_chaine(7,1,9);
          date:=chaine;
          efface_ligne_1;
          write(' nom du nouveau fichier :');
          entree_chaine(26,1,4);
          gotoxy(31,1);
          if chaine=varcompte.fichier_courant then
                              write('-->est le fichier courant...');
          gotoxy(70,1);
          write(chr(2));
          write('Ok (O/N) ?');
          read(kbd,reponse);
     until upcase(reponse)='O';
     varcompte.fichier_courant:=chaine;
     rubrique.date:=date;
     rubrique.numero:='.........';
     rubrique.objet:='Fermeture--->';
     rubrique.operation:=0;
     rubrique.nouvel_avoir:=dernier_avoir;
     rubrique.rentre_le:='..'+varcompte.fichier_courant;
     enregistrement(filesize(fichier));
     close(fichier);
     enregistrement_variables;
     rubrique.objet:='Initialisation';
     rubrique.operation:=dernier_avoir;
     rubrique.rentre_le:=date;
     varcompte.index_osn:=0;
     assign(fichier,nom_du_compte+varcompte.fichier_courant+'.dat');
     rewrite(fichier);
     reset(fichier);
     enregistrement(filesize(fichier));
     nom_du_fichier:=varcompte.fichier_courant;
end;

procedure autre_fichier;

begin
     efface_ligne_1;
     close(fichier);
     write(' Nom du nouveau fichier :');
     entree_chaine(26,1,4);
     nom_du_fichier:=chaine;
     assign(fichier,nom_du_compte+nom_du_fichier+'.dat');
     {$I-}
     reset(fichier);
     {$I+}
     if ioresult<>0 then
          begin
               gotoxy(40,1);
               write(chr(2));
               write('--> N''existe pas...');
```

```
                nom_du_fichier:=varcompte.fichier_courant;
                assign(fichier,nom_du_compte+nom_du_fichier+'.dat');
                reset(fichier);
                repeat until keypressed;
           end;
end;


procedure suite;

begin
     ouv_compte:=true;
     efface_ligne_1;
     gotoxy(10,1);
     write
    ('Fermeture & Ouverture fichier - Travail sur un autre fichier');;
     lowvideo;
     gotoxy(10,1);
     write('F');
     if nom_du_fichier<>varcompte.fichier_courant then
                                write('ermeture & Ouverture fichier');
     gotoxy(42,1);
     write('T');
     normvideo;
     repeat
          read(kbd,reponse);
     until upcase(reponse) In ['F','T'];
     case upcase(reponse) of
           'F' : if nom_du_fichier=varcompte.fichier_courant then
                                                  Fermeture_ouverture;
           'T' : Autre_fichier;
     end;
     ouv_compte:=false;
end;
```

TRAVAIL.INC

Lors du choix de travail sur un fichier fermé, ce fichier est utilisé, grâce à la procédure **travail_sur_fichier** qu'il contient.

Il sera ainsi possible d'effectuer toutes les possibilités de mise à jour que possède le travail sur le fichier courant.

```
{==========}
{travail.inc}
{==========}

procedure travail_sur_fichier;
          {permet de travailler sur un fichier ferme}
begin
     repeat
        efface_ligne_1;
        write
        ('Lecture - Insertion - Recherche Numero ');
        write
        ('- Mise a jour - Sauvegarde(fin) - ');
        write('Suite');
        lowvideo;
        gotoxy(1,1);
        write('L');
        gotoxy(75,1);
        write('u');
        gotoxy(11,1);
        write('I');
        if nom_du_fichier<>varcompte.fichier_courant then
                                                write('nsertion');
        gotoxy(56,1);
        write('S');
        gotoxy(23,1);
        write('R');
        gotoxy(42,1);
        write('M');
        write(chr(2));
        normvideo;
        repeat
              read(kbd,reponse);
        until upcase(reponse) In ['L','I','R','S','M','U'];
        case upcase(reponse) of
             'L' : lecture_d_une_page;
             'I' : if nom_du_fichier=varcompte.fichier_courant then
                                                    insertion;
             'R' : recherche_numero;
             'M' : mise_a_jour;
             'U' : suite;
             'S' : sauvegarde_fin;
        end;
    until upcase(reponse)='S';
end;
```

CHOIX.INC

Les deux procédures de ce fichier permettent d'entrer les caractéristiques du compte à ouvrir, ou de celui à consulter.

autre_compte permet d'entrer les trois lettres du nom du compte lorsque ce n'est pas un compte du type CL (Compte sur Livret), CCP (Compte chèque) ou CE (Compte Epargne).

choix_du_compte permet la sélection du type de compte à utiliser. Si le choix est autre que ceux proposés, la procédure précédente est appelée.

```
{=========}
{choix.inc}
{=========}

procedure autre_compte;

begin
     efface_ligne_1;
     write(' Autre compte (3 lettrees) :');
     entree_chaine(29,1,3);
     nom_du_compte:=chaine;
end;

procedure choix_du_compte;

begin
     efface_ligne_1;
     write('  CCP  -  CL  -  CE  -  autre');
     if ouv_compte<>true then write(' - ouverture d''un Compte');
     lowvideo;
     gotoxy(5,1);
     write('P');
     gotoxy(12,1);
     write('L');
     gotoxy(18,1);
     write('E');
     gotoxy(24,1);
     write('A');
     gotoxy(32,1);
     if ouv_compte<>true then write('O');
     normvideo;
     write(chr(2));
     repeat
          read(kbd,reponse);
     until upcase(reponse) In ['P','L','E','A','O'];
     case upcase(reponse) of
          'P' : nom_du_compte:='CCP';
          'L' : nom_du_compte:='CL';
          'E' : nom_du_compte:='CE';
          'A' : autre_compte;
          'O' : ouv_compte:=true;
     end;
end;
```

OUVCCP.INC

Dans ce fichier se trouve la procédure d'ouverture de compte appelée ouverture_du_compte.

Dans cette procédure sont demandées toutes les caractéristiques du compte : Nom, Adresse, Ville, Code postal, Nom que vous donnez au premier fichier, Avoir initial.

Si vous confirmez ces entrées, la date et le nom de l'utilisateur vous sont demandés, pour initialiser le fichier, et entrer sur la feuille de calcul.

```
{==========}
{ouvccp.inc}
{==========}

procedure ouverture_du_compte;

begin
     efface_ligne_1;
     gotoxy(25,1);
     write('*** OUVERTURE D''UN COMPTE ***');
     repeat
        window(20,59,10,23);
        clrscr;
        gotoxy(25,2);
        write('Date :');
        gotoxy(10,4);
        write('Centre..... Numero c/c.....');
        gotoxy(5,6);
        write('Nom :');
        gotoxy(1,7);
        write('Adresse :');
        gotoxy(3,9);
        write('Ville :');
        gotoxy(2,11);
        write('Nom du 1er fichier :');
        gotoxy(2,12);
        write('avoir :');
        normvideo;
        entree_chaine(50,11,9);
        varcompte.date_de_mise_a_jour:=chaine;
        entree_chaine(29,14,12);
        int_compte.centre:=chaine;
        entree_chaine(43,14,12);
        int_compte.cc:=chaine;
        entree_chaine(30,15,28);
        int_compte.nom:=chaine;
        entree_chaine(30,16,28);
        int_compte.intitule1:=chaine;
        entree_chaine(30,17,28);
        int_compte.intitule2:=chaine;
        entree_chaine(30,18,28);
        int_compte.ville:=chaine;
        entree_chaine(42,20,4);
        varcompte.fichier_courant:=chaine;
        entree_nombre_sans_formatage(29,21,10);
        rubrique.operation:=variable_reelle;
        window0;
        lowvideo;
```

```
   gotoxy(25,23);
   write(' Ceci est-il correct : (O/N)? ');
   normvideo;
   write(chr(2));
   read(kbd,reponse);
until upcase(reponse)='O';
window(25,54,23,23);
clrscr;
variable_booleenne:=false;
choix_du_compte;
efface_ligne_1;
assign(intitule_compte,'int'+nom_du_compte+'.dat');
{$I-}
     reset(intitule_compte);
     if IOResult<>0 then enregistrement_de_l_intitule
                    else
                         begin
                            writeln(chr(7));
                            efface_ligne_1;
                            write('Ce fichier existe deja');
                            repeat until keypressed;
                            gotoxy(1,1);
                            write
                            ('Effacement,et initialisation: (O/N)?');
                            read(kbd,reponse);
                            if upcase(reponse)='O' then
                                       enregistrement_de_l_intitule;
                         end;
     varcompte.index_osn:=1;
     varcompte.mise_a_jour_par:='';
     varcompte.dif_rent:=rubrique.operation;
     enregistrement_variables;
     rubrique.date:=varcompte.date_de_mise_a_jour;
     rubrique.numero:='';
     rubrique.objet:='Initialisation';
     rubrique.nouvel_avoir:=rubrique.operation;
     rubrique.rentre_le:=rubrique.date;
     assign(fichier,nom_du_compte+varcompte.fichier_courant+'.dat');
     rewrite(fichier);
     reset(fichier);
     enregistrement(filesize(fichier));
     ouv_compte:=false;
end;
```

MENU.INC

Ce fichier contient les procédures initiales affichant le menu de départ, l'intitulé d'un compte existant, et la procédure de lancement de la gestion.

page_entete affiche la page de présentation de la gestion de compte.

tableau_intitule affiche les caractéristiques du compte (nom, numéro, ...)

menu_1 est la procédure de lancement permettant la lecture des caractéristiques du compte sur la disquette, et ouverture du fichier concerné.

```
{==========}
{menu.inc}
{==========}

procedure page_entete;

begin
     window(20,59,10,23);
     clrscr;
     gotoxy(5,2);
     write('** Gestion d''un compte cheque **');
     gotoxy(10,3);
end;

procedure tableau_intitule;

begin
     lecture_intitule;
     assign(compte_variable,'var'+nom_du_compte+'.dat');
     lire_fichier_des_variables;
     window(21,58,14,23);
     write('  Centre        Numero c/c');
     gotoxy(1,2);
     write(int_compte.centre);
     gotoxy(14,2);
     write(int_compte.cc);
     gotoxy(1,4);
     writeln(int_compte.nom);
     writeln(int_compte.intitule1);
     if int_compte.intitule2<>'' then writeln(int_compte.intitule2);
     writeln(int_compte.ville);
     writeln;
     write('Mis a jour le :');
     writeln(varcompte.date_de_mise_a_jour);
     write('            par :');
     writeln(varcompte.mise_a_jour_par);
     write('Fichier en cours :');
     write(varcompte.fichier_courant);
     repeat until keypressed;
end;

procedure menu_1;

begin
     border(0,0);
     ink(0,0,0);
     ink(1,0,0);
     clrscr;
     cadre;
     window(19,60,9,24);
     lowvideo;
     clrscr;
     normvideo;
     page_entete;
     ink(1,23,23);
     ouv_compte:=false;
     window0;
     choix_du_compte;
     efface_ligne_1;
     if ouv_compte=true then
                          begin
```

```
                                ouverture_du_compte;
                                ink(1,0,0);
                                window0;
                                for ligne_d_affichage:=9 to 24 do
                                                efface_ligne_d_affichage;
                                ink(1,23,23);
                            end
                        else
                            begin
                                repeat
                                    assign
                        (intitule_compte,'int'+nom_du_compte+'.dat');
                                    efface_ligne_1;
                                    window(1,23,1,1);
                                    {$I-}
                                    reset(intitule_compte);
                                    {$I+}
                                    if ioresult<>0 then
                                            begin
                                              efface_ligne_1;
                                              write(chr(2));
          write(nom_du_compte,' n''est pas sur cette disquette...');
                                              repeat
                                              until keypressed;
                                              ouv_compte:=true;
                                              choix_du_compte;
                                              variable_booleenne:=false;
                                            end
                                                  else
                                            begin
                                              efface_ligne_1;
                                              tableau_intitule;
                                              variable_booleenne:=true;
                                            end;
                                 until variable_booleenne=true;
                                 assign
                                 (fichier,nom_du_compte
                                 +varcompte.fichier_courant+'.dat');
                                 reset(fichier);
                                 ink(1,0,0);
                                 window(19,60,9,24);
                                clrscr;
                                window0;
                                lignes_centrales;
                                ink(1,23,23);
                            end;
      date:=varcompte.date_de_mise_a_jour;
      lecture(filesize(fichier)-1);
      ouv_compte:=false;
      dernier_avoir:=rubrique.nouvel_avoir;
      nom_du_fichier:=varcompte.fichier_courant;
      ligne_d_affichage:=5;
      tableau_rempli:=false;
      travail_sur_fichier;
      window0;
      write(chr(3));
end;
```

CRÉATION DU PROGRAMME EXECUTABLE

Une fois tous ces fichiers entrés et sauvegardés sur disquette à l'aide de l'éditeur fourni dans Turbo Pascal ou d'un traitement de texte, nous vous proposons la procédure de compilation suivante :

— Rappelez-vous de bien avoir répondu N à la question

Include Error Messages (Y/N) ?

sinon relancez Turbo Pascal

— Initialisez Main file à l'aide d'une chaîne vide (pas de fichier principal).

— Initialisez Work file avec le nom COMPTE.PAS.

— Choisissez Options dans le menu principal de TURBO.

— Dans le sous menu présenté, choisissez l'option Comfile, en effet le programme résultant de la compilation sera trop grand pour résider en mémoire avec Turbo Pascal, il sera donc compilé sur disquette.

— Revenez au menu principal en frappant sur Quit.

— Lancez alors la compilation par Compile.

Après quelques petites minutes, le fichier COMPTE.COM est créé, et Turbo Pascal vous affiche :

```
Loading A:COMPTE.PAS
Compiling — — > A:COMPTE.COM
1376 lines
Code: 10721 byte (20E2-4AC3)
Free: 19363 Byte (4AC4-9667)
Data: 1498 byte (9668-9C42)
```

Effectuez immédiatement une copie du programme COMPTE.COM que vous pourrez utiliser, et conservez la précédente disquette avec ses fichiers Pascal.

UNE UTILISATION PRATIQUE DU LOGICIEL

Sous CP/M 2.2, vous pouvez utiliser l'utilitaire SETUP pour créer une disquette auto-start, lors du lancement de CP/M, selon la procédure suivante :

— Effectuez une copie de SETUP.COM sur la disquette contenant COMPTE.COM.

— Lancez SETUP.

— A la première question :

```
**Initial Command Buffer Empty
Is this correct (Y/N) ?
```

répondez No et frappez **COMPTE↑M** et <RETURN> au message :

Enter Initial Command Buffer

— Répondez ensuite oui à toutes les autres questions (15 au total) jusqu'à relancement de CP/M qui doit démarrer automatiquement sur le logiciel de gestion.

— Effectuez une copie du logiciel **PIP.COM** ou **FILECOPY.COM** livrés avec CP/M 2.2 sur la disquette de travail.

Après chaque utilisation du programme **COMPTE.COM**, nous vous conseillons, lors du retour sous CP/M d'effectuer une copie de tous les fichiers créés ou modifiés par la commande :

PIP B: = A:*.DAT