Solution Problème n°1

```
program systeme_equations;

const M=5;

var a: array[1..M, 1..M] of real;
    b: array[1..M] of real;
    x: array[1..M] of real;
    n: integer;
    i,j: integer;
    s: real;

begin

   repeat
      write('Entrez le nombre d''equations du systeme: ');
      readln(n);
   until (n>=2) and (n<=M); (* 1 point pour cela *)
   writeln('Entrez les coefficients du systeme:');
   for i:=1 to n do
      for j:=i to n do
         begin
            write('a',i:2,j:2,' = ');
            readln(a[i,j]);
         end;
   writeln('Entrez les termes du second membre');
   for i:=1 to n do
      begin
         write('b',i,'=');
         readln(b[i]);
      end;
            (* 1 point pour cela *)
   for i:=n downto 1 do
      begin
         s:=b[i];
         for j:=n downto i+1 do
            s:=s-a[i,j]*x[j];
         x[i]:=s/a[i,i];
      end;
(* 2 points pour cela *)
   writeln('La solution du systeme est:');
   for i:=1 to n do writeln('x',i,'=',x[i]:4:2);
   readln;

end.
```

Solution Problème n°2

```
program Test;
var mot : string;
function estNombre (c:char) :boolean;
begin
 if ord(c) in [ord('0')..ord('9')] then
    estNombre := true
  else
    estNombre := false;
end;

procedure num2str (var s: string);
var  i: integer;
  begin
  for i:=1 to length(s) do
     if estNombre(s[i]) then s[i]:=chr(ord(s[i])-ord('0')+ord('A'))
end;

begin
  writeln('Un mot');
  read(mot);
  num2str(mot);
  write(mot);
  readln;
  readln;
end.
```

Solution Problème n°3

1)

```
CONST
      Max = 300;

 TYPE

      Eleve = Record
        Nom, Prénom : string ;
        Date : string ;
        Median, Final, TP : real ;
      End ;


   VAR
      F : File of Eleve ;
      Tab: Array [1..MAX] of Eleve;
```

2) Pas de difficulté majeure (c'est comme en TD) : Utiliser REWRITE(f), RESET(f,x), Read(f,x), Write(f,x), While not EOF(f) … sans oublier CLOSE(f) à la fin.

a) Créer le fichier.

```
Procedure CREATION ;
Var   i, N: integer;
      Etu : Eleve ;
      UV : string ;
Begin
   Write('Nom de l''UV ?') ;
   Readln(UV) ;
   Assign(F,UV) ;
   Rewrite(F) ;
   Write('nombre d''étudiants à saisir ?') ; Readln(N) ;
   For i := 1 to N do
     Begin
          Write('Nom : ') ; Readln(etu.nom) ;
          Write('Prénom : ') ; Readln(etu.prenom) ;
          Write('Date de Naissance : ') ; Readln(etu. Date);
          Write('Médian : ') ; Readln(etu.median) ;
          Write('Final : ') ; Readln(etu.final) ;
          Write('TP : ') ; Readln(etu.TP) ;
          Write(F,etu);
     End;
   Close(F);
   End;
```

b) Ajouter un étudiant au fichier de cette UV.

Il faut ajouter les élèves **en fin de fichier**. Donc deux possibilités : soit aller directement en fin de fichier en ouvrant le fichier avec l'instruction APPEND(F) au lieu de RESET(F), soit ouvrir avec RESET(F) et parcourir tout le fichier jusqu'à la fin :

```
Procedure AJOUT ;
Var etu : eleve;
Begin
   Write('Nom de l''UV ?') ;
```

```
   Readln(UV) ;
   Assign(F,UV) ;
   Reset(F);
   While not EOF(F) do
     Read(F,etu);
   Write('nombre d''étudiants à saisir ?') ; Readln(N) ;
   For i := 1 to N do
      Begin
           Write('Nom : ') ; Readln(etu.nom) ;
           Write('Prénom : ') ; Readln(etu.prenom) ;
           Write('Date de Naissance : ') ; Readln(etu. Date);
           Write('Médian : ') ; Readln(etu.median) ;
           Write('Final : ') ; Readln(etu.final) ;
           Write('TP : ') ; Readln(etu.TP) ;
           Write(F,etu);
      End;
   Close(F);
  End;
```

c) Afficher la liste des étudiants n'ayant pas la moyenne dans cette UV.

```
Procedure MOYENNE ;
Var   moyenne: integer;
      Etu : Eleve ;
      UV : string ;
Begin
   Write('Nom de l''UV ?') ;
   Readln(UV) ;
   Assign(F,UV) ;
   Reset(F) ;
   While not EOF(F) do
      Begin
           Read(F,etu);
           Moyenne:=(etu.median+etu.final+etu.TP)/3;
           If Moyenne < 10
             Then
                Begin
```

```
                    Write(etu.nom) ;
                    Write(etu.prenom) ;
                    Write(etu. Date);
                    Write(etu.median) ;
                    Write(etu.final) ;
                    Write(etu.TP) ;
                    Write(moyenne);
                End;
      End;
   Close(F);
   End;
```

d) Rechercher un étudiant de cette UV à partir de son nom de famille et de son prénom. On supposera que deux étudiants peuvent avoir le même nom, mais qu'ils ne peuvent pas avoir à la fois le même nom ET le même prénom.

```
Procedure RECHERCHE ;
Var   Etu: Eleve ;
      UV, nom, prenom : string ;
Begin
   Write('Nom de l''UV ?') ;
   Readln(UV) ;
   Write('Nom et Prénom de l''étudiant recherché ?') ;
   Readln(nom) ;
   Readln(prenom) ;
   Assign(F,UV) ;
   Reset(F) ;
   Repeat
          Read(F,etu);
          If (etu.nom = nom) AND (etu.prenom=prenom)
            Then
                Begin
                    Write(etu.nom) ;
                    Write(etu.prenom) ;
                    Write(etu.Date);
                    Write(etu.median) ;
                    Write(etu.final) ;
```

```
                    Write(etu.TP) ;
                    Write('Moyenne : ',(etu.final+etu.median+etu.tp)/3);
                End;
    Until EOF(F) OR ((etu.nom = nom) AND (etu.prenom=prenom))
  Close(F);
  End;
```

e) Supprimer un étudiant du fichier de cette UV. Le plus simple est d'utiliser un tableau.

```
Procedure SUPPRIME ;
Var   i,j: integer;
      UV, nom, prenom : string ;
Begin
   Write('Nom de l''UV ?') ;
   Readln(UV) ;
   Write('Nom et Prénom de l''étudiant à supprimer ?') ;
   Readln(nom) ;
   Readln(prenom) ;
   Assign(F,UV) ;
   I:=0;
   Reset(F) ;
   While not EOF(F) do
      Begin
          i:=i+1;
          Read(F,Tab[i]);
          If (etu.nom = nom) AND (etu.prenom=prenom)
            Then
              i:=i-1;
    End;
   Close(F);
   Assign(F,UV) ;
   Rewrite(F) ;
   For j:=1 to i do
     Write(F,tab[j]);
   Close(F);
   End;
```

Solution Problème n°4

- On utilise une fonction récursive “RechercheSommet()”, qui prend comme entrée un tableau, le début et la position de recherche et la fin de la position de recherche.

6	*8*	*9*	*12*	*16*	*17*	19	*11*	*9*	*7*	*4*	*3*

- On divise le tableau d'entrée en 2 parties : soit ***mid*** l'entier qui indique le milieu du tableau.

6	*8*	*9*	*12*	*16*	*17*	19	*11*	*9*	*7*	*4*	*3*

- Le sommet se trouve dans l'une des deux parties, soit la partie “gauche” ou le partie “droite”, pour décider ou est ce qu'on va continuer la recherche, on compare deux éléments des deux parties, on continue la recherche dans la partie qui contient l'élément le plus grand entre les deux éléments comparés.

Pour l'exemple on continu la recherche dans la partie droite du tableau.

- On relance la recherche par la fonction RechercheSommet() dans la nouvelle partie.

6	*8*	*9*	*12*	*16*	*17*	19	*11*	*9*	*7*	*4*	*3*

6	*8*	*9*	*12*	*16*	*17*	19	*11*	*9*	*7*	*4*	*3*

Algorithme

```
RechercheSommet(Tab[],  Entier Debut, Entier Fin)
Debut
        Soit un Entier mid initialisé à 0

        Si (Debut = Fin) Alors
                RechercheSommet ← Tab[Debut]
        Sinon
                Debut
                        mid = (Debut + Fin) / 2

                        Si (Tab[mid] > Tab[mid+1]) //Pour choisir la prochaine partie
                                RechercheSommet ← RechercheSommet(Tab,Debut,mid)//gauche
                        Sinon
                                RechercheSommet ← RechercheSommet(Tab,mid+1, Fin)//droit
                Fin
Fin
```