LE GOUVERNEMENT
DU GRAND-DUCHÉ DE LUXEMBOURG
Ministère de l'Éducation nationale,
de l'Enfance et de la Jeunesse

EXAMEN DE FIN D'ÉTUDES SECONDAIRES
2017

BRANCHE	SECTION(S)	ÉPREUVE ÉCRITE
Informatique - Partie théorique	**B**	*Durée de l'épreuve* 50 min
		Date de l'épreuve
		Numéro du candidat

1. Calcul du PGCD (3+3 = 6 points)

a) Donner l'algorithme d'Euclide qui calcule le PGCD de deux nombres entiers par la division.

b) Décrire brièvement son fonctionnement à l'aide du calcul du PGCD de 24 et 36.

Solution

a) Algorithme

```
function euclideDivi(a,b:integer):integer;
var c:integer;
begin
  while b>0 do begin
    c:=a mod b;
    a:=b;
    b:=c
  end;
  euclideDivi:=a
end;
```

b) euclideDivi(24,36)

Itération	**c**	**a**	**b**
/	/	24	36
1	24	36	24
2	12	24	12
3	0	***12***	0

Donc: euclideDivi(24,36) = 12

2. Algorithme de tri (6+3 = 9 points)

a) Donner une version récursive de l'algorithme de tri par sélection.

b) Noter les modifications subies par la liste contenant les éléments P;Y;T;H;O;N lors de l'application de cet algorithme

Solution

a) Algorithme (Tri par Sélection – version récursive)

```
1  procedure triSelectionR(var liste:TListBox; debut:integer);
2  var j,min:integer;
3  begin
4    min:=debut;
5    for j:=debut+1 to liste.Items.Count-1 do
6      if liste.Items[j]<liste.Items[min] then
7        min:=j;
8    echange(liste,debut,min);
9    if debut<liste.Items.Count-2 then
10     triSelectionR(liste,debut+1)
11 end;
```

Appel de la procédure : `triSelectionR(liste, 0);`

b) Exemple d'exécution

P	Y	T	H	O	N
H	Y	T	P	O	N
H	N	T	P	O	Y
H	N	O	P	T	Y
H	N	O	P	T	Y
H	N	O	P	T	Y
H	N	O	P	T	Y

3. Algorithme de Recherche (7+2 = 9 points)

a) Donner une version itérative de l'algorithme de recherche dichotomique.

b) Décrire brièvement le fonctionnement de cette fonction.

Solution

a) Recherche dichotomique – version itérative

```
1  function rechDichoI(liste:TListBox; cle:string):integer;
2  var milieu,g,d:integer;
3  begin
4    g:=0;
5    d:=liste.Items.Count-1;
6    milieu:=(g+d) div 2;
7    while (cle<>liste.Items[milieu]) and (g<=d) do begin
8      if cle<liste.Items[milieu] then
9        d:=milieu-1
10     else
11       g:=milieu+1;
12     milieu:=(g+d) div 2
13   end;
14   if cle=liste.Items[milieu] then
15     rechDichoI:=milieu
16   else
17     rechDichoI:=-1
18 end;
```

b) Idée

On divise la liste en deux parties. On regarde si la clé correspond à l'élément au milieu de la liste. Si c'est le cas on a terminé, sinon on détermine la partie qui contient la clé et on recommence la recherche dans la partie contenant la clé. La liste ne doit pas forcément contenir la clé, mais **elle doit être triée.**

4. Programme inconnu - Correction de fautes (6 points)

Déterminez et corrigez les erreurs logiques et syntaxiques de la partie de programme indiquée ci-dessous.

L'extrait de programme doit chercher et transférer le minimum d'une série de notes (entiers compris entre 1 et 60 inclus) contenues dans le composant `lbListe` de type `TlistBox` dans le libellé `lblResultat`.

```
procedure TForm1.btnMinimumClick(Sender: TObject);
var min, i:integer;
begin
    min:=0;
    for i:=0 to lbListe.Items.Count do
        if lbListe.Items[i] > min then
              min:=StrToInt(lbListe.Items[min]);
    lblResultat:='Note minimale: '+StrToInt(min);
end;
```

Solution

```
procedure TForm1.btnMinimumClick(Sender: TObject);
var min, i:integer;
begin
    min:=60;  // ou premier élément de la liste
    for i:=0 to lbListe.Items.Count-1 do
        if StrToInt(lbListe.Items[i]) < min then
              min:=StrToInt(lbListe.Items[i]);
    lblResultat.Caption:='Note minimale: '+IntToStr(min);
end;
```