

CORRECTIONS DES EXERCICES DU CONCOURS D'ENTREE EPITA 2005

REMARQUE :

Les programmes en Pascal, C sont des exemples et peuvent être discutés en terme d'implémentation et de construction. Le choix qui a été fait, est celui d'une découpe fonctionnelle et procédurale importante.

Cela permet de mieux faire ressortir un algorithme principal.

Ensuite chaque tâche est détaillée dans la procédure ou fonction correspondante.

D'autre part, il n'y a pas forcément d'optimisation entre les programmes Pascal et C qui peuvent être de simple traduction les uns des autres.

A) SI ROME...

Le principe est le suivant :

Pour effectuer le calcul, nous allons d'abord traduire les deux nombres romains à additionner en deux nombres décimaux que nous allons additionner. La somme étant, dans la mesure du possible (si elle est inférieure ou égale à 3999), convertie en nombre romain.

Les programmes qui suivent présentent deux solutions différentes basées sur la table de conversion suivante, où les règles de conversion sont appliqués de haut en bas.

1000 M
900 CM
500 D
400 CD
100 C
90 XC
50 L
40 XL
10 X
9 IX
5 V
4 IV
1 I

Le programme en Pascal définit plusieurs tables (unités, dizaines, centaines et milliers) alors que le programme en C définit juste une chaîne de caractères. Ensuite, chaque nombre romain est « démonté » morceau par morceau pour être converti. La correspondance étant obtenue soit par les tables, soit par la chaîne.

Programme Pascal (Delphi mode console)

```
program Rome;  
  
{ $APPTYPE CONSOLE }  
  
uses  
    SysUtils;  
  
const  
    (* Unités de 0 à 90 *)  
    unites : array[0..9] of string = ( '', 'I', 'II', 'III', 'IV', 'V', 'VI', 'VII',  
    'VIII', 'IX' );
```

```

(* Dizaines de 0 à 90 *)
dizaines : array[0..9] of string = ( '', 'X', 'XX', 'XXX', 'XL', 'L', 'LX',
'LXX', 'LXXX', 'XC' );

(* Centaines de 0 à 900 *)
centaines : array[0..9] of string = ( '', 'C', 'CC', 'CCC', 'CD', 'D', 'DC',
'DCC', 'DCCC', 'CM' );

(* Milliers de 0 à 3000 *)
milliers : array[0..3] of string = ( '', 'M', 'MM', 'MMM' );

Var
  FichierEntree, FichierSortie : Text;

procedure OuvertureFichier;
begin
  Assign(FichierEntree, 'ROME.IN');
  Reset(FichierEntree);
  (* Ouverture de ROME.IN en
lecture *)
  Assign(FichierSortie, 'ROME.OUT');
  Rewrite(FichierSortie);
  (* Ouverture de ROME.OUT en
écriture *)
end;

function romain2arabe(nombre:string):integer;
var
  valeur,i : integer;
begin
  (* calcul des valeurs de prem *)
  valeur:=0;
  (* milliers *)
  i:=3;
  while (i>0) and (pos(milliers[i],nombre)<>1) do i:=i-1;
  if i>0 then begin
    valeur:=valeur+i*1000;
    nombre:=copy(nombre,length(milliers[i])+1,length(nombre));
  end;
  (* centaines *)
  i:=9;
  while (i>0) and (pos(centaines[i],nombre)<>1) do i:=i-1;
  if i>0 then begin
    valeur:=valeur+i*100;
    nombre:=copy(nombre,length(centaines[i])+1,length(nombre));
  end;
  (* dizaines *)
  i:=9;
  while (i>0) and (pos(dizaines[i],nombre)<>1) do i:=i-1;
  if i>0 then begin
    valeur:=valeur+i*10;
    nombre:=copy(nombre,length(dizaines[i])+1,length(nombre));
  end;
  (* unités *)
  i:=9;
  while (i>0) and (pos(unites[i],nombre)<>1) do i:=i-1;
  if i>0 then begin
    valeur:=valeur+i;
    nombre:=copy(nombre,1,length(unites[i]));
  end;

  romain2arabe:=valeur;
end;

Procedure LectureEtTraitement;
Var
  ds, premier, second, somme : integer;
  ligne, prem, sec, res : string;

Begin
  readln(FichierEntree,ds);
  while (ds > 0) do
  Begin

```

```

readln(FichierEntree, ligne);
prem:=copy(ligne,1,pos('+',ligne)-1);
sec:=copy(ligne,length(prem)+2,length(ligne)-length(prem)-2);
write(FichierSortie,prem,'+',sec,'='); (* séparation des deux nombres *)

(* calcul des valeurs de prem *)
premier:=romain2arabe(prem);
(* calcul des valeurs de sec *)
second:=romain2arabe(sec);

somme:=premier+second;

if somme>3999
  then res:='CONCORDIA CUM VERITATE'
  else res:=Concat(milliers[somme DIV 1000],centaines[(somme MOD 1000) DIV
100],dizaines[(somme MOD 100) DIV 10],unites[somme MOD 10]);
  ds:=ds-1;
  writeln(FichierSortie,res);
End;
End;

procedure FermetureFichier;
begin
  Close(FichierEntree);
  Close(FichierSortie);
end;

begin
  OuvertureFichier;
  LectureEtTraitement;
  FermetureFichier;
end.

```

Programme C (DevC++ 4)

```

#include <string.h>
#include <stdio.h>

FILE          *FichierEntree; /* Fichier TEXT pour l'entrée */
FILE          *FichierSortie; /* Fichier TEXT pour la sortie */
int cval[256];

void OuvertureFichier()
{
  if ((FichierEntree = fopen("ROME.IN", "r")) == NULL)
  {
    perror("ROME.IN");
    exit(1);
  }
  rewind(FichierEntree);
  /* Ouverture de ROME.IN en lecture */
  FichierSortie = fopen("ROME.OUT", "w");
  if (FichierSortie != NULL)
    rewind(FichierSortie);
  else
    FichierSortie = tmpfile();
  if (FichierSortie == NULL)
  {
    perror("FichierSortie");
    exit(1);
  }
}

```

```

/* Ouverture de ROME.OUT en écriture */
}

void FermetureFichier()
{
    if (FichierEntree != NULL)
        fclose(FichierEntree);
    FichierEntree = NULL;
    if (FichierSortie != NULL)
        fclose(FichierSortie);
    FichierSortie = NULL;
}

int Romain2Arabe(char* romain)
{
    int valeur = 0;
    while (*romain)
    {
        if (cval[*romain] < cval[*romain+1])
            { valeur += cval[*romain+1]-cval[*romain]; romain+=2; }
        else valeur += cval[*romain++];
    }
    return valeur;
}

void Arabe2Romain(int num, char* dig)
{
    if (!num) return;
    Arabe2Romain(num/10, dig+2);
    switch(num %= 10)
    {
        case 0: break;

        case 3: fprintf(FichierSortie,"%c",*dig);
        case 2: fprintf(FichierSortie,"%c",*dig);
        case 1: fprintf(FichierSortie,"%c",*dig); break;

        case 4: fprintf(FichierSortie,"%c%c",*dig,dig[1]); break;
        case 9: fprintf(FichierSortie,"%c%c",*dig,dig[2]); break;

        default: fprintf(FichierSortie,"%c",dig[1]);
            while (num>5) { fprintf(FichierSortie,"%c%",*dig); num--; }
    }
}

void LectureEtTraitement()
{
    int i, n, ds, premier, second;
    char ipt[1000];
    char digits[]="IVXLCDM";
    char* c;

```

```

for (i=0; i<256; i++) cval[i] = 0;
for (n=1, i=0; i<7; i++)
  { cval[digits[i]] = n;
    n*=((i%2)?2:5);
  }

fscanf(FichierEntree,"%d", &ds);
while (ds--)
  {
  fscanf(FichierEntree,"%s", ipt);
  fprintf(FichierSortie,"%s", ipt);
  premier = Romain2Arabe(strtok(ipt, " +"));
  second = Romain2Arabe(strtok(NULL, " +="));
  if (premier+second > 3999) fprintf(FichierSortie, "%s", "CONCORDIA CUM VERITATE\n\n");
  else { Arabe2Romain(premier+second, digits);
        fprintf(FichierSortie,"%s","\n\n");
      }
  }
}

```

```

int main(int argc, char *argv[])
{
  OuvertureFichier();
  LectureEtTraitement();
  FermetureFichier();

  return 0;
}

```

B) DISTANCE MAXIMUM

Le principe est le suivant :

Une façon simple de trouver la solution est de calculer pour chaque paire $(X[i], Y[j])$ sa distance et de conserver la plus grande calculée. Ce programme prendra n^2 comparaisons.

Une solution un peu plus efficace est obtenue en ne comparant que les paires $(X[i], Y[j])$, lorsque $i < j$. Le nombre de comparaisons pas alors à $n(n-1)/2$.

Si nous nous servons du fait que les séquences sont décroissantes, nous pouvons obtenir un programme encore plus efficace qui prendra environ n comparaisons.

Un principe d'un tel programme pourrait être le suivant (il y en a d'autres):

Nous utiliserons 4 variables : m , i , j , and n . Les variables i et j sont les indices des tableaux X et Y , n correspond à la longueur de ces deux tableaux et m est la distance maximum calculée.

Sa valeur est : $m = \max \{ d(X[k], Y[l]) \text{ avec } k < i \text{ ou } l < j \}$ avec $m=0$ pour $i=j=0$. Notons que si $i=n$ ou $j=n$, m représente alors le maximum recherché.

Si $X[i] \leq Y[j]$ alors pour tout $k \geq i$ $X[k] \leq Y[j]$. Ainsi toutes les paires $(X[k], Y[j])$ auront une distance au plus $d(X[i], Y[j])$. De là, nous pouvons déduire que $\max \{ m, d(X[i], Y[j]) \} = \max \{ d(X[k], Y[l]) \text{ avec } k < i \text{ ou } l < j+1 \}$

Si $X[i] > Y[j]$, alors pour tout $k \geq j$ $X[i] > Y[k]$, et toutes les paires $(X[i], Y[k])$ auront au plus une distance 0. D'où $m = \max \{ d(X[k], Y[l]) \text{ avec } k < i+1 \text{ ou } l < j \}$.

Cela donne le programme suivant où nous avons initialement $i=0$, $j=0$, et $m=0$. Si $i=n$ ou $j=n$, nous avons fini et m représenté le maximum que nous recherchons. A chaque tour, nous effectuons une comparaison et soit i soit j est incrémenté. Le nombre maximum de comparaisons sera alors au plus de $2n-1$.

Programme Pascal (Delphi mode console)

```

program distancemaximum;

{$APPTYPE CONSOLE}

uses
  SysUtils;

Var
  FichierEntree, FichierSortie : Text;

procedure OuvertureFichier;
begin
  Assign(FichierEntree, 'DISTANCE.IN');
  Reset(FichierEntree);
en lecture *)
  Assign(FichierSortie, 'DISTANCE.OUT');
  Rewrite(FichierSortie);
en écriture *)
end;

Procedure LectureEtTraitement;
Var
  t, i, j, n, m : integer;
  X, Y : array [0 .. 1000] of integer;

Begin
  readln(FichierEntree, t);
  while (t > 0) do
  Begin
    readln(FichierEntree, n);
    for i := 0 to (n - 1) do read(FichierEntree, X[i]);
    for i := 0 to (n - 1) do read(FichierEntree, Y[i]);
    i := 0;
    j := 0;
    m := 0;
    while ((i < n) and (j < n)) do
    Begin
      if (Y[j] >= X[i]) then
      Begin
        if (j - i > m) then m := j - i;
        j := j + 1;
      End
      else i := i + 1;
    End;
    writeln(FichierSortie, 'La distance maximum est ', m);
    t := t - 1;
    writeln(FichierSortie);
  End;
End;

procedure FermetureFichier;
begin
  Close(FichierEntree);
  Close(FichierSortie);
end;

begin
  OuvertureFichier;
  LectureEtTraitement;
  FermetureFichier;

```

end.

Programme C (DevC++ 4)

```
#include <string.h>
#include <stdio.h>

FILE      *FichierEntree;  /* Fichier TEXT pour l'entrée */
FILE      *FichierSortie; /* Fichier TEXT pour la sortie */

void OuvertureFichier()
{
    if ((FichierEntree = fopen("DISTANCE.IN", "r")) == NULL)
    {
        perror("DISTANCE.IN");
        exit(1);
    }
    rewind(FichierEntree);
    /* Ouverture de DISTANCE.IN en lecture */
    FichierSortie = fopen("DISTANCE.OUT", "w");
    if (FichierSortie != NULL)
        rewind(FichierSortie);
    else
        FichierSortie = tmpfile();
    if (FichierSortie == NULL)
    {
        perror("FichierSortie");
        exit(1);
    }
    /* Ouverture de DISTANCE.OUT en écriture */
}

void FermetureFichier()
{
    if (FichierEntree != NULL) fclose(FichierEntree);
    FichierEntree = NULL;
    if (FichierSortie != NULL) fclose(FichierSortie);
    FichierSortie = NULL;
}

void LectureEtTraitement()
{
    int  t,i,j,n,m,X[1000],Y[1000]; /* arrays for sequence X and Y */

    fscanf(FichierEntree, "%d", &t);
    while (t>0)
    {
        fscanf(FichierEntree, "%d",&n);
        for (i=0; i<n; i++) fscanf(FichierEntree, "%d",&X[i]);
        for (i=0; i<n; i++) fscanf(FichierEntree, "%d",&Y[i]);

        i=0; j=0; m=0;
        while (i<n && j<n)
        {
            if (Y[j] >= X[i])
            {
                if ( (j-i)>m ) m=j-i;
                j=j+1;
            }
            else i=i+1;
        }
        fprintf(FichierSortie, "The maximum distance is %d\n",m);

        t--;
        fprintf(FichierSortie, "\n");
    }
}
```

Remarque : A une question correspond au moins une réponse juste.
Cochez la ou les bonnes réponses.

Barème : Une bonne réponse = +1
Pas de réponse = 0
Une mauvaise réponse = -1

Question-1 :
Do while $I < 10$
{
A = A + 1
Print A
}

La suite d'instructions précédente correspond à:

- A- une alternative
- B- une boucle*****
- C- l'impression des dix premiers chiffres
- D- l'impression de la somme des 9 premiers chiffres *****

Question-2 : If $X > 10$ then $X = X*2$ else $X = X + 1$

La suite d'instructions précédente correspond à:

- A- une alternative*****
- B- une boucle
- C- l'impression des dix premiers chiffres
- D- l'impression de la somme des 9 premiers chiffres

Question-3 : Q'imprime ce programme:

```
I = 1
Étiquette1
    Si  $I < 10$  alors  $I = I * 1$ 
        Imprimer I
         $I = I + 1$ 
    Aller à étiquette1
```

- A- le logarithme de I
- B- la puissance 10 de I
- C- les 9 premiers chiffres*****
- D- les 10 premiers chiffres

Question-4 : Parmi les instructions suivantes, laquelle est incorrecte ?
(B est un booléen, E et R sont des entiers)

- A- $R = E * 2$
- B- $E = R + 10$

- C- Si B est vrai alors $R = 0$
- D- Sinon $B = B + 10$ *****

Question-5 : Lequel n'est pas un système d'exploitation ?

- A- windows
- B- unix
- C- winsock*****
- D- linux

Question-6 : Un octet est codé sur:

- A- 8bits*****
- B- 16 bits
- C- 32 bits
- D- 64 bits

Question-7 : Sur combien d'octets on peut coder la valeur 256

- A- un
- B- deux*****
- C- trois
- E- quatre

Question-8 : En quel langage est codée l'instruction suivante ?

- A- java
- B- C
- C- Camel*****
- D- c#

Question-9 : la fonction principale d'un modem est :

- A- transformer le codage ASCII en EBCDIC
- B- protéger l'ordinateur d'une chute de tension
- C- adapter le signal de la ligne à celui de l'ordinateur*****
- D- adapter le signal de l'ordinateur à celui de la ligne*****

Question-10 : Quelle est la valeur de A

```
Pour I = 1 à 5
    A = 2
    A = A + I
Fin
Afficher A
```

- A- 5
- B- 6
- C- 7*****
- D- 8

Question-11 : ADSL est une technologie de :

- A- supervision de réseau local
- B- transport de la voix sur la boucle locale*****
- C- transport de données sur la boucle locale*****
- D- contrôle d'accès aux base de données

Question-12 : l' adresse suivante 192.12.42.56 correspond à une adresse

- A- téléphonique
- B- internet*****
- C- numeris
- D- transpac

Question-13 : Quel est l'un des avantages des câbles à fibre optique dans les réseaux ?

- A- leur prix est abordable
- B- ils sont faciles à installer
- C- ils sont insensibles aux interférences électromagnétiques*****
- D- ils peuvent avoir une longueur importante*****

Question-14: Quelle est la fonction de mise à la terre de sécurité dans un ordinateur ?

- A- elle relie le fil sous tension au châssis
- B- elle empêche les parties métalliques de transmettre une tension dangereuse dans le châssis*****
- C- elle relie le fil neutre au châssis
- D- elle empêche la transmission de tensions dangereuses aux parties métalliques*****

Question-15 : Quel terme décrit la conversion de données binaires sous forme qui leur permet de circuler sur une liaison de communication physique ?

- A- le codage*****
- B- le décodage
- C- le cryptage
- D- le décryptage

Question-16 : dans quelle couche du modèle OSI se trouve la carte réseau ?

- A- liaison*****
- B- réseau
- C- transport

D- session

Question-17 : Quel est le terme utilisé pour décrire le nombre maximum de bits pouvant être transféré dans un délai donnée ?

- A- une impédance
- B- une propagation
- C- une atténuation
- D- une bande passante****

Question-18 : Lequel de ces éléments décrit le mieux l'objet d'une page Web menant à une nouvelle page lorsque vous cliquez dessus ?

- A- un logiciel de redirection réseau
- B- un lien hypertexte*****
- C- un navigateur Web
- D- le code ascii

Question-19 : La RS232C correspond à :

- A- une interface de programmation
- B- une interface de communication*****
- C- une application industrielle
- D- une norme de l'EIA*****

Question-20 : Dans un ordinateur les données sont représentées sous forme de signal :

- A- lumineux
- B- électrique numérique****
- C- électrique analogique
- D- alphanumérique

Question-21 : Un bus d'adresses sur 32 bits représente une capacité d'adressage maximale de :

- A- 32×2 (* puissance)
- B- 2×32 *****
- C- 32
- D- 1

Question-22 : IP (Internet Protocol) décrit :

- A- une messagerie électronique
- B- un transfert de fichier
- C- un protocole de communication****
- D- le réseau ethernet

Question-23 : En algorithmique on parle de deux types d'algorithmes, lesquels ?

- A- récursif****
- B- binaire
- C- itératif*****
- D- complexe

Question-24 : Quels sont les résultats du ou exclusif ?

- A- $1 + 1 = 0$ ****
- B- $1 + 1 = 1$
- C- $1 + 0 = 1$ ****
- D- $0 + 0 = 1$

Question-25 : Quel est le langage le plus proche de la machine (langage de bas niveau) ?

- A- java
- B- html
- C- pascal
- D- assembleur*****