

examen d'informatique

I enregistrement

Compléter la procédure

```
type frac=record deno,nume:integer end;
procedure addfrac(a,b:frac;var c:frac); // c:=a+b
  qui calcule par exemple  $5/6 + 3/14 = 22/21$ .
```

II fichiers de texte

Ecrire un programme qui copie un fichier de texte dans un autre en renversant l'ordre des lettres dans chaque mot. Par exemple, si le fichier `marquise.txt` contient
Marquise, vos beaux yeux
me font mourir d'amour.

Lors de l'exécution du programme on verra :

```
Quel fichier voulez-vous renverser ? marquise.txt
Comment voulez-vous appeler la copie renversée ? esiuqram.txt
```

Le fichier `esiuqram.txt` sera créé et contiendra :

```
esiuqraM, sov xuaeb xuey
em tnof riruom d'ruoma.
```

III fichier binaire

Compléter le programme

```
var f:file of integer; ...
assign(f,'A:TRUC.DAT');
reset(f); ...
  qui ajoute 2 à chacun des nombres impairs contenus dans le fichier A:TRUC.DAT.
```

IV tableaux

Compléter les procédures `litvec` et `litmat` dans le programme suivant.

Qu'affiche-t-il sur l'écran quand on l'exécute ?

Qu'y a-t-il dans le fichier `abc.txt` ?

```

const n=6;
type vec=array[1..n] of integer;
      mat=array[1..n] of vec;
procedure affvec(var f:text;a:vec);
var i:integer;
begin
  for i:=1 to n do write(f,a[i]:3);
  writeln(f)
end;
procedure affmat(var f:text;a:mat);
var i:integer;
begin
  writeln(f);
  for i:=1 to n do affvec(f,a[i])
end;
procedure afftam(var f:text;a:mat);
var i:integer;
begin
  writeln(f);
  for i:=n downto 1 do affvec(f,a[i])
end;
procedure litvec(var f:text;var a:vec); (* à compléter *)
procedure litmat(var f:text;var a:mat); (* à compléter *)
const a:vec=( , , , , , );
var b,c:vec;  m:mat;  i,j:integer;  f:text;
begin
  assign(f,'abc.txt'); rewrite(f);
  affvec(output,a);
  for i:=1 to n do b[a[i]]:=i;
  for i:=1 to n do c[i]:=a[a[i]];
  for i:=1 to n do for j:=1 to n do m[i,j]:=b[i]+c[j];
  affvec(output,c);      affvec(f,b);
  affvec(output,b);      affvec(f,a);
  affvec(output,a);      affvec(f,c);
  afftam(output,m);      affmat(f,m);
  affmat(output,m);
  for i:=n downto 1 do m[i,i]:=a[i];  affmat(output,m);
  for i:=n downto 1 do m[i] :=a ;  affmat(output,m);
  for i:=n downto 1 do m[i,i]:=b[i];  affmat(output,m);
  close(f); reset(f);
  litmat(f,m);
  affmat(output,m);
  close(f)
end.

```

barème

I enregistrement 2.5 à 5.5 points

Compléter la procédure

```
type frac=record deno,nume:integer end;
procedure addfrac(a,b:frac;var c:frac); // c:=a+b
  version simple :
begin
  c.den:=a.den*b.den;           1 point
  c.num:=a*den*b.num+a.num*b.den; 1 point
  reduit(c)                     0.5 point
end;
  version avec un pgcd :
var p:integer;
begin
  c.den:=a.den*b.den;           1 point
  c.num:=a*den*b.num+a.num*b.den; 1 point
  p:=pgcd(c.den,c.num);         0.5 point
  c.num:=c.num div p;           0.5 point
  c.den:=c.den div p            0.5 point
end;
  version avec deux pgcd :
var d,p:integer;
begin
  d:=pgcd(a.den,b.den);         0.5 point
  a.den:=a.den div p;           0.25 point
  b.den:=b.den div p;           0.25 point
  c.num:=a*den*b.num+a.num*b.den; 1 point
  p:=pgcd(d,c.num);             0.5 point
  c.num:=c.num div p;           0.5 point
  c.den:=d div p*a.den*b.den    1.5 point
end;
  plus 1 point pour la fonction pgcd.
  moins 1 point s'il y a des read
  moins 0,5 s'il y a un write
```

II fichiers de texte 6.5 points

Ecrire un programme qui copie un fichier de texte dans un autre en renversant l'ordre des lettres dans chaque mot.

```
var f,g:text;
    t:string;
    i,j,k:integer;
    c:char;
begin
```

```

write('Quel fichier voulez-vous renverser ? '); readln(t);      0,5 point
assign(f,t); reset(f);                                         0,5 point
write('Comment voulez-vous appeler la copie renversée ? ');readln(t); 0,5
assign(g,t); rewrite(g);                                       0,5 point
while not eof(f) do                                           0,5 point
begin
  readln(f,t);                                               0,5 point
  j:=1;
  for i:=1 to length(t)+1 do
  if(i>length(t)) or (upcase(t[i])<'A') or (upcase(t[i])>'Z') then 0,5 pt
  begin
    k:=i-1; (* on retourne le mot t[j..i-1] *)                2 points
    while j<k do
    begin
      c:=t[j]; t[j]:=t[k]; t[k]:=c;
      j:=j+1;
      k:=k-1
    end;
    j:=i+1
  end;
  writeln(g,t)      0,5 point
end;
close(f);          0,25 point
close(g)           0,25 point
end.

```

III fichier binaire 4 points

Compléter le programme

```

var f:file of integer; ...
assign(f,'A:TRUC.DAT');
reset(f); ...
  qui ajoute 2 à chacun des nombres impairs contenus dans le fichier A:TRUC.DAT.
while not eof(f) do      0,5 point
begin
  read(f,i);             0,5 point
  if odd(i) then        0,5 point
  begin
    i:=i+2;             0,5 point
    seek(f,filepos(f)-1); 1 point
    write(f,i)          0,5 point
  end
end;
close(f)                0,5 point

```

IV tableaux 13 points

Compléter les procédures `litvec` et `litmat` dans le programme suivant. 2 points

Qu'affiche-t-il sur l'écran quand on l'exécute ? 9 points

Qu'y a-t-il dans le fichier `abc.txt` ? $4 \times 0.5 = 2$ points

```
procedure litvec(var f:text;var a:vec); (* à compléter *) (* 1pt *)
var i:integer;
begin
  for i:=1 to n do read(f,a[i]);
  readln(f)
end;
procedure litmat(var f:text;var a:mat); (* à compléter *) (* 1pt *)
var i:integer;
begin
  for i:=1 to n do litvec(f,a[i])
end;
const a:vec=(1,3,4,2,6,5);
var b,c:vec;
    m:mat;
    i,j:integer;
    f:text;
begin
  assign(f,'abc.txt'); rewrite(f);
  affvec(output,a); (* 0.5pt *)
  for i:=1 to n do b[a[i]]:=i;
  for i:=1 to n do c[i]:=a[a[i]];
  for i:=1 to n do for j:=1 to n do m[i,j]:=b[i]+c[j];
  affvec(output,c);      affvec(f,b); (* 1pt *)
  affvec(output,b);      affvec(f,a); (* 1pt *)
  affvec(output,a);      affvec(f,c); (* 0.5pt *)
  afftam(output,m);      affmat(f,m); (* 1pt *)
  affmat(output,m);      (* 1pt *)
  for i:=n downto 1 do m[i,i]:=a[i];  affmat(output,m); (* 1pt *)
  for i:=n downto 1 do m[i] :=a ;    affmat(output,m); (* 1pt *)
  for i:=n downto 1 do m[i,i]:=b[i];  affmat(output,m); (* 1pt *)
  close(f); reset(f);
  litmat(f,m);
  affmat(output,m);      (* 1pt *)
  close(f)
end.
```