

TP Structure des Ordinateurs et Applications

Corrigé de la Série de TP N°4- (Tests : SI...FIN-SI SI...SINON...FIN-SI)

Structures de contrôle conditionnelles ou tests alternatifs

Ces structures sont utilisées pour décider de l'exécution d'un bloc d'instructions : est-ce qu'un bloc d'instruction sera exécuté ou non. Ou bien, pour choisir entre l'exécution de deux blocs différents.

Nous avons deux types de structures conditionnelles :

1. Structure conditionnelle simple :

Un test simple contient un seul bloc d'instructions. Selon une condition (expression logique), on décide est ce que le bloc d'instructions sera exécuté ou non. Si la condition est vraie, on exécute le bloc, sinon on ne l'exécute pas. La syntaxe d'un test alternatif simple est donnée comme suit :

<u>Si</u> (condition) <u>Alors</u> <Bloc_Inst_Si> ; <u>Fin-Si</u> ;	Traduit →	<u>if</u> (condition) <u>then</u> <u>begin</u> <Bloc_Inst_Si> ; <u>end</u> ;
---	-----------	---

2. Structure conditionnelle alternée ou double :

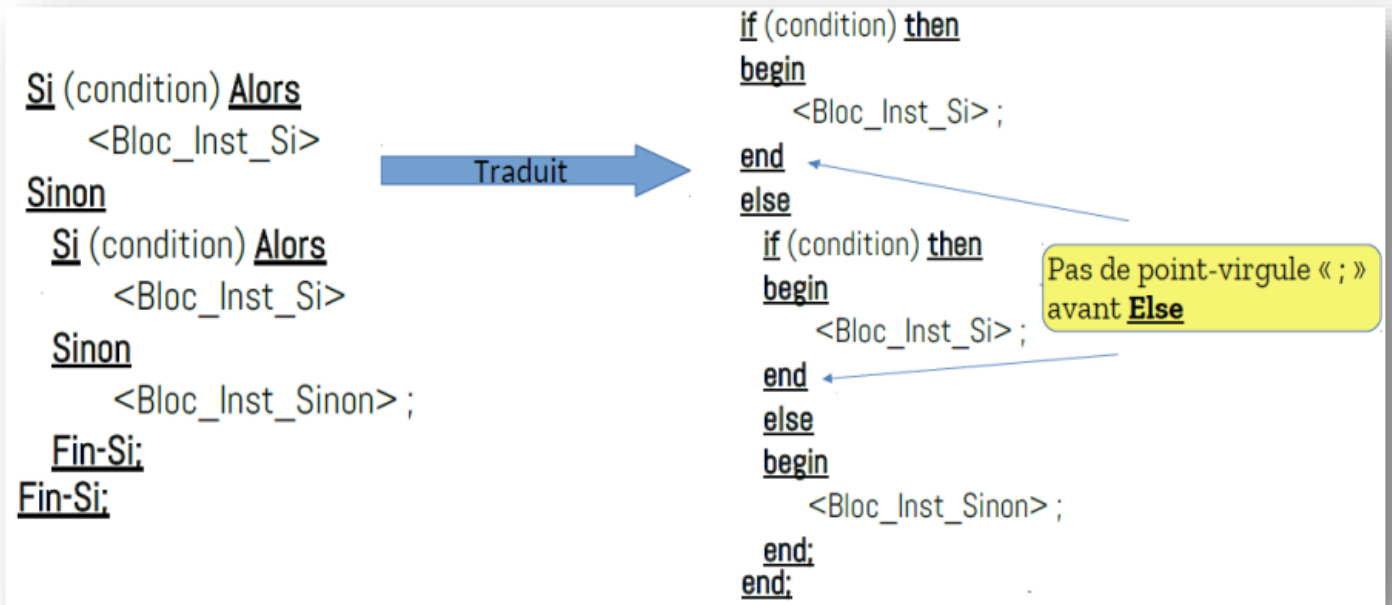
Un test double contient deux blocs d'instructions : on est amené à décider entre le premier bloc ou le second. Cette décision est réalisée selon une condition (expression logique ou booléenne) qui peut être vraie ou fausse. Si la condition est vraie on exécute le premier bloc, sinon on exécute le second. La syntaxe d'un test alternatif double est :

<u>Si</u> (condition) <u>Alors</u> <Bloc_Inst_Si> <u>Sinon</u> <Bloc_Inst_Sinon> ; <u>Fin-Si</u> ;	Traduit →	<u>if</u> (condition) <u>then</u> <u>begin</u> <Bloc_Inst_Si> ; <u>end</u> <u>else</u> <u>begin</u> <Bloc_Inst_Sinon> ; <u>end</u> ; <u>end</u> ;
--	-----------	---

Pas de point-virgule « ; » avant Else

Nous avons aussi, les **structures conditionnelles doubles et imbriquées** :

Un test double et imbriqué, tout comme un test double, contient deux blocs instructions avec au moins un des deux blocs (bloc Si et/ou bloc Sinon) est composé d'une instruction de condition simple ou double. Donc un test double et imbriqué contient au moins trois blocs d'instructions avec au moins deux conditions. La syntaxe d'un test alternatif double imbriqué avec trois blocs d'instructions est :



Dans les deux types de structure de contrôle conditionnelle, lorsque le bloc d'instructions est composé d'au moins deux instructions, les deux mots clés **begin** et **end** sont obligatoires dans le programme.

Par contre, si le bloc instruction est composé d'une seule instruction, les deux mots clés **begin** et **end** sont facultatifs (optionnels).

Par ailleurs, l'instruction qui précède immédiatement le mot clé Sinon ou Else ne doit pas se terminer par un « **point-virgule** »

Corrigé de l'exercice N°01 : (Algorithme → Programme)

1 / Traduction de l'algorithme en programme Pascal :

Algorithme	Code PASCAL
Algorithme exo_01; Variables N : Entier; Début Ecrire ("Entrez Un Nombre Entier : "); Lire (N); Si (N = 0) Alors Ecrire ("Le Nombre Est Nul Et Pair."); Sinon Si (N > 0) Alors Si (N Mod 2 = 0) Alors Ecrire ("Le Nombre Est Positif Et Pair."); Sinon Ecrire ("Le Nombre Est Positif Et Impair."); Finsi; Sinon Si (N Mod 2 = 0) Alors Ecrire ("Le Nombre Est Négatif Et Pair."); Sinon Ecrire ("Le Nombre Est Négatif Et Impair."); Finsi; Finsi; Finsi; Fin.	<pre> program exo_01; var N: integer; begin write('Entrez un nombre entier : '); readln(N); if (N = 0) then writeln('Le nombre est nul et pair.') else begin if (N > 0) then begin if (N mod 2 = 0) then writeln('Le nombre est positif et pair.') else writeln('Le nombre est positif et impair.'); end else begin if (N mod 2 = 0) then writeln('Le nombre est negatif et pair.') else writeln('Le nombre est negatif et impair.'); end; end; end. </pre>

2/ Compilation et exécution du programme pour: $N = 12$; $N = -9$; $N = 0$

The image shows a Pascal code editor on the left with the code for 'exo01.pas'. The code is as follows:

```

program exo_01;
var
  N: integer;
begin
  write('Entrez un nombre entier : ');
  readln(N);
  if (N = 0) then
    writeln('Le nombre est nul et pair.')
  else
    begin
      if (N > 0) then
        begin
          if (N mod 2 = 0) then
            writeln('Le nombre est positif et pair.')
          else
            writeln('Le nombre est positif et impair.');
        end
      else
        begin
          if (N mod 2 = 0) then
            writeln('Le nombre est negatif et pair.')
          else
            writeln('Le nombre est negatif et impair.');
        end;
      end;
    end.

```

On the right, there are three execution windows of 'MyPascal V1.20.5' showing the program's output for different input values:

- Window 1 (C:\Users\Maison\Desktop\...):**

```

Entrez un nombre entier : 12
Le nombre est positif et pair.

```
- Window 2 (C:\Users\Maison\Desktop\seri...):**

```

Entrez un nombre entier : -9
Le nombre est negatif et impair.

```
- Window 3 (C:...):**

```

Entrez un nombre entier : 0
Le nombre est nul et pair.

```

3/ Déroulement de l'algorithme pour:

☞ $N = -9$;

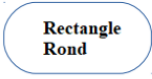


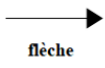



Instructions	Variables	Affichage
	N	
ECRIRE ("Entrez un nombre entier : "):	/	Entrez un nombre entier :
LIRE (N);	-9	/
SI (N = 0) Alors False ECRIRE ("Le nombre est nul et pair."):	-9	/
SINON SI (N > 0) Alors False	-9	/
SINON True ECRIRE ("Le nombre est négatif et impair."):	-9	Le nombre est négatif et impair.

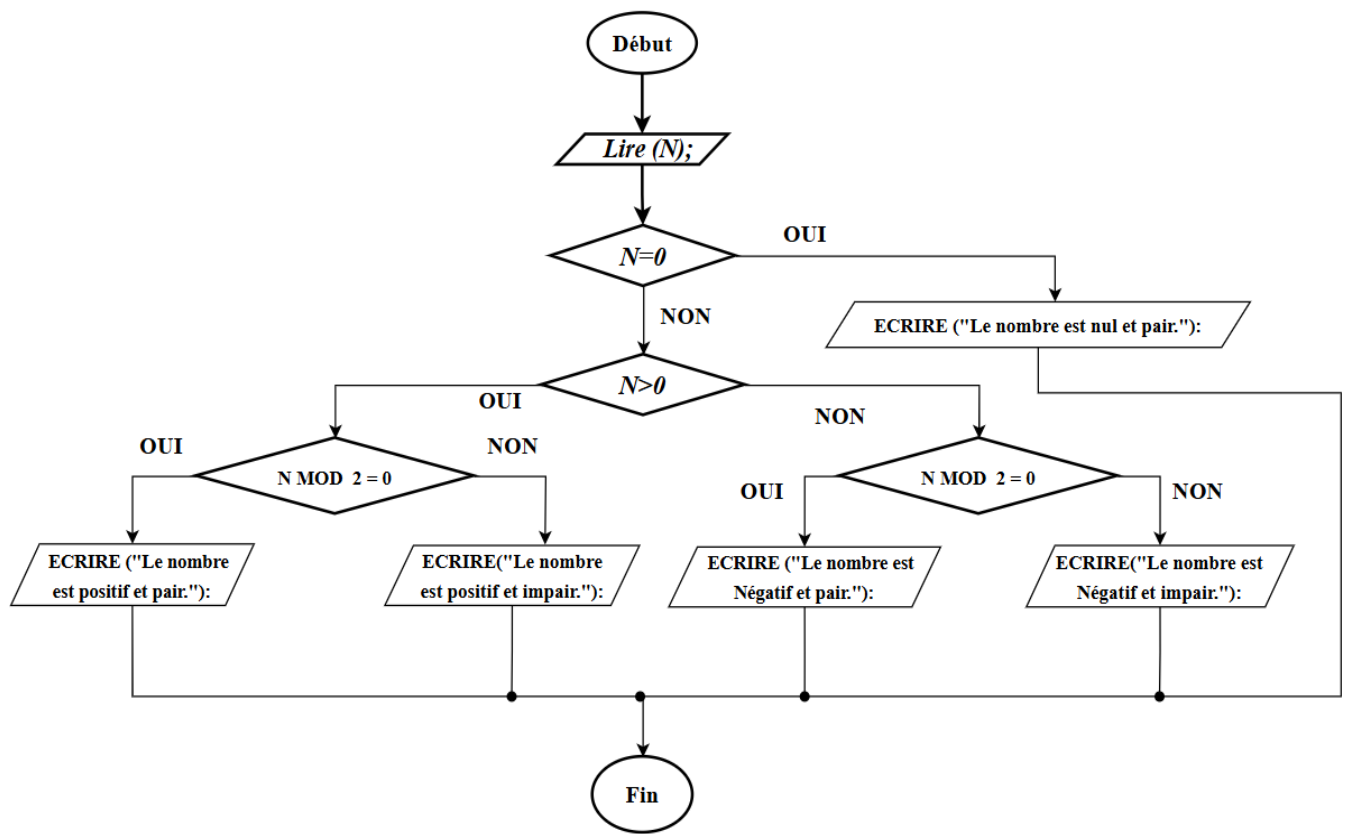
4/ Organigramme (Algorithme)

L'algorithme, ou organigramme, est un diagramme qui représente les étapes d'un algorithme sous forme visuelle. Voici les éléments clés à prendre en compte :

Symboles (Formes) courants (es) :

- **Rectangle** : Représente une action ou une opération (par exemple, une instruction ou une opération de calcul).
- **Losange** : Indique une décision (par exemple, une condition "oui" ou "non").
- **Rectangle rond** : Utilisé pour représenter le début ou la fin du processus.
- **Parallélogramme** : Est utilisé pour l'écriture et la lecture de données (entrées / sorties).
- **Flèches** : Montrent la direction du flux entre les étapes.

Les différentes formes d'organigramme (Algorithme)			
Formes	Sémantique / Sense	Formes	Sémantique / Sense
	Représente le début et la Fin de l'organigramme		Tests et décision : on écrit le test à l'intérieur du losange
	Entrées / Sorties : Lecture des données et écriture des résultats.		Ordre d'exécution des opérations (Enchaînement)
	Calculs, Traitements		Connecteur
	Sous-Programmes Portion du programme considérée comme une simple opération		

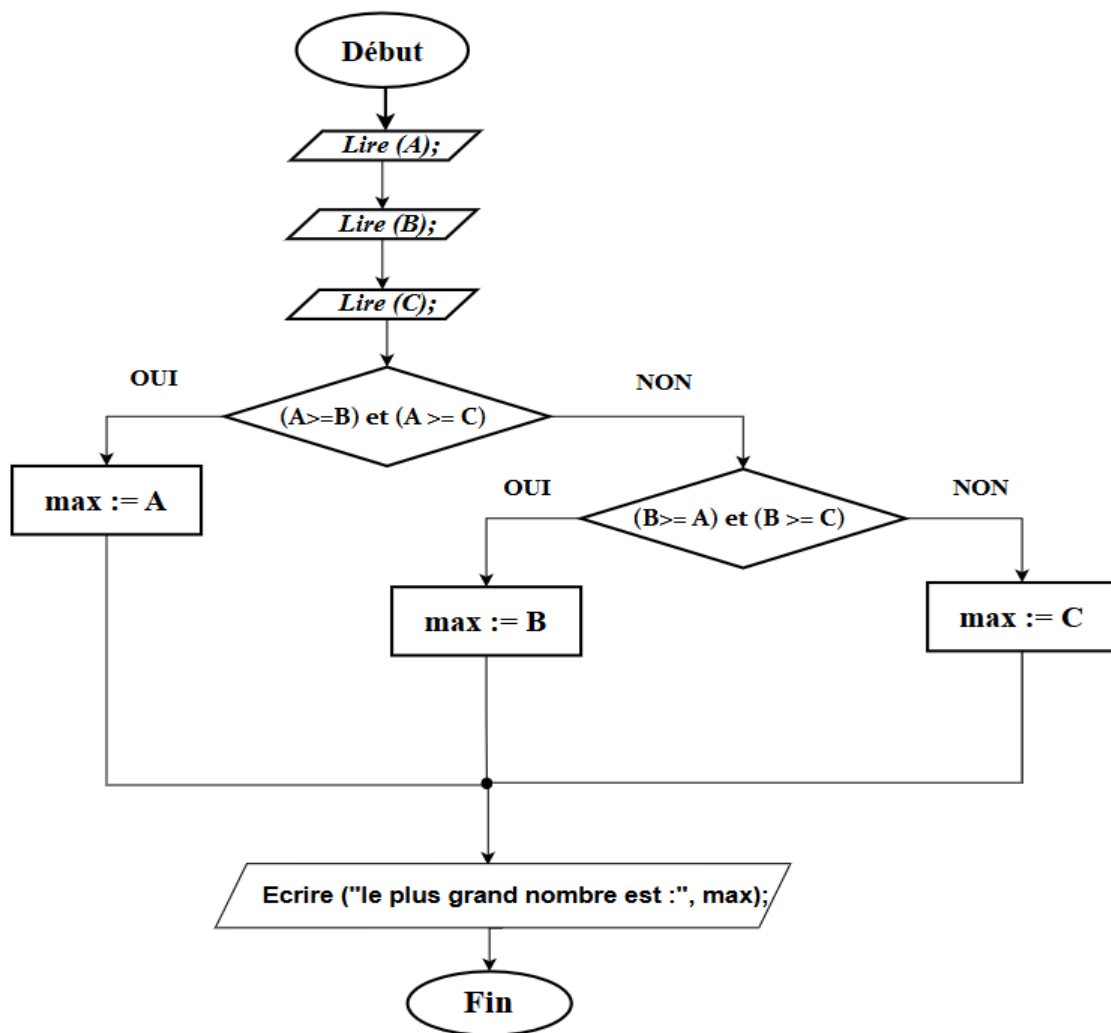


Corrigé de l'exercice N°02 :

1- Écrire un programme en Pascal qui demande à l'utilisateur de saisir trois nombres entiers. Le programme doit afficher le plus grand des trois.

Algorithme	Programme Pascal
Algorithme exercice_02 ; Variables A, B, C, max : entier Début Ecrire ("Entrez Le Premier Nombre : "); Lire (A); Ecrire ("Entrez Le Deuxieme Nombre : "); Lire (B); Ecrire ("Entrez Le Troisieme Nombre : "); Lire (C); Si (A >= B) et (A >= C) alors max ← A; Sinon; si (B >= A) et (B >= C) alors max ← B; Sinon max ← C; Finsi; Finsi; Ecrire ("Le Plus Grand Nombre Est : ", Max); Fin.	Program Exercice_02 ; Var A, B, C, max: integer; Begin Writeln('--- Determiner le plus grand de trois nombres ---'); Write('Entrez le premier nombre : '); Readln(A); Write('Entrez le deuxieme nombre : '); Readln(B); Write('Entrez le troisieme nombre : '); Readln(C); { Comparaison des trois nombres } If (A >= B) and (A >= C) then Max := A else if (B >= A) and (B >= C) then Max := B else Max := C; Writeln; Writeln('Le plus grand nombre est : ', max); End.

2- Donnez son Organigramme (Algorithme)



Compilation et Exécution du Programme sur My PASCAL

```
Program Exercice_02 ;
Var
A, B, C, max: integer;
Begin
  Writeln('--- Determiner le plus grand de trois nombres ---');
  Write('Entrez le premier nombre : ');
  Readln(A);
  Write('Entrez le deuxieme nombre : ');
  Readln(B);
  Write('Entrez le troisieme nombre : ');
  Readln(C);
  { Comparaison des trois nombres }
  If (A >= B) and (A >= C) then
    Max := A
  else if (B >= A) and (B >= C) then
    Max := B
  else
    Max := C;
  Writeln;
  Writeln('Le plus grand nombre est : ', max);
End.
```

MyPascal V1.20.5 (Exécution) C:\Users\Maison\De... — □ ×

```
--- Determiner le plus grand de trois nombres ---
Entrez le premier nombre : 13
Entrez le deuxieme nombre : -7
Entrez le troisieme nombre : 8
Le plus grand nombre est : 13
```

Corrigé de l'exercice N°03 : (Algorithmme → Programme)

Algorithmme	Programme Pascal
Algorithmme Exo03; Variables stockInitial, entree, sortie, stockRestant : entier; Début Ecrire ("entrez la quantité initiale en stock : "); Lire (stockInitial); Ecrire ("entrez la quantité de produit entrant : "); Lire (entree); Ecrire ("entrez la quantité de produit sortant : "); Lire (sortie); stockrestant ← stockinitial + entree – sortie; Si (Stockrestant < 0) Alors Ecrire "erreur : stock insuffisant !" Sinon Si (Stockrestant = 0) Alors Ecrire ("stock épuisé !"); Sinon Si (Stockrestant <= 10) Alors Ecrire ("stock faible : reste ", stockrestant, " produits"); Sinon Ecrire ("stock suffisant : reste ", stockrestant, " Produits"); Finsi; Fin.	program Exo03; var stockInitial, entree, sortie, stockRestant: integer; begin writeln('Entrez la quantite initiale en stock : '); readln(stockInitial); writeln('Entrez la quantite de produit entrant : '); readln(entree); writeln('Entrez la quantite de produit sortant : '); readln(sortie); stockRestant := stockInitial + entree - sortie; if stockRestant < 0 then writeln('Erreur : stock insuffisant !') else if stockRestant = 0 then writeln('Stock epuise !') else if stockRestant <= 10 then writeln('Stock faible : reste ', stockRestant, ' produits') else writeln('Stock suffisant : reste ', stockRestant, ' produits'); readln; end.

Compilation et Exécution du Programme sur My PASCAL

```

program Exo03;
var
    stockInitial, entree, sortie, stockRestant: integer;
begin
    writeln('Entrez la quantite initiale en stock : ');
    readln(stockInitial);
    writeln('Entrez la quantite de produit entrant : ');
    readln(entree);
    writeln('Entrez la quantite de produit sortant : ');
    readln(sortie);
    stockRestant := stockInitial + entree - sortie;
    if stockRestant < 0 then
        writeln('Erreur : stock insuffisant !')
    else if stockRestant = 0 then
        writeln('Stock epuise !')
    else if stockRestant <= 10 then
        writeln('Stock faible : reste ', stockRestant, ' produits')
    else
        writeln('Stock suffisant : reste ', stockRestant, ' produits');
    readln;
end.

```

MyPascal V1.20.5 (Exécution) C:\Users\Maiso...

```

Entrez la quantite initiale en stock :
125
Entrez la quantite de produit entrant :
300
Entrez la quantite de produit sortant :
200
Stock suffisant : reste 225 produits

```


Corrigé de l'exercice N°04 : (Programme PASCAL)

1/ Programme PASCAL

Algorithme	Programme Pascal
Algorithme Exo_04; Variables PU, QTE, TOT, REM, PAY : REEL Début Ecrire("Entrez Le Prix Unitaire Du Produit : "); Lire(Pu); Ecrire("Entrez La Quantite Achetee : "); Lire(Qte); TOT ← PU * QTE; Si (TOT < 100) ALORS REM ← 0; Sinon SI (TOT ≤ 500) ALORS REM ← TOT * 0.05; Sinon REM ← TOT * 0.10; Finsi; PAY ← TOT - REM; Ecrire("Montant Total : ", Tot, " Da"); Ecrire("Remise Appliquee : ", Rem, " Da"); Ecrire("Total A Payer : ", Pay, " Da"); Fin.	Program Exo_04; var PU, QTE, TOT, REM, PAY: real; Begin writeln('Entrez le prix unitaire du produit : '); readln(PU); writeln('Entrez la quantite achetee : '); readln(QTE); TOT := PU * QTE; if (TOT < 100) then REM := 0 else if (TOT ≤ 500) then REM := TOT * 0.05 else REM := TOT * 0.10; PAY := TOT - REM; writeln('Montant total : ', TOT:0:2, ' DA'); writeln('Remise appliquee : ', REM:0:2, ' DA'); writeln('Total a payer : ', PAY:0:2, ' DA'); readln; End.

Compilation et Exécution du Programme sur My PASCAL

```
Program Exo_04;
var
  PU, QTE, TOT, REM, PAY: real;
Begin

  writeln('Entrez le prix unitaire du produit : ');
  readln(PU);
  writeln('Entrez la quantite achetee : ');
  readln(QTE);

  TOT := PU * QTE;

  if (TOT < 100) then
    REM := 0
  else if (TOT ≤ 500) then
    REM := TOT * 0.05
  else
    REM := TOT * 0.10;

  PAY := TOT - REM;

  writeln('Montant total : ', TOT:0:2, ' DA');
  writeln('Remise appliquee : ', REM:0:2, ' DA');
  writeln('Total a payer : ', PAY:0:2, ' DA');

  readln;
End.
```

MyPascal V1.20.5 (Exécution) C:\Use...
Entrez le prix unitaire du produit :
135
Entrez la quantite achetee :
4
Montant total : 540.00 DA
Remise appliquee : 54.00 DA
Total a payer : 486.00 DA