

TP Informatique 2

Corrigé de la série de TP N°1 – Tableaux à une dimension - Vecteurs

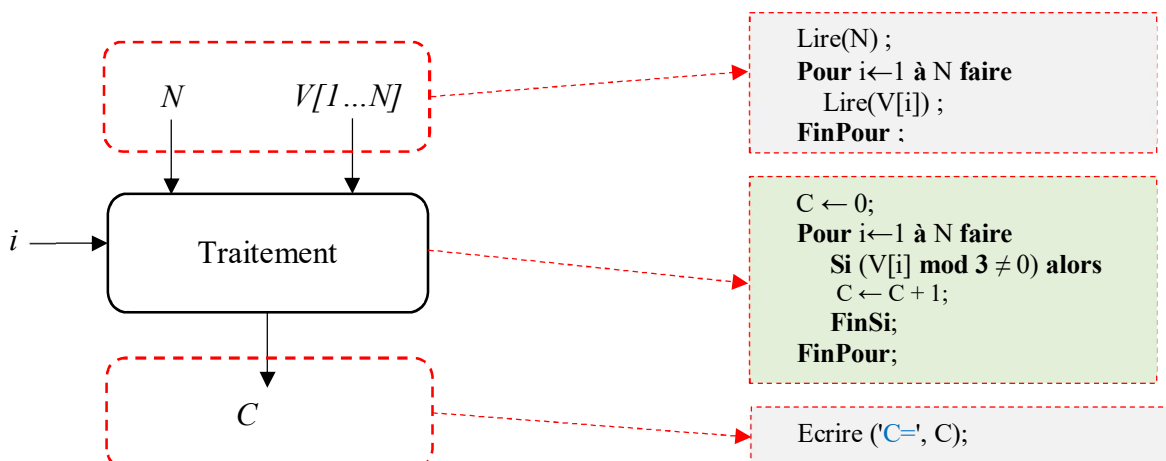
Exercice N°01 : Algorithme → Programme PASCAL

Soit l'algorithme suivant :

<p>Algorithme Vecteur;</p> <p>Variables V : Tableau [1..100] d'entier ; N,i, C : Entier;</p> <p>Début</p> <p style="color: red;">{-*-*- Entrées -*-*-}</p> <p>Ecrire ('Spécifiez la taille de V: '); Lire(N); Ecrire ('Donner les éléments de V: '); Pour i←1 à N faire Lire(V[i]); FinPour;</p> <p style="color: red;">{-*-*- Traitements -*-*-}</p> <p>C← 0; Pour i←1 à N faire Si (V[i] mod 3 ≠ 0) alors C← C+ 1; FinSi; FinPour;</p> <p style="color: red;">{-*-*- Sorties -*-*-}</p> <p>Ecrire (' C=', C) ;</p> <p>Fin.</p>	<p>Questions :</p> <ol style="list-style-type: none"> 1- Traduire l'algorithme en Programme PASCAL. 2- Compiler et exécuter le programme pour : N = 4 et V= [1, 3, 7, 9]. 3- Dérouler l'algorithme pour les valeurs de N et V ci-dessus ? 4- Déduire ce que fait l'algorithme ? 5- Modifier l'algorithme/programme pour calculer le produit P montré ci-dessous : $P = \prod_{i=1}^N (i * V[i])$
---	---

Solution :

Les variables d'entrée, variables de sortie et la partie traitement sont présentées dans le schéma ci-dessous :



1- Traduire l'algorithme en programme PASCAL

Algorithme	Programme Pascal
<p>Algorithme Vecteur;</p> <p>Variables V : Tableau [1..100] d'entier ; N,i, C : Entier;</p> <p>Début</p> <p>{-*-*- Entrées -*-*-}</p> <p>Ecrire ('Spécifiez la taille de V: '); Lire(N); Ecrire ('Donner les éléments de V: '); Pour i←1 à N faire Lire(V[i]); FinPour;</p> <p>{-*-*- Traitements -*-*-}</p> <p>C ← 0; Pour i←1 à N faire Si (V[i] mod 3 ≠ 0) alors C ← C + 1; FinSi; FinPour;</p> <p>{-*-*- Sorties -*-*-}</p> <p>Ecrire (' C=', C) ;</p> <p>Fin.</p>	<p>Program Vecteur ;</p> <p>Var V: array[1..100] of integer; N, i, C : integer;</p> <p>Begin</p> <p>{-*-*- Entrées -*-*-}</p> <p>Write(' Spécifiez la taille de V: '); Read(N); Write('Donner les éléments de V:'); For i:=1 to N do {Lecture des éléments du vecteur V} Read(V[i]);</p> <p>{-*-*- Traitements -*-*-}</p> <p>C := 0; For i:=1 to N do { recherche des éléments qui ne sont pas multiples de 3} if (V[i] mod 3 <> 0) then C:=C+1; { compter les éléments qui ne sont pas divisibles par 3}</p> <p>{-*-*- Sorties -*-*-}</p> <p>Write (' C=',C); { Affichage du résultat Le nombre d"éléments qui ne sont pas multiples de 3 }</p> <p>End.</p>

2- Compiler et exécuter le programme pour : N = 4 et T=[1, 3, 7, 9].

The image shows a screenshot of a Pascal IDE. On the left, the source code for 'MyWork 011.pas' is displayed, corresponding to the Pascal program in the table above. The code is as follows:

```

1 Program Vecteur ;
2 Var
3   V: array[1..100] of integer;
4   N, i, C : integer;
5 Begin
6   {-*-*- Entrées -*-*-}
7   Write(' Spécifiez la taille de V: ');
8   Read(N);
9   Write('Donner les éléments de V:');
10  For i:=1 to N do {Lecture des éléments du vecteur V}
11    Read(V[i]);
12
13  {-*-*- Traitements -*-*-}
14  C := 0;
15  For i:=1 to N do { recherche des éléments qui ne sont pas multiples de 3}
16    if (V[i] mod 3 <> 0) then
17      C:=C+1; { compter les éléments qui ne sont pas divisibles par 3}
18
19  {-*-*- Sorties -*-*-}
20  Write (' C=',C); { Affichage du résultat Le nombre d"éléments qui ne sont pas multiples de 3 }
21 End.
22

```

On the right, a window titled 'MyPascal V1.20.5 (Exéc)' shows the execution output:

```

Spécifiez la taille de V: 4
Donner les éléments de V:1 3 7 9
C= 2

```

3- Dérouler l'algorithme pour : N = 4 et T=[1, 3, 7, 9].

Dérouler un algorithme (ou un programme) consiste à exécuter manuellement les instructions de cet algorithme et à visualiser l'impact de ces instructions sur les variables.

Autrement dit, dérouler un algorithme permet de visualiser les changements des valeurs des variables (évolution des valeurs).

Instructions	Variables				Affichage
	N	i	V	C	
Ecrire(' Spécifiez la taille de V: ');	/	/	/	/	Spécifiez la taille de V:
Lire(N)	4	/	/	/	//
Ecrire('Donner les éléments de V: ');	//	/	/	/	Donner les éléments de V::
Pour i←1 à N faire Lire(T[i]) Fin-Pour	//	1 2 3 4	1 2 3 4 [1, 3, 7, 9]	/	//
C ← 0	//	/	//	0	//
Pour i=1 Si V[1] mod 3 <> 0 Alors 1 mod 3 <> 0 True C ← C + 1; C ← 0+1=1;	//	1	//	1	//
Pour i=2 Si V[2] mod 3 <> 0 Alors 3 mod 3 <> 0 False	//	2	//	//	//
Pour i=3 Si V[3] mod 3 <> 0 Alors 7 mod 3 <> 0 True C ← C + 1 ; C ← 1+1=2	//	3	//	2	//
Pour i=4 Si V[4] mod 3 <> 0 Alors 9 mod 3 <> 0 false	//	4	//	2	//
Ecrire('C = ', C) ;	//		//	//	C=2

4- Dédurre la fonction de cet algorithme ?

L'algorithme a pour but **de compter le nombre d'éléments dans un vecteur qui ne sont pas des multiples de 3.**

5- Modifier l'algorithme pour calculer le produit de chaque élément V[i] du tableau multiplié par son indice i, c'est-à-dire calculer :

$$P = \prod_{i=1}^N (i * V[i])$$

```

MyWork 0111.pas
1 Program Vecteur ;
2 Var
3   V: array[1..100] of integer;
4   N, i: integer;
5   P: integer;
6 Begin
7
8   {-*- Entrées -*-}
9   Write('Donner la taille du vecteur V : ');
10  Read(N);
11  Writeln('Donner les composantes du vecteur V : ');
12  For i := 1 to N do {Lecture des éléments du vecteur}
13    Read(V[i]);
14  .....
15  {-*- Traitements -*-}
16  P := 1; {Initialisation du produit à 1}
17
18  For i := 1 to N do {Parcourir chaque élément du vecteur}
19    P := P * (i * V[i]); {Calculer le produit (i * V[i]) et mettre à jour P}
20  .....
21  {-*- Sorties -*-}
22  Writeln('Le produit P = ', P); {Affichage du produit final}
23
24 End.

```

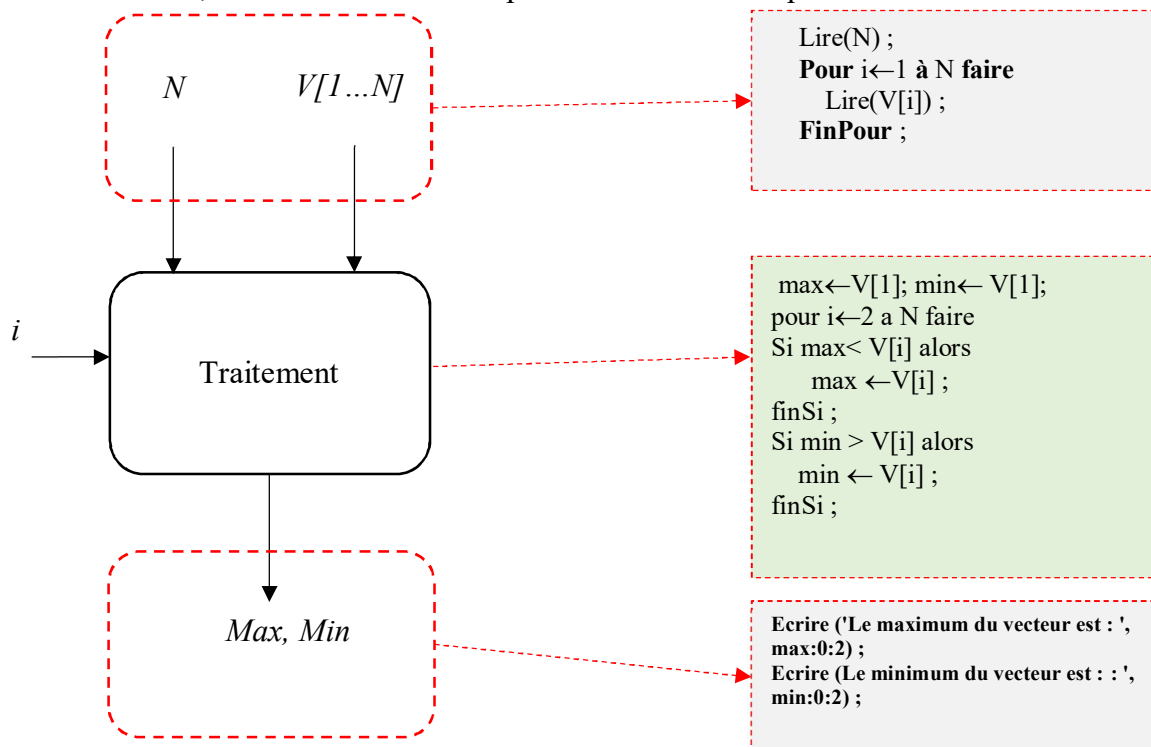
Exercice N°02 : Lecture et analyse d'un vecteur

Écrire un programme Pascal qui permet de :

1. Saisir un vecteur V de N réels (où N est un nombre donné par l'utilisateur).
2. Trouver et afficher le maximum et le minimum des éléments du vecteur.

Solution :

Les variables d'entrée, variable de sortie et la partie traitement sont présentées sur le schéma ci-dessous :



Algorithme/Programme PASCAL

Algorithme	Programme PASCAL
<p>Algorithme MaxMin_Vecteur;</p> <p>Variables</p> <p>V : Tableau [1..100] de réel ; i, N : entier ; Max, Min : réel ;</p> <p>Début</p> <p>{-*-*- Entrées -*-*-}</p> <p>Ecrire('Donner la taille du vecteur V : '); Lire(N); Ecrire('Donner les composantes du vecteur V : ');</p> <p>Pour i ← 1 à N faire Lire(V[i]); FinPour ;</p> <p>{-*-*- Traitement -*-*-}</p> <p>Max ← V[1]; Min ← V[1];</p> <p>Pour i ← 2 à N faire Si V[i] > Max alors Max ← V[i]; Finsi ; Si V[i] < Min alors Min ← V[i]; Finsi ;</p> <p>FinPour ; Ecrire('Le maximum du vecteur est : ', Max:8:2); Ecrire('Le minimum du vecteur est : ', Min:8:2);</p> <p>Fin.</p>	<p>Program VecteurMaxMin;</p> <p>Var</p> <p>V : array [1..100] of real ; i, N : integer ; Max, Min: real;</p> <p>Begin</p> <p>{-*-*- Entrées -*-*-}</p> <p>Write('Donner la taille du vecteur V : '); Read(N); Writeln('Donner les composantes du vecteur V : ');</p> <p>For i := 1 to N do Read(V[i]);</p> <p>{-*-*- Traitements -*-*-}</p> <p>Max := V [1]; Min := V [1];</p> <p>For i := 2 to N do</p> <p> Begin if V[i] > Max then Max := V[i]; if V[i] < Min then Min := V[i]; End;</p> <p>{-*-*- Sorties -*-*-}</p> <p>Writeln('Le maximum du vecteur est : ', Max:0:2); Writeln('Le minimum du vecteur est : ', Min:0:2);</p> <p>End.</p>

Exercice N°03 : Recherche d'une valeur réelle dans un vecteur

Écrire un programme Pascal qui permet de :

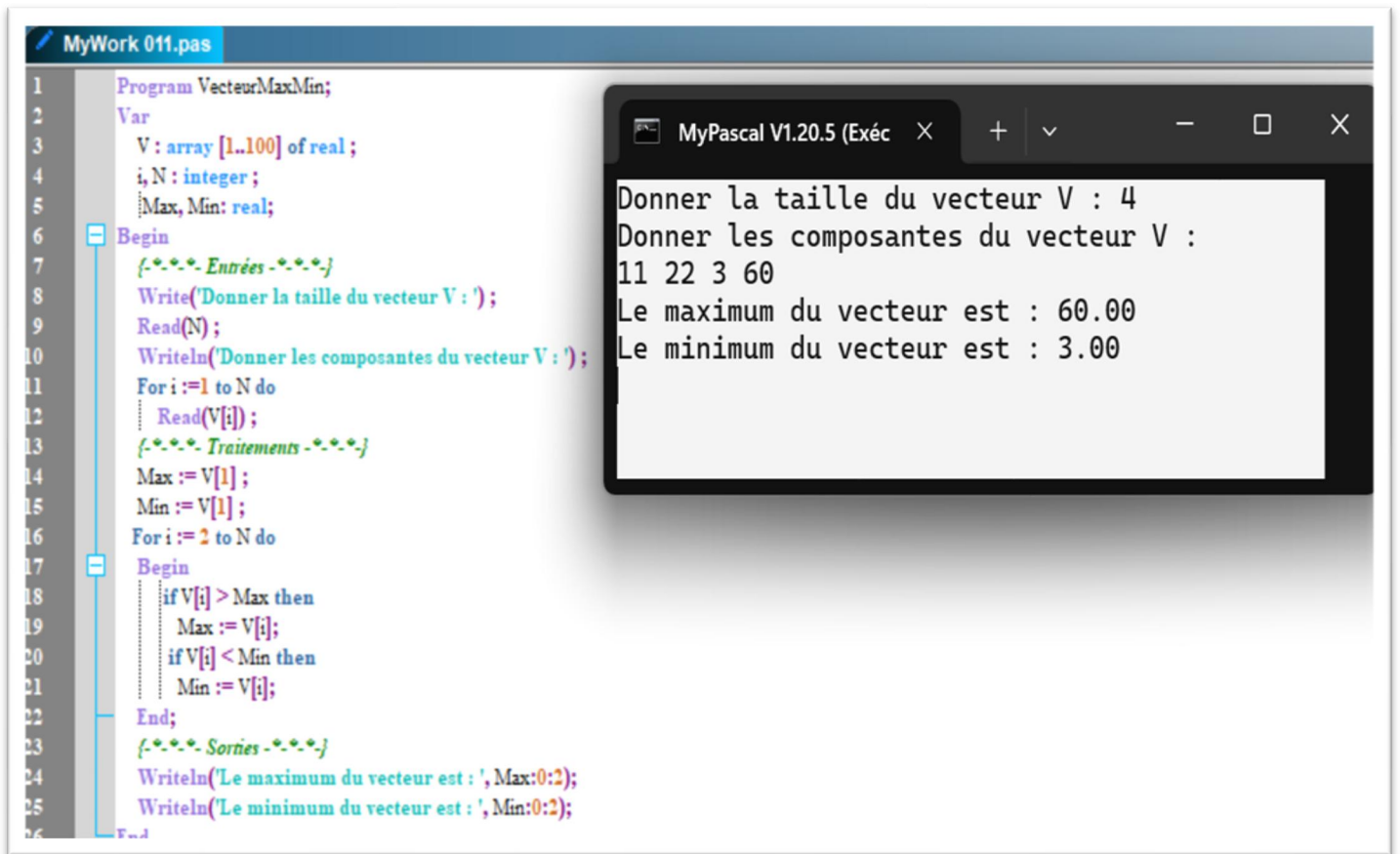
1. Saisir un vecteur V de N réels.
2. Saisir une valeur X .
3. Vérifier si X est présent dans le vecteur et afficher son indice. Si X n'est pas trouvé, afficher le message suivant :
"La valeur X n'existe pas dans V ."

Solution :

Les variables d'entrée, variables de sortie et la partie traitement sont présentées dans le schéma ci-dessous

Algorithme/Programme PASCAL

Algorithme	Programme PASCAL
<p>Algorithme Recherchedunevaleur ;</p> <p>Variables</p> <p>T : Tableau [1..100] de réel ;</p> <p>i, N: entier ;</p> <p>X :réel ;</p> <p>Existe : booléen ;</p> <p>Début</p> <p style="text-align: center;">{-*-*- Entrées -*-*-}</p> <p>Ecrire('Spécifiez la taille de V: ');</p> <p>Lire(N) ;</p> <p>Ecrire('Donner les éléments de V: ');</p> <p>Pour i←1 à N faire</p> <p style="padding-left: 20px;">Lire(V[i]) ;</p> <p>FinPour ;</p> <p>Ecrire('Choisir une valeur de X:');</p> <p>Lire(X) ;</p> <p>Existe ← faux ;</p> <p style="text-align: center;">{-*-*- Traitement -*-*-}</p> <p>Pour i←1 à N faire</p> <p style="padding-left: 20px;">Si V[i] = X alors</p> <p style="padding-left: 40px;">Existe ← vrai ;</p> <p style="padding-left: 40px;">Ecrire('La valeur ', X:8:2, ' est trouvée à la position ', i) ;</p> <p style="padding-left: 20px;">FinSi ;</p> <p style="padding-left: 20px;">FinPour ;</p> <p style="text-align: center;">{-*-*- Sorties -*-*-}</p> <p>Si (Existe= faux) alors</p> <p>Ecrire('La valeur ', X:8:2, ' n'existe pas dans le tableau.');</p> <p>FinSi ;</p> <p>Fin.</p>	<p>Program Recherchedunevaleur;</p> <p>Var</p> <p>V : array [1..100] Of real ;</p> <p>N, i : integer;</p> <p>X: real;</p> <p>Existe: boolean;</p> <p>Begin</p> <p style="text-align: center;">{-*-*- Entrées -*-*-}</p> <p>write('Spécifiez la taille de V: ');</p> <p>read(N);</p> <p>write ('Donner les éléments de V: ');</p> <p>For i:=1 To N Do</p> <p style="padding-left: 20px;">Begin</p> <p style="padding-left: 40px;">read(V[i]);</p> <p style="padding-left: 20px;">End;</p> <p>write('Choisir une valeur de X: ');</p> <p>read(X);</p> <p style="padding-left: 20px;">Existe := false;</p> <p style="text-align: center;">{-*-*- Traitement -*-*-}</p> <p>For i:=1 To N Do</p> <p style="padding-left: 20px;">Begin</p> <p style="padding-left: 40px;">If (V[i]=X) Then</p> <p style="padding-left: 60px;">Begin</p> <p style="padding-left: 80px;">Existe := true;</p> <p style="padding-left: 60px;">writeln ('l'indice de la valeur',X, 'est' ,i);</p> <p style="padding-left: 40px;">End;</p> <p style="padding-left: 20px;">End;</p> <p style="text-align: center;">{-*-*- Sorties -*-*-}</p> <p>If (existe = false) Then</p> <p style="padding-left: 20px;">write ('la valeur ',X, ' n'ésiste pas dans V');</p> <p>End.</p>



Exercice 04 : Calculs avec des vecteurs

On considère les deux vecteurs suivants qui représentent la position de deux points matériels dans l'espace :

$$v_1 = (1, -1, 2); v_2 = (0, -3, 2).$$

Écrire un programme en Pascal qui permet de :

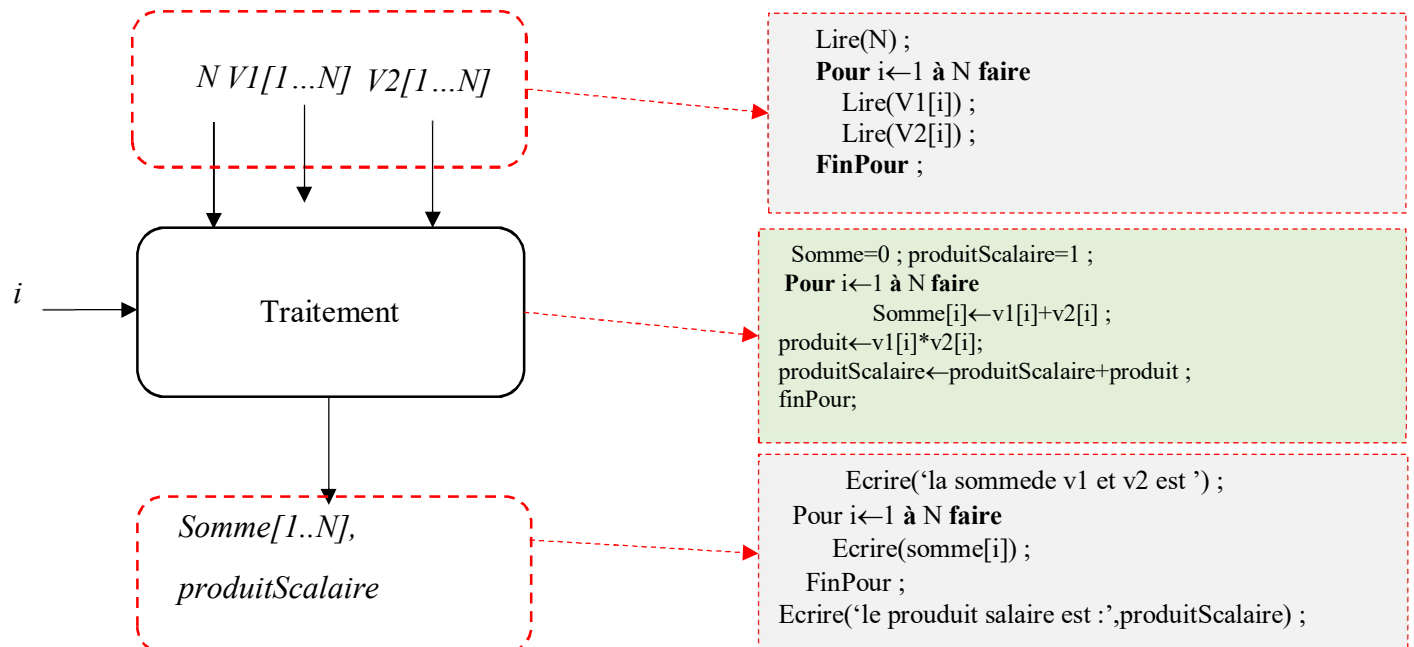
1. Calculer la somme des vecteurs v_1 et v_2 et leur produit scalaire.
2. Calculer le module de chaque vecteur.

On donne : $v_1 \cdot v_2 = \sum_{i=1}^N v_1[i] * v_2[i]$ et $\|v_{xyz}\| = \sqrt{x^2 + y^2 + z^2}$

Solution 1 (méthode 1):

Question 01 :

Les variables d'entrée, variable de sortie et la partie traitement sont présentées dans le schéma ci-dessous :



Algorithme	Programme PASCAL
<p>Algorithme SommeEtProduitScalaire</p> <p>Variables</p> <p>v1, v2, somme : Tableau[1..3] de réel ;</p> <p>i : entier ;</p> <p>produitScalaire : réel ;</p> <p>Début</p> <p style="text-align: center;">{-*-*- Entrées *-*-*-}</p> <p>Ecrire('Donner les éléments de V1 :'); Pour i ← 1 à 3 faire Lire(v1[i]); FinPour ;</p> <p>Ecrire('Donner les éléments de V2 :'); Pour i ← 1 à 3 faire Lire(v2[i]); FinPour ;</p> <p style="text-align: center;">{-*-*- Traitement *-*-*-}</p> <p>ProduitScalaire ← 0 ;</p> <p>Pour i ← 1 à 3 faire somme[i] ← v1[i] + v2[i] ; produitScalaire ← produitScalaire + v1[i] * v2[i] ; FinPour ;</p> <p style="text-align: center;">{-*-*- Sorties *-*-*-}</p> <p>Ecrire('La somme des vecteurs v1 et v2 est :'); Pour i ← 1 à 3 faire Ecrire(somme[i] : 0 : 2, ' '); FinPour ;</p> <p>Ecrire('Le produit scalaire entre v1 et v2 est : ', produitScalaire : 0 : 2) ; Fin.</p>	<p>Program SommeEtProduitScalaire;</p> <p>Var</p> <p>v1, v2, somme: array[1..3] of real;</p> <p>i : integer;</p> <p>produitScalaire: real;</p> <p>Begin</p> <p style="text-align: center;">{-*-*- Entrées *-*-*-}</p> <p>write('Donner les éléments de V1: '); for i := 1 to 3 do begin read(v1[i]); end;</p> <p>write('Donner les éléments de V2: '); for i := 1 to 3 do begin read(v2[i]); end;</p> <p style="text-align: center;">{-*-*- Traitement *-*-*-}</p> <p>ProduitScalaire := 0;</p> <p>for i := 1 to 3 do begin somme[i] := v1[i] + v2[i]; produitScalaire := produitScalaire + v1[i] * v2[i]; end;</p> <p style="text-align: center;">{-*-*- Sorties *-*-*-}</p> <p>write('La somme des vecteurs v1 et v2 est :'); for i := 1 to 3 do begin write(somme[i]:0:2); end;</p> <p>write('Le produit scalaire entre v1 et v2 est : ', produitScalaire:0:2); end.</p>


```

MyWork 012.pas  MyWork 011.pas
1 Program SommeEtProduitScalaire;
2 Var
3   v1, v2, somme: array[1..3] of real; i : integer; produitScalaire: real;
4 Begin
5   {-.-.-.- Entrées -.-.-.-}
6   write('Donner les éléments de V1: ');
7   for i := 1 to 3 do
8     begin
9       read(v1[i]);
10    end;
11  write('Donner les éléments de V2: ');
12  for i := 1 to 3 do
13    begin
14      read(v2[i]);
15    end;
16  {-.-.-.- Traitement -.-.-.-}
17  produitScalaire := 0;
18  for i := 1 to 3 do
19    begin
20      somme[i] := v1[i] + v2[i];
21      produitScalaire := produitScalaire + v1[i] * v2[i]; end;
22  {-.-.-.- Sorties -.-.-.-}
23  write('La somme des vecteurs v1 et v2 est:');
24  for i := 1 to 3 do
25    begin
26      write(somme[i]:0:2, ' ');
27    end;
28  write('Le produit scalaire entre v1 et v2 est : ', produitScalaire:0:2);
29 end.

```

MyPascal V1.20.5 (Exécution)

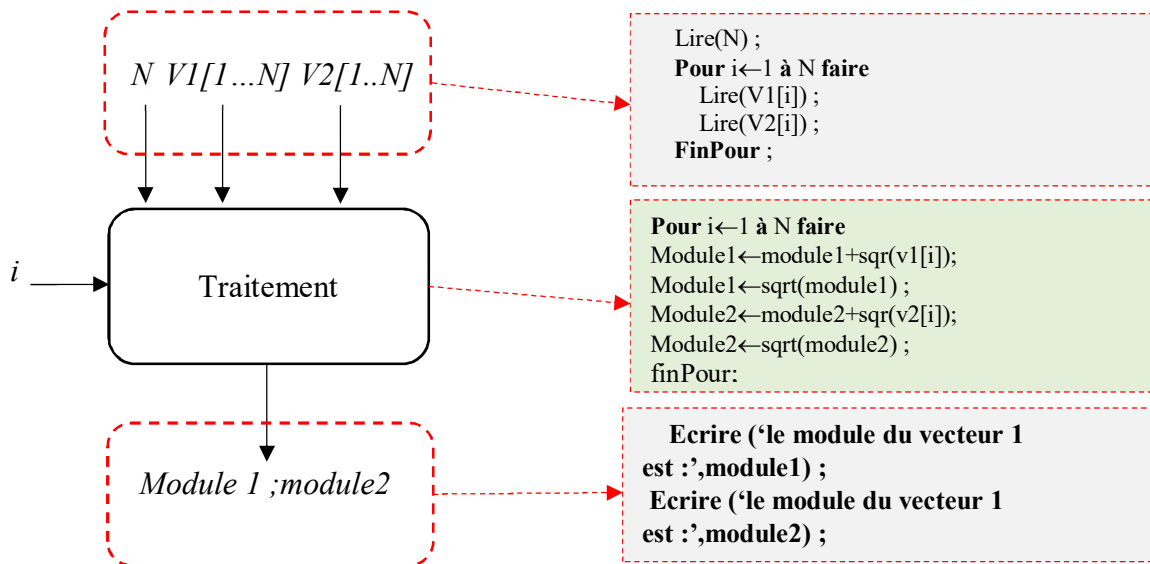
```

Donner les éléments de V1: 1 -1 2
Donner les éléments de V2: 0 -3 2
La somme des vecteurs v1 et v2 est :1.00 -4.00 4.00
Le produit scalaire entre v1 et v2 est : 7.00

```

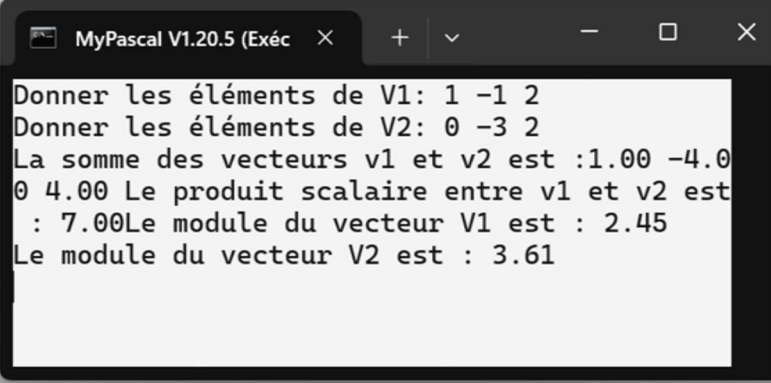
Question 02 :

Les variables d'entrée, de sortie et la partie traitement sont présentées dans le schéma ci-dessous :



Algorithme	Programme PASCAL
<p>Algorithme SommeEtProduitScalaire</p> <p>Variables</p> <p>v1, v2, somme : Tableau[1..3] de réel ;</p> <p>i : entier ;</p> <p>produitScalaire, moduleV1, moduleV2 : réel ;</p> <p>Début</p> <p style="text-align: center;">{-*-*- Entrées *-*-*-}</p> <p>Ecrire('Donner les éléments de V1 :'); Pour i ← 1 à 3 faire Lire(v1[i]); FinPour ;</p> <p>Ecrire('Donner les éléments de V2 :'); Pour i ← 1 à 3 faire Lire(v2[i]); FinPour ;</p> <p style="text-align: center;">{-*-*- Traitement *-*-*-}</p> <p>ProduitScalaire ← 0 ; moduleV1 ← 0; moduleV2 ← 0;</p> <p>Pour i ← 1 à 3 faire somme[i] ← v1[i] + v2[i] ; produitScalaire ← produitScalaire + v1[i] * v2[i] ; moduleV1 ← moduleV1 + sqr(v1[i]); moduleV2 ← moduleV2 + sqr(v2[i]);</p> <p>FinPour ; moduleV1 ← sqrt(moduleV1); moduleV2 ← sqrt(moduleV2);</p> <p style="text-align: center;">{-*-*- Sorties *-*-*-}</p> <p>Ecrire('La somme des vecteurs v1 et v2 est :'); Pour i ← 1 à 3 faire Ecrire(somme[i] : 0 : 2, ' '); FinPour ;</p> <p>Ecrire('Le produit scalaire entre v1 et v2 est : ', produitScalaire : 0 : 2); Ecrire('Le module du vecteur V1 est : ', moduleV1:0:2); Ecrire('Le module du vecteur V2 est : ', moduleV2:0:2); Fin.</p>	<p>Program SommeEtProduitScalaire;</p> <p>Var</p> <p>v1, v2, somme: array[1..3] of real;</p> <p>i : integer;</p> <p>produitScalaire, moduleV1, moduleV2: real;</p> <p>Begin</p> <p style="text-align: center;">{-*-*- Entrées *-*-*-}</p> <p>write('Donner les éléments de V1: '); for i := 1 to 3 do begin read(v1[i]); end;</p> <p>write('Donner les éléments de V2: '); for i := 1 to 3 do begin read(v2[i]); end;</p> <p style="text-align: center;">{-*-*- Traitement *-*-*-}</p> <p>ProduitScalaire := 0; moduleV1 := 0; moduleV2 := 0;</p> <p>for i := 1 to 3 do begin somme[i] := v1[i] + v2[i]; produitScalaire := produitScalaire + v1[i] * v2[i]; moduleV1 := moduleV1 + sqr(v1[i]); moduleV2 := moduleV2 + sqr(v2[i]);</p> <p>end;</p> <p>moduleV1 := sqrt(moduleV1); moduleV2 := sqrt(moduleV2);</p> <p style="text-align: center;">{-*-*- Sorties *-*-*-}</p> <p>write('La somme des vecteurs v1 et v2 est :'); for i := 1 to 3 do begin write(somme[i]:0:2, ' '); end;</p> <p>write('Le produit scalaire entre v1 et v2 est : ', produitScalaire:0:2); writeln('Le module du vecteur V1 est : ', moduleV1:0:2); writeln('Le module du vecteur V2 est : ', moduleV2:0:2); end.</p>

```
MyWork 013.pas
1 Program SommeEtProduitScalaire;
2 Var v1, v2, somme: array[1..3] of real; i: integer; produitScalaire, moduleV1, moduleV2: real;
3 Begin
4     {-*- Entrées -*-}
5     write('Donner les éléments de V1: ');
6     for i := 1 to 3 do
7         begin
8             read(v1[i]);
9         end;
10    write('Donner les éléments de V2: ');
11    for i := 1 to 3 do
12        begin
13            read(v2[i]);
14        end;
15    {-*- Traitement -*-}
16    produitScalaire := 0; moduleV1 := 0; moduleV2 := 0;
17    for i := 1 to 3 do
18        begin
19            somme[i] := v1[i] + v2[i]; produitScalaire := produitScalaire + v1[i] * v2[i];
20            moduleV1 := moduleV1 + sqr(v1[i]); moduleV2 := moduleV2 + sqr(v2[i]);
21        end; moduleV1 := sqrt(moduleV1); moduleV2 := sqrt(moduleV2);
22    {-*- Sorties -*-}
23    write('La somme des vecteurs v1 et v2 est:');
24    for i := 1 to 3 do
25        begin
26            write(somme[i]:0:2, ' ');
27        end;
28    write('Le produit scalaire entre v1 et v2 est: ', produitScalaire:0:2);
29    writeln('Le module du vecteur V1 est: ', moduleV1:0:2); writeln('Le module du vecteur V2 est: ', moduleV2:0:2);
30 end
```



Solution 2 (méthode 2):

Programme PASCAL

```
program exercice_01;
var
V1, V2, S: array[1..3] of real;
i: integer;
m1, m2, PS, S1, S2: real;
Begin

                                     {-*-*- Entrées -*-*-}
write('Donner les éléments de V1: ');
  for i := 1 to 3 do
    begin
      read(v1[i]);
    end;
write('Donner les éléments de V2: ');
  for i := 1 to 3 do
    begin
      read(v2[i]);
    end;

                                     {-*-*- Traitement -*-*-}
PS := 0;
S1 := 0;
S2 := 0;
for i := 1 to 3 do
  begin
    S[i] := V1[i] + V2[i];
    PS := PS + V1[i] * V2[i];
    S1 := S1 + sqr(V1[i]);
    S2 := S2 + sqr(V2[i]);
  end;
m1 := sqrt(S1);
m2 := sqrt(S2);

                                     {-*-*- Sorties -*-*-}
writeln('Le module de V1 est : ', m1:0:2);
writeln('Le module de V2 est : ', m2:0:2);
writeln('Le produit scalaire entre V1 et V2 est : ', PS:0:2);
writeln('La somme de V1 et V2 est : ', S[1]:0:2, ', ', S[2]:0:2, ', ', S[3]:0:2);
End.
```

```
1 program exercice_01;
2 var V1, V2, S: array[1..3] of real; i: integer; m1, m2, PS, S1, S2: real;
3 Begin
4     {-*- Entrées -*-}
5     write('Donner les éléments de V1: ');
6     for i := 1 to 3 do
7         begin
8             read(v1[i]);
9         end;
10    write('Donner les éléments de V2: ');
11    for i := 1 to 3 do
12        begin
13            read(v2[i]);
14        end;
15
16    {-*- Traitement -*-}
17    PS := 0; S1 := 0; S2 := 0;
18    for i := 1 to 3 do
19        begin
20            S[i] := V1[i] + V2[i];
21            PS := PS + V1[i] * V2[i];
22            S1 := S1 + sqr(V1[i]);
23            S2 := S2 + sqr(V2[i]);
24        end;
25    m1 := sqrt(S1); m2 := sqrt(S2);
26    {-*- Sorties -*-}
27    writeln('Le module de V1 est : ', m1:0:2); writeln('Le module de V2 est : ', m2:0:2);
28    writeln('Le produit scalaire entre V1 et V2 est : ', PS:0:2);
29    writeln('La somme de V1 et V2 est : ', S[1]:0:2, ' ', S[2]:0:2, ' ', S[3]:0:2);
30 End.
```

```
MyPascal V1.20.5 (Exécution)
Donner les éléments de V1: 1 -1 2
Donner les éléments de V2: 0 -3 2
Le module de V1 est : 2.45
Le module de V2 est : 3.61
Le produit scalaire entre V1 et V2 est : 7.00
La somme de V1 et V2 est : 1.00 -4.00 4.00
```