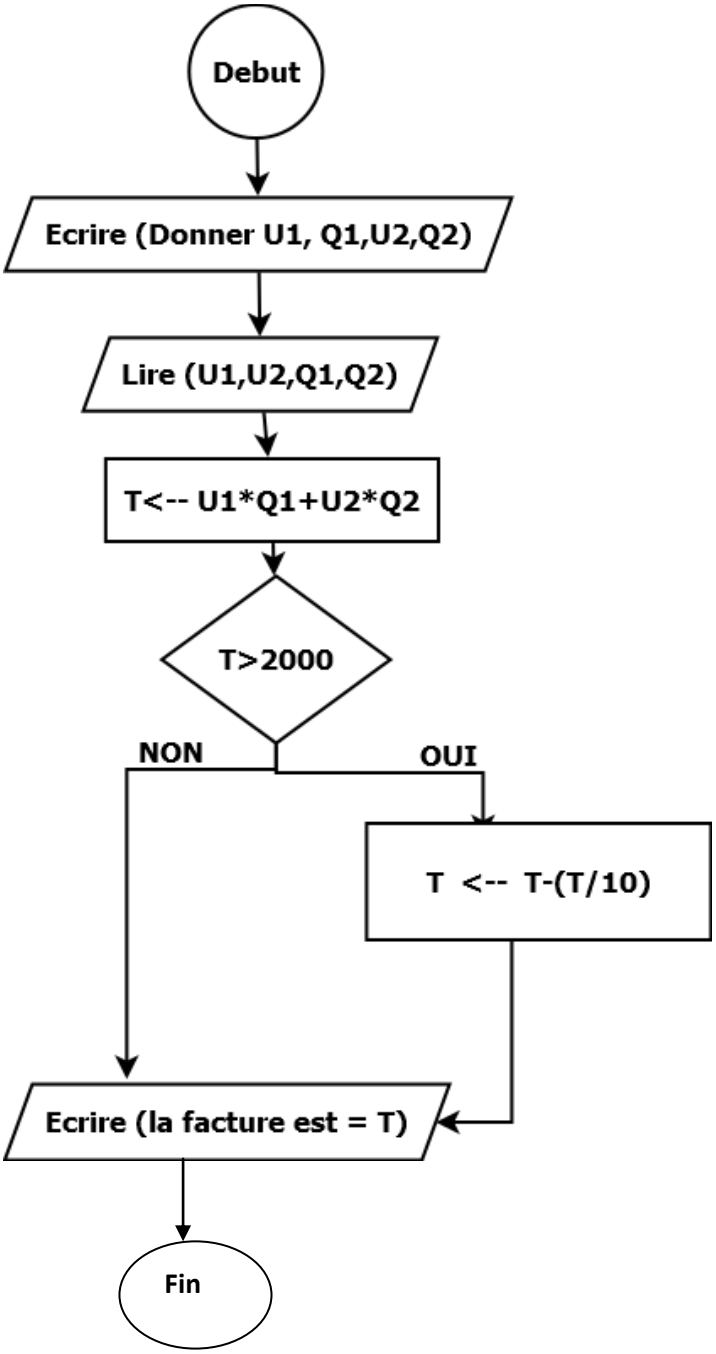
Exercice 1 :

Un commerçant veut établir la facture d'un client ayant acheté deux produits P1 et P2 ont les quantités et les prix unitaires respectifs sont Q1, U1, Q2, U2.

1. Donner l'organigramme correspondant sachant que le commerçant effectue une remise de 10% pour toute facture qui dépasse la somme de 2000 DA.
2. Donner l'algorithme correspondant



```
Algorithme Facture_Client;
Var U1, U2, Q1, Q2, R,T: Réel;
Debut
     Ecrire ('Donner U1 U2 Q1 et Q2');
     Lire (U1, U2, Q1, Q2);
     T ← U1* Q1 + U2 *Q2;
     Si T>2000 Alors R ← T/10;
                     T ← T-R;      /* Commenter mon algorithme */
     Fsi;
     Ecrire(' La facture est = ',T);
 Fin.
```

Exercice 2 :

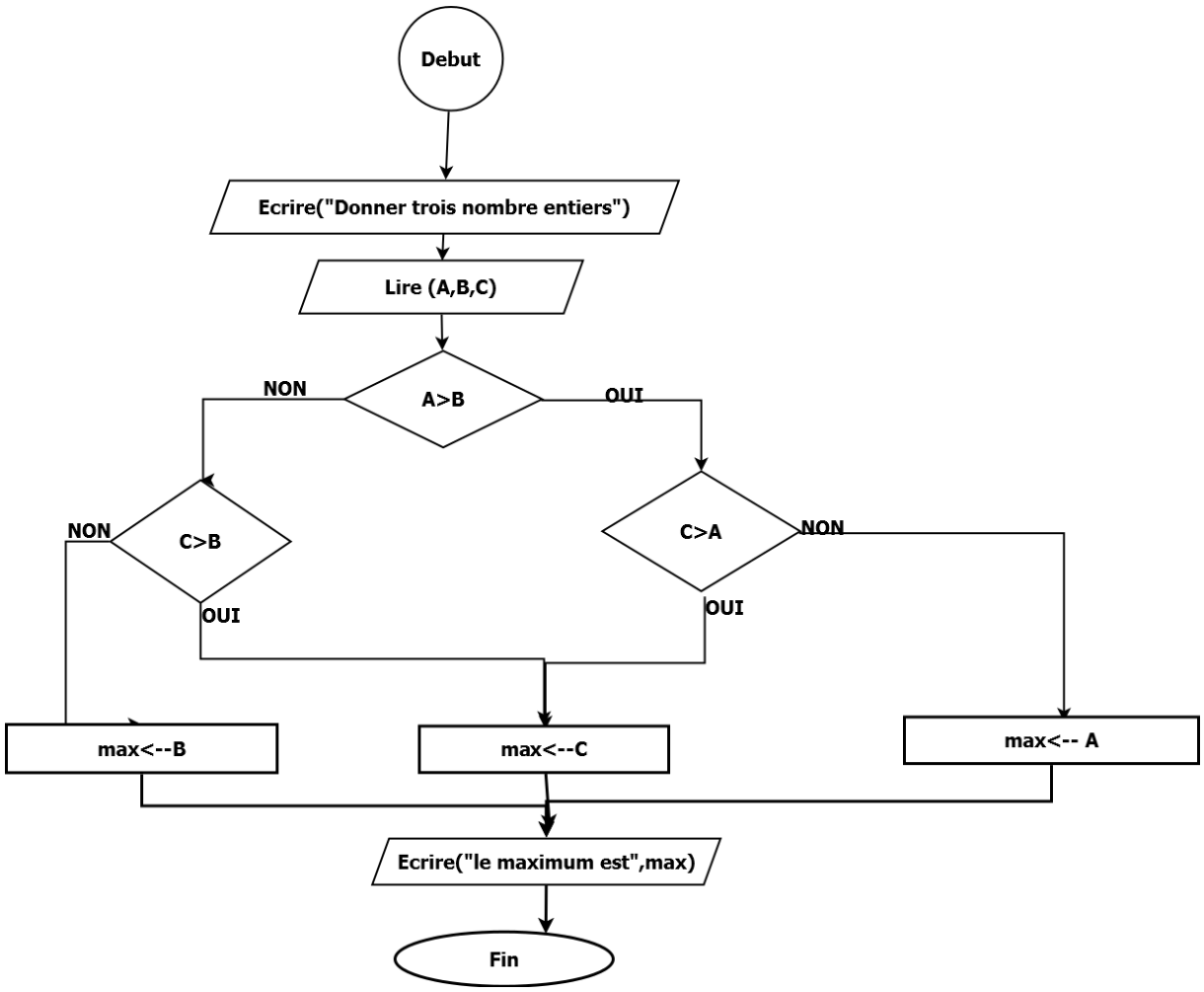
Dresser un organigramme puis donner l'algorithme permettant de résoudre chacun des problèmes suivants :

1- Détermination du Max de trois nombres entiers (A , B , C).

2- L'ordonnancement de trois nombres entiers (X , Y , Z) dans l'ordre décroissant.

3- La recherche des racines d'un trinôme de la forme $A\,x^2 + B\,x + C = 0$ dans l'espace des nombres réels.

1- <u>Organigramme du Max de trois nombres entiers (A , B , C):</u>



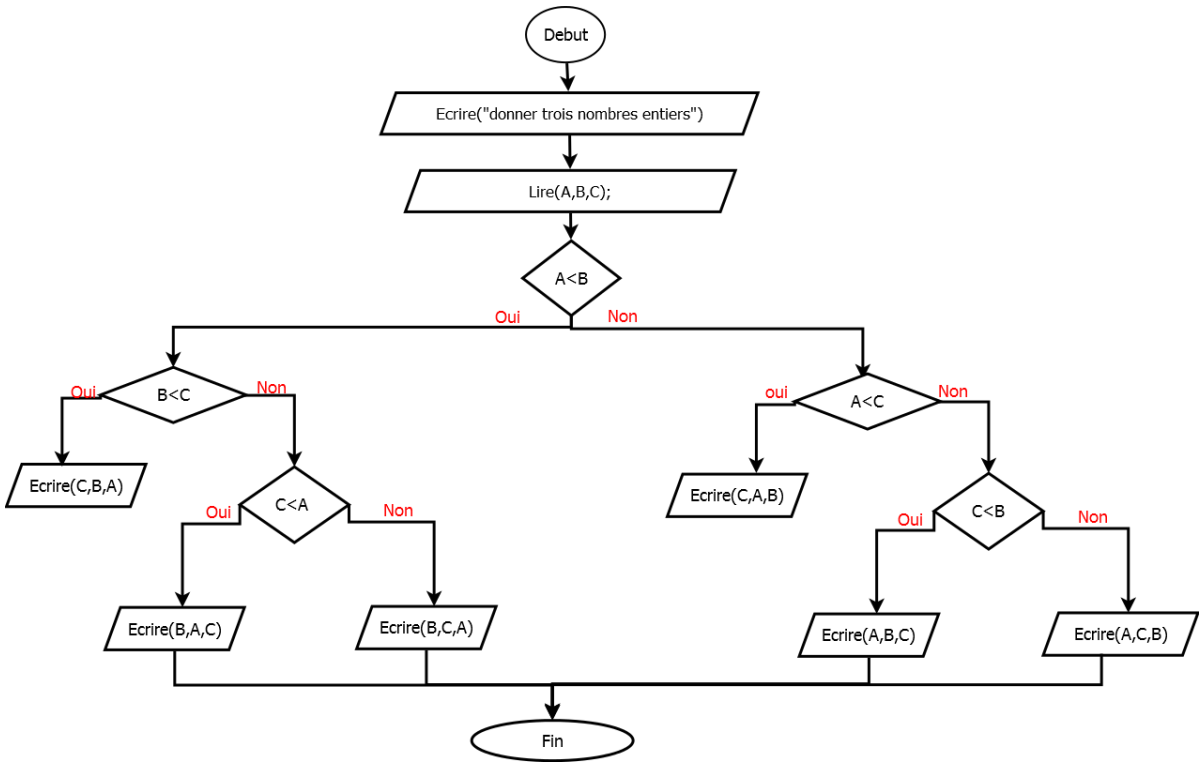

```
1-Algorithme Maximum_Trois_Entier;
Var A,B,C, Max: entier;
Debut
Ecrire('donner trois nombres entiers');
lire(A,B,C);
Si  A>B  Alors
     Si     C >A      Alors     Max⟵ C;
     Sinon    Max ⟵  A;
     Fsi;
Sinon
     Si   C>B     Alors Max ⟵ C;
     Sinon Max⟵ B;
     Fsi;
Fsi;
Ecrire (' le maximum de',A,B,C, 'est', Max);
Fin.
```

```
donner trois nombres entiers
5
6
0

Max=6
le maximum de 5 6 0
est 6
```

2- L'ordonnancement de trois nombres entiers (X , Y , Z) dans l'ordre décroissant.

Ordre décroissant veut dire du plus grand au plus petit



Ordre décroissant veut dire du plus grand au plus petit

1-Algorithme Ordonner;

```
Var A,B,C: entier;
Debut
Ecrire("donner trois nombres entiers");
lire(A,B,C);
```

B C A

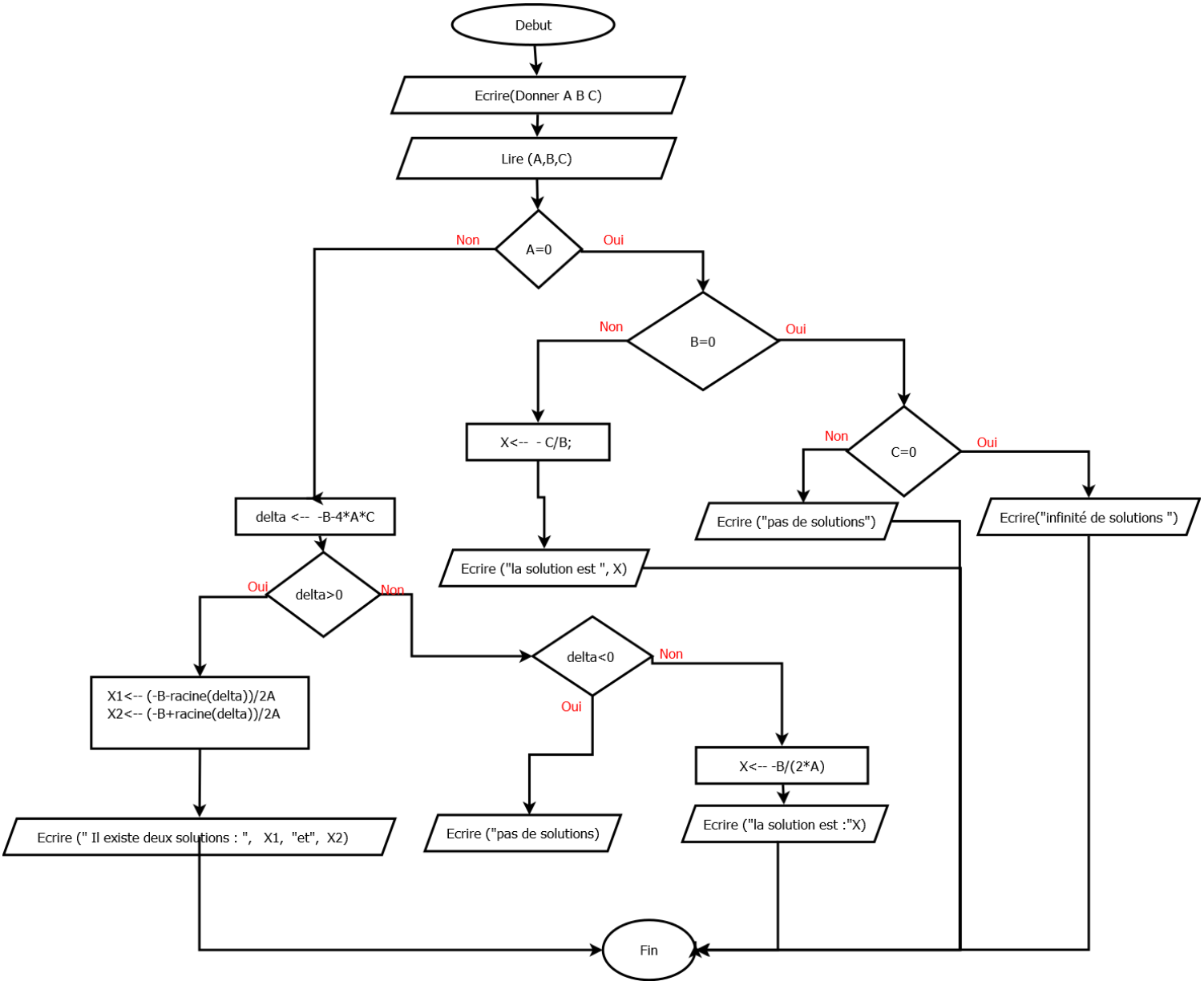
```
Si  A<B Alors
     Si    B<C    Alors    Ecrire (C,B,A);
     Sinon Si    C<A  Alors    Ecrire (B,A,C);
           Sinon   Ecrire  (B,C,A);
           Fsi;
     Fsi;
Sinon
     Si    A<C      Alors    Ecrire (C,A,B);
     Sinon   Si    C<B     Alors    Ecrire (A,B,C);
             Sinon   Ecrire (A,C,B);
             Fsi;
     Fsi;
Fsi;
Fin.
```

```
donner trois nombres entiers
5
8
-2

8 5 -2
```

3- La recherche des racines d'un trinôme de la forme $A x^2 + B x + C = 0$ dans l'espace des nombres réels.

```
Algorithme Equation_deuxieme_degres;
Var  A,B,C: réels;
     delta, X, X1, X2: Réel;
Debut
Ecrire("donner trois nombre réels");
lire (A,B,C);
Si (A=0) Alors
      Si(B=0) Alors
           Si(C=0)Alors Ecrire("infinité de solutions ");
           Sinon Ecrire ("pas de solutions");
           Fsi;
      Sinon
          X<--  - C/B;
          Ecrire ("la solution est ", X);                    racine sqrt
                                                          puissance pow
       Fsi;
Sinon
     delta <--  B*B-4*A*C;
     Si delta>0 Alors X1<--(-B-racine(delta))/2*A;/*les parenthèses sont
                                                    importantes*/
                    X2<-- (-B+racine(delta))/2*A;
                Ecrire (" Il existe deux solutions : ",  X1, "et", X2);
     Sinon Si delta=0 Alors X<-- -B/(2*A);
                    Ecrire ("la solution est :",X);
           Sinon   Ecrire ("pas de solutions dans R");
           Fsi;
      Fsi;
Fsi;
Fin.
```



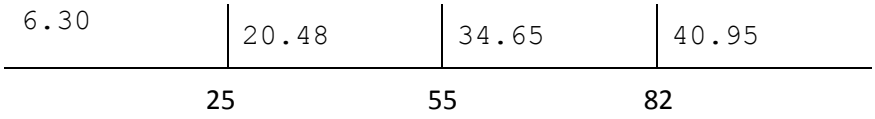
Exercice 3 :

Ecrire un algorithme qui calcule le coût de la quantité d'eau consommée par un abonné sachant que la SEAAL utilise un tarif par tranche représenté par le tableau suivant :

TRANCHE	QUANTITE (M3)	PRIX UNITAIRE (DA)
1	25	6.30
2	30	20.48
3	27	34.65
4	LE RESTE	40.95

Exemple :qauntité consommée Q= 80 M3 prix=25*6.3 + (30)*20.48 +(80-55)*34.65

```
Q=85    cout=25*6.30+5*20.48
```



```
Algorithme Cout_quantite_eau
Var
  Cout, Q: Réel;
Debut
Ecrire("donner la coût de la quantité d'eau consommée");
lire (Q);
Si Q<=25 Alors Cout<-- 6.30 * Q;
Sinon
     Si Q<= 55
     Alors Cout <--25*6.30+(Q-25)*20.48;
     Sinon Si Q<=82 Alors
               Cout<--25*6.30+ 30*20.48+(Q-55)*34.65;
          Sinon
               Cout<--25*6.30+ (55-25)*20.48+ (82-55)*34.65+(Q-
               82)* 40.95;
          Fsi;
     Fsi;

Fsi;
Ecrire ("le cout de la quantité ",Q,"consommée est=" , cout);
Fin.
```

Exercice 4 :

Ecrire un algorithme permettant d'afficher la saison en introduisant le numéro du mois.

```
Algorithme Saison;
l'expression Cas vaut:
Cas x Vaut
          val-1: inst-1;
          val-2 : inst-2;
          .....
          Sinon: inst-N;
     FinCas ;
Algorithme Saison;
Var M :entier ;
Début
     Ecrire("Donner un numéro de mois entre 1 et 12");
     Lire(M) ;
     Cas M Vaut
          12,1,2 : Ecrire("La saison est : HIVER" );
          3,4,5 : Ecrire('La saison est : PRINTEMPS");
          6,7,8 : Ecrire("La saison est : ETE");
```

```
            9,10,11 : Ecrire("La saison est : AUTOMNE");
            Sinon : Ecrire ("il faut introduire un numéro de mois entre 1 et 12");
      FinCas ;
Fin.
```

Exercice 5 :

1- Calcul du nombre de valeurs positives, du nombre de valeurs négatives, la somme de valeurs négatives, la somme de valeurs positives et le nombre des valeurs nulles d'une suite de N entiers.

1- Avec la boucle pour:

```
Algorithme NB_Positives_Negatives_Nulles ;
Var  N,I, X, NBpos, NBneg, NBnul, SOMpos, Somneg :entier ;
Debut
      Ecrire("Donner un entier N") ;
      Lire(N) ;
      NBpos←0; NBneg←0; NBnul←0; SOMpos←0; Somneg<--0;/*Initialisation*/

pour I<--1 à N
Faire
      Lire(X);
      Si X>0 Alors NBpos← NBpos+1;
                   SOMpos← SOMpos+X;
      Sinon Si X<0 Alors NBneg← NBneg+1;
                         SOMneg← SOMneg+X;
            Sinon NBnul ← NBnul+1;
            Fsi;
      Fsi;
Fait ;
Ecrire("Le nombre de valeur positives est:", NBpos, "Le nombre de valeur
  négatives est:", NBneg, "Le nombre de valeur nulles est:", NBnul,"La
  somme des valeurs positives est", SOMpos, ,"La somme des valeurs
  négatives est", SOMneg) ;
Fin.
```

Avec la boucle Tant que:

```
Algorithme NB_Positives_Negatives_Nulles ;
Var  N,I, X, NBpos, NBneg, NBnul, SOMpos, Somneg :entier ;
Debut
      Ecrire("Donner un entier N") ; Lire(N) ;
      NBpos←0; NBneg←0; NBnul←0; SOMpos←0; Somneg<--0;
I<--1;    /*initialisation de la condition*/
Tantque I<=N
Faire
      Lire(X);
      Si X>0 Alors NBpos← NBpos+1;
                   SOMpos← SOMpos+X;
      Sinon Si X<0 Alors NBneg← NBneg+1;
                         SOMneg← SOMneg+X;
            Sinon NBnul ← NBnul+1;
            Fsi;
      Fsi;
I<--I+1; /*Incrémentation de I, donc changement de la condition à un
               certain moment*/
Fait ;
```

```
Ecrire("Le nombre de valeur positives est:", NBpos, "Le nombre de valeur
  négatives est:", NBneg, "Le nombre de valeur nulles est:", NBnul,"La somme
  des valeurs positives est", SOMpos, ,"La somme des valeurs négatives est",
  SOMneg) ;
Fin.
```

2- Affichage du nombre de lettres majuscules et celui de lettres minuscules à partir d'une séquence de caractères se terminant par le caractère '#'.

On peut pas utiliser la boucle pour car on ne sait pas combien de fois la boucle va se répéter

Dans un système informatique, à chaque caractère est associé une valeur numérique : son **code ASCII** (*American Standard Code for Information Interchange*). L'ensemble des codes est recensé dans une table nommée "***table des codes ASCII***". Quand on stocke un caractère en mémoire (*dans une variable*), on mémorise en réalité son code ASCII. Un code ASCII est codé sur un octet (*huit bits*).

```
Algorithme NBmajuscule_NBminuscule;
Var
     c: caractère;
     nbmin, nbmaj: entier;
Debut
nbmin←0; nbmaj←0;
lire(c);
Tant que c<>'#'
faire
       Si c>='a' et c<='z' Alors nbmin← nbmin+1;
        Sinon Si c>='A' et c<='Z' Alors nbmaj← nbmaj+1; Fsi;
        Fsi;
        lire(c);
Fait;
Ecrire ("le nombre de lettre majuscules=", nbmaj,"le nombre de
ettres          minuscules=",nbmin);
Fin.
```

Autre Solution correcte mais on fait le test même si c='#' :

```
Algorithme NBmajuscule_NBminuscule;
Var
     c: caractère;
     nbmin, nbmaj: entier;
Debut
nbmin←0; nbmaj←0;
c<--' '; /*initialiser c à une valeur autre que '#'*/
Tant que c<>'#'
     Faire
           lire(c); /*changement de la condition*/
           Si c>= 'a' et c<='z' Alors nbmin← nbmin+1;
           sinon Si c>='A' et c<='Z' Alors nbmaj←nbmaj+1;Fsi;
           Fsi;
     Fait;
```

```
 Ecrire ("le nombre de lettre majuscules=", nbmaj,"le nombre de
ettres minuscules=",nbmin);
 Fin.
```

3- Recherche du minimum et du maximum dans un ensemble de N nombres réels.

```
Algorithme MaxMin;
Var I,N :entier ;
Max, Min, X: Réel;
Début
ecrire ('donner N'); lire(N);
Si N>0  Alors
     /* Lire le premier élément, puis initialiser le Min et le Max à cette valeur*/
Lire(X) ; Max←X ; Min←X ;
Pour I ←1 à N-1   /* la boucle se répète N-1 fois*/
Faire
          /* lire la suite des éléments et mettre à jour le Min et le Max*/
     Lire(X) ;
     Si Max<X Alors Max←X; Fsi;
     Si Min>X Alors Min←X; Fsi;
Fait ;
Ecrire("Le Minimun des valeurs est: ",Min," le Maximum est : ",Max) ;
Sinon écrire ("la valeur de N n'est pas valide");
Fsi;
Fin.
```

4- Calcul du quotient et reste de la division de deux entiers A et B sans utiliser l'opération de division

```
Algorithme QuotReste ;
Var A,B,Q,R :entier ;
Début
     Répéter
          Ecrire("Donner deux entiers A et B") ;
          Lire(A,B) ;
     jusqu'à A>0 et B>0 et A>=B;

     Q ←0 ; R ←A ;
     Tantque R>=B
          Faire
          Q ←Q+1;
          R ←R-B;
     Fait ;
     Ecrire("Le Quotient de A/B est : ",Q, " Le reste de A/Best : ",R) ;
Fin.
```

5- Vérification si un entier positif X est premier ou non

```
Algorithme Premier;
Var X,M,I :entier ;
Pr :booléen ;
Début
 Ecrire("Donner un entier X") ;
```

```
 Lire(X) ;
/*X est premier s'il a deux diviseurs distincts 1 et lui-même, mais
le 1 non */
Pr←Vrai;
Si X=1
     Alors Pr←Faux
     Sinon M←X DIV 2;
     I ←2;
     Tantque I <= M et Pr
     Faire /*si on trouve un diviseur on arrête la boucle
     Si X MOD I=0 Alors Pr←Faux Fsi;
     I ←I+1 ;
     Fait;
Fsi ;
Si  Pr  Alors  Ecrire(X,'"est  premier")  Sinon  Ecrire(X,"n'est  pas
premier");
Fsi;
Fin.
```

6- Calcul de la somme des nombres parfaits compris entre 5 et N. (*Un nombre entier positif A est parfait s'il est égal à la somme de ses diviseurs <A*).

```
Algorithme SommeParfait;
Var N,I,J, SomParfait, SomDiv :entier ;

Début
Répéter
      Ecrire("Donner un entier N") ;
      Lire(N) ;
jusqu'à N>5;
SomParfait ← 0;
Pour I← 5 à N
Faire
     SomDiv←1;// Le 1 est un diviseur
     Pour J←2 à I DIV 2
     Faire  si I MOD J=0 Alors SomDiv←SomDiv+J; Fsi;
     Fait;
Si SomDiv = I Alors SomParfait←SomParfait+I; Fsi;
Fait;
Ecrire('la  somme  des  nombres  parfaits  entre  5  et',  N,  'est',
omParfait);
Fin.
```

7- L'affichage de l'alphabet complet ('A' à 'Z') ou ('a' à 'z').

```
Algorithme Affichage_Alphabet;
Var c: caractere;
    X: entier;
Debut
Ecrire('Si vous voulez afficher l'alphabet en minuscule taper 1 et
si vous voulez afficher l'alphabet en majuscule taper 2');
```

```
Lire(X);
Cas X vaut:
1:
   Pour c← 'a' jusqu'à 'z'
        Faire ecrire (c);
   Fait;
2:
   Pour c← 'A' jusqu'à 'Z'
       Faire ecrire (c);
   Fait;
Sinon Ecrire ('votre choix est invalide');
FinCas;
Fin.
```

Calcul de la somme des chiffres d'un entier positif.

```
Algorithme SommeChiff;
Var N,S,Q :entier ;
Début
Répéter
Ecrire('Donner un entier naturel N') ;
Lire(N) ; Jusqu'à N≥0 ;
S←0 ; Q←N ;
Tantque Q>0
Faire
      S←S+Q MOD 10;
      Q← Q DIV 10;
Fait ;
Ecrire('La somme des chiffres qui composent ',N,' est :',S) ;
Fin.
```

Exercice 6 :

Ecrire l'algorithme permettant de déterminer le PGCD de deux nombres entiers A et B en utilisant les méthodes suivantes :

Méthode 1 : soustractions successives.
PGCD (3465 , 1575)

A		B		Reste
3465	–	1575	=	1890
1890	-	1575	=	315
1575	–	**315**	=	1260
1260	–	315	=	945
945	–	315	=	630
630	–	315	=	315
315	–	315	=	0

Méthode 2 : division euclidienne.
PGCD (7038 , 5474)

A		B	Reste
7038	/	5474	**1564**
5474	/	**1564**	**782**
1564	/	**782**	0
3465	/	1575	**315**
1575	/	**315**	0

```
Algorithme PGCDSoustraction;
Var A,B,X,Y,R: Entier;
Debut
Répéter Ecrire('Donner deux entiers');Lire(A,B);
jusqu'à A>0 et B>0;
Si B>A Alors
    /*Permutation entre A et B*/
       X←A;
       A←B;
       B←X;
Fsi;
X←A;  Y←B;     /*sauvegarder A et B*/
R← A-B;
Tantque R<>0
Faire
     Si R>=B Alors
            A←R;
      Sinon A←B;
            B←R;
      Fsi;
R←A-B;
Fait
Ecrire('le PGCD de ',X, 'et',Y, 'est', B); /*On peut afficher A ou B*/
Fin
```

```
Algorithme PGCDDivision;
Var A,B,X,Y,R: Entier;
Debut
Répéter Ecrire('Donner deux entiers');Lire(A,B);
jusqu'à A>0 et B>0;
Si B>A Alors
    /*Permutation entre A et B*/
```

```
        X←A;
        A←B;
        B←X;
Fsi;
X←A;   Y←B;
R← A MOD B;
Tantque R<>0
Faire
      A←B;
      B←R;
      R← A MOD B;
Fait
Ecrire('le PGCD de ',X, 'et',Y, 'est', B); /*On affiche le B comme PGCD*/
Fin
```

Exercice 7 :

Ecrire un algorithme qui détermine et affiche la Nème valeur de la suite (U_N) sachant que :

$U_0 = 0$; $U_1 = 1$; $U_2 = 2$; $U_N = U_{N-1} + U_{N-3}$ pour $N > 2$.

```
Algorithme suite;
Var X,Y,Z,Res,I,N:entier;
Debut
   Ecrire('Donner un entier') ;
Repeter Lire(N) ; Jusqu'à N≥0 ;
X←0 ;
Y←1 ;
Z←2 ;
Pour I←3 à N
      Faire
      Res←Z+X ;
      X←Y ; Y←Z ; Z←Res ;
Fait ;
Cas N Vaut
     0 : Res ←X ;
     1 : Res ←Y ;
     2 : Res←Z ;
Fincas ;
Ecrire("Le terme", N ,"de la suite est: ",Res) ;
Fin.
```