Travaux Dirigés 2

Notes de cours :

- Les trois instructions répétitives sont :
  - Tant que <condition> faire <action> ftq ;
  - Pour <compteur> ← <valInitiale> a <valFinale> faire <action> fpour ;
  - Repeter <action> Jusqu'a <condition> ;
- Les délimiteurs de bloc répétitif ftq, fpour sont obligatoires même s'il n'y a qu'une seule instruction.
- <compteur> est une variable entière. Sa modification est interdite dans le bloc répétitif. <valFinale> est une expression dont l'évaluation donne un entier ou un réel.
- On suppose qu'il n'y a pas d'opérateurs (ni de fonctions) prédéfinis pour le modulo, la division entière ou la puissance ;

Exercice 1 :

Algorithme Lisible ; Var A,B,S,I : entier ; Début Lire(A,B) ; S ←A ; I ←1 ; Tantque I <= B Faire S ←S+I ; I ←I+1 ; ftq ; Écrire(S) ; Fin.

1. L'algorithme Lisible est il facilement lisible ?
2. Modifier la présentation de Lisible pour qu'il soit plus compréhensible.
3. L'utilisation de Tant que au lieu de pour et repeter dans ce traitement est elle judicieuse ? Justifier.

Exercice 2 :

Pour réaliser la somme de N nombres lus, un débutant a proposé la solution suivante :

```
Algorithme SommeNbr ;
Var      N,nbr,S,I : entier ;
Début
    ecrire('donner le nombre d elements');
    Lire(N) ;
    Pour I ←1 a N Faire
      ecrire('donner vos elements:');
      Lire(nbr);
    fpour ;
    Pour I ←1 a N Faire
      S ←0 ;
      S ← S+nbr;
      Ecrire('La somme de vos elements est :',S) ;
    fpour ;
Fin.
```

1. Dérouler cet algorithme pour les valeurs N=4 et les éléments : 10, 15, 22, 8 ; puis déduire que sa solution est fausse.
2. A votre avis quel raisonnement erroné a conduit ce débutant a commettre ces erreurs ?

Exercice 3 :

Écrire un algorithme qui détermine la somme des N premiers nombres entiers positifs. Exemple : si N=4 alors on cherche la somme 1+2+3+4=10.

Exercice 4 :

Ecrire un algorithme qui détermine la puissance $n^{ième}$ d'un (n entier positif ou nul) nombre a donné(i.e. : a^n).

Exercice 5 :

Ecrire un algorithme qui détermine le reste de la division d'un nombre entier A par un nombre entier B.

Exercice 6 :

Écrire un algorithme qui vérifie si un nombre donné est divisible par 2.

Exercice 7 :

Écrire un algorithme qui détermine le minimum d'une suite de N nombres entiers.

Exercice 8 :

Un nombre est dit premier s'il n'accepte comme diviseur que 1 et lui même.
Écrire un algorithme qui vérifie si un nombre est premier.

Exercice 9 :

Écrire un algorithme qui détermine le PGCD de deux nombres.

Exercice 10 :

Ecrire un algorithme qui permet d'évaluer la série S_n suivante pour n donné :

$$S_n = 1 - \frac{1}{3} + \frac{1}{5 \cdot 2} + \cdots + \frac{(-1)^n}{(2n+1) \cdot n!}$$

Exercice 11 :

Construire un algorithme permettant, pour un nombre N à quatre chiffres, de permuter son premier chiffre avec le dernier et le deuxième avec le troisième.

Exercice 12 :

Soient les algorithmes suivants :

```
Algorithme version1 ;
Var      N,M,I,J,comp: entier ;
Début
  Lire(N,M) ;
  comp←1;
  I←1 ;
  Tantque I <= N Faire
        J←1 ;
        Tantque J <= M Faire
              ecrire(I,J,comp);
              comp ← comp+1 ;
              J←J+1 ;
```

```
        ftq ;
        I←I+1 ;
  ftq ;
Fin.
Algorithme version2 ;
Var     N,M,I,J,comp: entier ;
Début
  Lire(N,M) ;comp←1;
  I←1 ;
  J←1 ;
  Tantque (I <= N) et (J <= M) Faire
        ecrire(I,J,comp);
        comp ← comp+1 ;
        J←J+1 ;
        I←I+1 ;
  ftq ;
Fin.
```

1. Dérouler les algorithmes précédents pour les valeurs N=3 et M=4.

Est-ce-qu'il y a une différence entre ce qu'ils affichent ? Justifier.

Exercices supplémentaires

Exercice 13 :

```
Algorithme Inconnu ;
Var     A,B,S,I    : entier ;
Début
    Lire(A,B) ;
    S ←A ;
    I ←1 ;
    Tantque  I ≤ B Faire
      S ←S+I ;
      I ←I+1 ;
    ftq ;
    Écrire(S) ;
Fin.
```

1. Dérouler cet algorithme avec les valeurs A=2 et B=4, et A=4 et B = 2 ?
2. En déduire ce que fait cet algorithme.

Exercice 14 :

```
Algorithme Bizarre1 ;
Var     A,S, Compteur  : entier ;
Début
    Lire(A) ;
    S ←0 ;
    Compteur ←1 ;
    Tantque Compteur <= A Faire
        S ←S+ Compteur ;
        Ecrire(Compteur) ;
    ftq ;
    Écrire(S) ;
Fin.
Algorithme Bizarre2 ;
Var     A,S, Compteur : entier ;
Début
  Lire(A) ;
  S ←0 ;
  Tantque Compteur <= A Faire
      S ←S+ Compteur ;
      Compteur ← Compteur +1 ;
      Ecrire(Compteur) ;
  ftq ;
  Écrire(S) ;
Fin.
```

- Dérouler les deux algorithmes ci-dessus. Que peut-on conclure ?

Exercice 15 :

Ecrire un algorithme qui détermine la puissance $n^{ième}$ d'un (n entier quelconque) nombre a donné(i.e. : a^n).

Exercice 16 :

Ecrire un algorithme qui détermine le quotient de la division d'un nombre entier A par un nombre entier B.

Exercice 17 :

Un nombre est dit parfait s'il est égal à la somme de ses diviseurs en omettant le nombre lui même.
Exemple : six est un nombre parfait car : 6 = 1+2+3.
Ecrire un algorithme qui vérifie si un nombre est parfait.

Exercice 18 :

Écrire un algorithme qui permet d'évaluer un polynôme P(x) d'ordre n pour une valeur donnée de x.

Exercice 19 :

Soit N un entier de cinq chiffres. Ecrire un algorithme qui permet d'afficher le chiffre du milieu et la somme des autres chiffres.

Exercice 20 :

Soit N un nombre entier composé de trois chiffres. Ecrire un algorithme qui permet de calculer la somme des cubes de ses chiffres.

Exercice 21 :

Déterminer la racine carré d'un nombre A revient à trouver un carré dont l'aire est A. En prenant un rectangle de côté arbitraire a_0 et de même aire, il est nécessaire que la longueur de l'autre côté soit A/a_0 . Or, en général, ce rectangle n'est pas carré. Pour le rendre plus carré, il suffit de prendre un rectangle dont la longueur est la moyenne arithmétique des deux côtés précédents (c-a-d: $a_1=(a_0+A/a_0)/2$) et dont l'aire est toujours A.
En substituant a_1 avec a_0, on peut répéter ce processus jusqu'à ce que $(a_1)^2$ soit très proche de A.
Écrire un algorithme qui détermine la racine carrée d'un nombre avec les deux premiers chiffres décimaux exacts.