heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Formation en emploi

Algorithmique

Recueil d'exercices

Auteurs : L. Larghi – B. Collet

Dernière mise à jour : septembre 2012

Remarques

- Les exercices sont subdivisés en séries
- Pour chaque série est spécifiée la matière couverte
- Chaque exercice est caractérisé par un degré de difficulté.
    - [f] facile
    - [m] moyen
    - [d] difficile

Table des matières

Série 1 : introduction

Matière couverte

Introduction
L'exécutant

Exercices

Exercice 1.1 [f]

Ecrire l'algorithme décrivant la marche à suivre permettant de calculer la somme entre 2 nombres.

Exercice 1.2 [m]

Ecrire l'algorithme décrivant la marche à suivre permettant le remplacement d'une ampoule grillée.

Exercice 1.3 [m]

Ecrire l'algorithme décrivant la marche à suivre permettant le remplacement d'un pneu crevé.

Exercice 1.4 [m]

Le robot trieur de pièces : ce robot doit remplir un tas constitué de pièces de 1, 2 et 5 francs, dans trois récipients destinés à recevoir les pièces des diverses valeurs. On demande d'écrire la marche à suivre pour ce robot.

Le robot comprend 4 opérations primitives :
- Prends : le robot saisit une pièce du tas.
- Dépose 1 : le robot dépose la pièce qu'il tient dans le récipient destiné à recevoir les pièces de 1 franc.
- Dépose 2 : le robot dépose la pièce qu'il tient dans le récipient destiné à recevoir les pièces de 2 francs.
- Dépose 5 : le robot dépose la pièce qu'il tient dans le récipient destiné à recevoir les pièces de 5 francs.

Conditions qui peuvent être utilisées :
- Le tas de pièces est vide.
- La pièce tenue en main est une pièce de 2 francs.
- La pièce tenue en main est une pièce de 5 francs.

et les contraires de ces conditions qu'on peut aussi lier par ET ou OU.

Remarques :
- Les récipients où ranger les pièces sont de taille suffisante pour recevoir n'importe quel nombre de pièces.
- Il peut, certains jours, n'y avoir aucune pièce à trier.
- Le robot ne peut tenir plusieurs pièces à la fois.

Exercice 1.5 [d]

Le robot transporteur de sable : ce robot doit transporter un tas de sable situé en A pour le déposer en B. Il dispose pour cela d'une brouette qu'il peut charger à l'aide d'une pelle. On demande d'écrire la marche à suivre pour ce robot.

Le robot comprend 4 opérations primitives :
- Mets une pelletée dans la brouette : le robot prend une pelletée de sable dans le tas A et la dépose dans la brouette.
- Transporte : le robot transporte la brouette de A en B.
- Vide : le robot déverse la brouette où elle se trouve.
- Ramène : le robot ramène la brouette de B en A.

Conditions qui peuvent être utilisées :
- Il y a encore du sable à transporter en A.
- La brouette est pleine.

et leur contraire.

Remarques :
- Il y a toujours du sable à transporter quand le robot commence.
- La brouette est toujours vide à côté du tas en A quand le robot commence.
- Le contenu de la brouette correspond à plusieurs pelletées de sable.
- Le travail terminé, la brouette peut rester n'importe où.

Série 2 : notions de base

Matière couverte

Déclaration des variables
Affectation
Sortie, Entrée

Exercices

Exercice 2.1 [f]

On demande d'exécuter les algorithmes (partiels) suivants et de déterminer les valeurs qui seront affichées à l'écran.

a)

```
Var a, b, c : entier
a ← 10
b ← 5
c ← 7
a ← b
b ← c
c ← a
Ecrire(a, b, c)
```

b)

```
Var a, b : réel
a ← 10
a ← a + 1
a ← a / 2
b ← a * 2
b ← b + a
Ecrire(a, b)
```

c)

```
Var a : entier
Lire (a)
a ← 7
Ecrire (a)
```

d)

```
Var a : entier
a ← 10
Lire (a)
Ecrire (a)
```

e)

```
Var a : entier
Lire (a)
a ← 8
Lire (a)
Ecrire (a)
```

f)

```
Var a, b, c : entier
Lire (a)        /* on lit 10 */
b ← 5
a ← a + b
b ← a * 2
Ecrire (a, b)
```

g)

```
Var a, b, c : entier
Var x,y : Réel
Lire (a)        /* on lit 10 */
b ← 8
```

```
x ← a
y ← b
Ecrire (a / b, a DIV b, a MOD b, a DIV b * b)      Ecrire (x / y, a + x / b, -
b * a / y)
```

Exercice 2.2 [m]

Ecrire un algorithme qui calcule et affiche la surface d'un triangle. Sa base et sa hauteur seront saisies au clavier.

Exercice 2.3 [m]

Ecrire un algorithme qui donne la circonférence d'un cercle dont le rayon est saisi au clavier.

Exercice 2.4 [m]

Ecrire un algorithme qui, au départ du prix hors TVA d'un article, donne pour résultat le prix de cet article TVA de 5% incluse.

Exercice 2.5 [m]

Etant donné le revenu imposable d'un contribuable, ce revenu étant compris entre 100'000 Frs et 150'000 Frs, on demande de concevoir un algorithme qui calcule l'impôt à payer par ce contribuable sachant que, dans la tranche de revenus entre 100'000 Frs et 150'000 Frs, l'impôt à payer s'élève à 15'000 Frs plus le 15% de la part des revenus qui excède 100'000 Frs.

Exercice 2.6 [m]

Etant donnés un montant exprimé en Frs et le taux de change du dollar US exprimé en Frs par dollar US, on demande de concevoir un algorithme qui convertit ce montant en dollars US et affiche le résultat ainsi obtenu

Exercice 2.7 [m]

Un concessionnaire automobile emploie un vendeur qu'il rémunère de la façon suivante : 5'000 Frs par mois plus 1'000 Frs par voiture vendue plus 5% du montant total des ventes mensuelles d'automobiles. Il vous demande votre aide pour concevoir un algorithme qui, à partir du nombre de voitures vendues au cours d'un mois par le vendeur et du montant total des ventes mensuelles, calcule et affiche le salaire mensuel du vendeur.

Exercice 2.8 [m]

Etant donnés les indices des prix à la consommation des mois de janvier et de février, on demande d'écrire un algorithme qui calcule et affiche le taux d'inflation entre ces deux mois.

Exercice 2.9 [m]

Etant donnés la composition, exprimée par le nombre d'enfants et le nombre d'adultes, d'une famille qui se présente à l'entrée d'un parc d'attraction et le nombre de tickets gratuits pour enfants que cette famille a découpés dans la presse locale, on demande de construire un algorithme qui détermine le montant épargné par cette famille grâce à ses tickets gratuits ainsi que le montant total à payer par la famille pour entrer dans ce parc. Le prix normal d'une entrée à ce parc est de 25 Frs par adulte et 15 Frs par enfant.

Exercice 2.10 [m]

Etant donné le montant d'un capital placé dans un compte d'épargne et le taux d'intérêt en pour-cent pratiqué par la caisse où ce compte d'épargne est placé, on demande de concevoir un algorithme qui calcule et affiche l'intérêt obtenu et la valeur acquise par ce capital au bout de la période indiquée par l'utilisateur.

Exercice 2.11 [m]

Etant donné le temps exprimé sous la forme heures, minutes et secondes, on demande d'écrire un algorithme qui calcule et affiche ce temps exprimé en secondes.

Exercice 2.12 [m]

L'organisateur d'une course automobile sur circuit fermé souhaite tenir les spectateurs régulièrement informés des performances réalisées par les divers concurrents, lors des séances d'essais. A cet effet, il pose le problème suivant. Etant donné la longueur du circuit (en km) et le temps mis par un concurrent pour effectuer un tour de circuit (ce temps est exprimé en minutes, secondes et millièmes de secondes), on demande d'écrire un algorithme qui calcule et affiche la vitesse horaire moyenne exprimée en km/h, à laquelle ce concurrent vient de parcourir un tour de circuit, sachant que cette vitesse est calculée selon la formule vitesse (km/h) = distance (km) / temps (h).

Exercice 2.13 [d]

On demande d'écrire les opérations d'affectation nécessaires pour permuter la valeur d'une variable X et celle d'une variable Y. (Seulement les instructions, on peut se passer des en-têtes.)

Exercice 2.14 [d]

On demande d'écrire les opérations d'affectation nécessaires pour permuter de façon circulaire les valeurs des variables X, Y, Z de la manière indiquée ci-dessous : (Seulement les instructions, on peut se passer des en-têtes.)

```
x -> y, y -> z, z -> x
```

Exercice 2.15 [d]

Etant donné le temps exprimé en secondes, on demande de concevoir un algorithme qui calcule et affiche ce temps en heures, minutes et secondes.

Série 3 : conditions et choix multiples

Matière couverte

Déclaration des variables
Affectation
Sortie, Entrée
Conditions et choix multiples

Exercices

Exercice 3.1 [f]

On demande d'exécuter les algorithmes suivants et de déterminer les valeurs qui seront affichées à l'écran.

a)

```
Var a : entier
a ← 10
Si a = 7
Alors
  a ← 0
FinSi
Ecrire (a)
```

b)

```
Var a : entier
a ← 7
Si a = 7
Alors
  a ← 0
FinSi
Ecrire (a)
```

c)

```
Var a : entier
Lire (a)        /* (a prendra la valeur 12) */
Si a = 7
Alors
  a ← 0
Sinon
  a ← 1
FinSi
Ecrire (a)
```

d)

```
Var a,x : entier
Lire (x)        /* (x prendra la valeur 12) */
Si x > 0
Alors
  a ← -1
Sinon
  a ← 0
FinSi
Ecrire (a)
```

Exercice 3.2 [m]

Etant donné un nombre quelconque, on demande de concevoir un algorithme qui affiche un des messages [plus petit que 10], [plus grand que 10], [égal à 10] selon la valeur du nombre.

Exercice 3.3 [m]

Etant donné la taille d'une personne en cm, son poids en kg et s'il s'agit d'une femme ou d'un homme, on demande de concevoir un algorithme qui affiche le nombre de kilocalories brûlées pendant une journée sans activité (métabolisme de base) selon la formule ($poids^{0.425} \times taille^{0.725} \times 0.172 \times K$), où K vaut 35 pour une femme et 40 pour un homme.

Exercice 3.4 [m]

Etant donné une équation du 1er degré `ax+b=0` entrée par l'utilisateur au moyen de ses coefficients a et b, on demande de concevoir un algorithme qui affiche :

- *"pas de solution"* si a=0 et b≠0,
- *"équation indéterminée"* si a=0 et b=0,
- *"une solution : x="* suivi de la solution (-b/a) dans les autres cas.

Exercice 3.5 [m]

Etant donné une équation du 2e degré $ax^2+bx+c=0$ entrée par l'utilisateur au moyen de ses coefficients a, b et c, on demande de concevoir un algorithme qui affiche :

- [équation du premier degré] si a vaut 0, et sinon,
- [pas d'équation] si a et b valent 0,
- [pas de solution] s'il n'existe pas de solution (cas où $b^2 - 4ac < 0$),
- [une solution] s'il existe une unique solution (cas où $b^2 - 4ac = 0$),
- [deux solutions] dans les autres cas (cas où $b^2 - 4ac > 0$).

Exercice 3.6 [m]

Etant donné le nom d'un pays, on demande de concevoir un algorithme qui affiche la capitale de ce pays si elle est connue ou le message [pays inconnu]. Les pays connus sont l'Allemagne (Berlin), l'Autriche (Vienne), la France (Paris), l'Italie (Rome) et la Suisse (Berne).

Exercice 3.7 [m]

Connaissant le montant du salaire horaire d'un ouvrier et étant donné le nombre d'heures passées au cours d'une journée, on demande d'écrire un algorithme qui affiche :

- le nombre d'heures normales passées,
- le nombre d'heures supplémentaires passées,
- le salaire à payer.

On conviendra que les heures passées au-delà des 8 heures par jour sont considérées comme supplémentaires et payées 50% plus cher que les autres, qui, elles, sont rémunérées à 18 Frs de l'heure.

Exercice 3.8 [m]

On demande de concevoir un algorithme qui permette à un jeune élève de vérifier s'il est capable de calculer la somme de deux nombres entiers et le produit de ces deux nombres. Pour chaque opération, on indiquera à l'élève si le résultat qu'il a introduit est correct ou faux. Les nombres entiers seront tirés au hasard dans l'intervalle allant de 1 à 10 (on utilisera l'instruction *aléatoire (inf, sup)* qui génère un nombre entier aléatoire compris entre *inf* et *sup* inclus).

Exercice 3.9 [m]

Etant donné le solde du compte en banque d'un client et le montant qu'il désire retirer de ce compte, on demande de concevoir un algorithme qui affiche soit le message [retrait accepté], le montant du retrait effectivement accepté et le nouveau solde du compte, soit le message [retrait refusé], sachant que :

- un retrait est refusé si le solde du compte est inférieur ou égal à -5'000 Frs,
- si après le retrait, le solde du compte est inférieur à -5'000 Frs, on ne peut accepter qu'un retrait partiel,
- le nouveau solde du compte après retrait doit être supérieur ou égal à -5'000 Frs.

Exercice 3.10 [m]

Etant donné le montant d'une facture, on demande de concevoir un algorithme qui calcule et affiche le montant de la ristourne éventuelle accordée à un client et le montant final à payer, sachant que :

- le montant de la ristourne éventuelle à appliquer au montant de la facture est de
  10% du montant de la facture si ce montant est supérieur à 7'500 Frs,
  5% du montant de la facture si ce montant est compris entre 4'000 Frs et 7'500 Frs,
  2% du montant de la facture si ce montant est supérieur ou égal à 2'000 Frs et inférieur à 4'000 Frs,
- aucune ristourne n'est accordée pour des factures dont le montant est inférieur à 2'000 Frs,
- le montant final à payer est le montant de la facture diminué du montant de la ristourne éventuelle.

Exercice 3.11 [d]

Etant donné deux nombres a et b, on demande de concevoir un algorithme qui, sans effectuer le produit, affiche les messages [produit positif], [produit négatif], [produit nul], selon que le produit est négatif, positif ou nul.

Exercice 3.12 [d]

La régie des postes souhaite qu'on écrive, à l'intention de ses guichetiers, un algorithme qui détermine la taxe à appliquer à un colis postal. A cet effet, elle pose le problème suivant : étant donné la longueur, la largeur et la hauteur du colis exprimées en cm et le poids du colis exprimé en kg, on demande de construire un algorithme qui produise pour résultat soit la taxe à appliquer au colis, soit le message [colis hors norme].
A cet effet on conviendra que :

- si le poids du colis dépasse 10 kg, le colis est considéré comme hors norme,
- si une des dimensions du colis (longueur, largeur, hauteur) excède 50 cm, ce colis est considéré comme hors norme,
- la taxe normale appliquée à un colis normal est de 5 Frs,
- si le poids est compris entre 5 et 10 kg, une surtaxe de poids, forfaitairement fixée à 3 Frs est perçue,
- si le volume du colis excède 27 dm^3, une surtaxe d'encombrement, forfaitairement fixée à 3 Frs est perçue,
- la surtaxe due au poids et celle due à l'encombrement sont cumulatives.

Exercice 3.13 [d]

Dans le but de conserver le patrimoine architectural d'une région, il a été décidé d'octroyer des primes à la rénovation des immeubles. Ces primes sont calculées en fonction du montant des travaux effectués en appliquant les règles suivantes :

- Pour avoir droit à la prime, le montant des factures (hors TVA) ne peut être inférieur à 100'000 Frs.
- Le montant de base de la prime est fixé à 30 % du montant des factures, et ne peut dépasser 50'000 Frs.
- Ce montant de base de la prime est ensuite majoré de 20 % par enfant à charge. Ce montant ainsi majoré ne peut toutefois pas excéder les 2/3 du montant des factures prises en considération.

On demande de concevoir un algorithme qui détermine la prime octroyée à un ménage en fonction du montant des travaux effectués et du nombre d'enfants à charge.

Exercice 3.14 [d]

Etant donnée une date exprimée sous la forme jour, mois, année, le mois étant exprimé sous forme numérique, on demande de concevoir un algorithme qui affiche :

- soit le message [année incorrecte] si l'année communiquée n'est pas comprise entre 1901 et 2099,
- soit le message [mois incorrect] si le mois n'est pas compris entre 1 et 12, l'année étant correcte,
- soit le message [jour incorrect] si le jour n'étant pas compatible avec l'année et le mois donnés, l'année et le mois étant correct,
- soit le message [date correcte] si le jour, le mois et l'année sont corrects et compatibles.

On notera que, pour les années dont le millésime est compris entre 1901 et 2099, toute année dont l'expression numérique est divisible par 4 est bissextile.

Série 4 : boucles

Matière couverte

Déclaration des variables
Affectation
Sortie, Entrée
Conditions et choix multiples
Boucles

Exercices

Exercice 4.1 [f]

On demande d'exécuter les algorithmes suivants et de déterminer les valeurs qui seront affichées à l'écran ainsi que leur signification.

a)

```
Var a, n, x, i : entier
n ← 4
a ← 0
Pour i Allant de 1 à n à pas de 1 Faire
    Lire (x)
    Si x > 0
    Alors
      a ← a + 1
    FinSi
Fait
Ecrire (a)
```

b)

```
Var a, n, x, i : entier
n ← 4
a ← 0
Pour i Allant de 1 à n Faire
    Lire (x)    (x prendra les valeurs de 7, 0, 6, -2)
    Si x > 0
    Alors
      a ← a + x
    FinSi
Fait
Ecrire (a)
```

c)

```
Var a, n, k, i : entier
k ← 5
n ← 3
Pour i Allant de 1 à n à pas de 1 Faire
    a ← k * k
    Ecrire (a)
Fait
```

d)

```
Var c, n, x, m, i : entier
n ← 4
c ← 0
m ← 0
Pour i Allant de 1 à n à pas de 1 Faire
    Lire (x)    /* (x prendra les valeurs de 7, 0, 6, -2) */
    Si (x > 0) et (x <= 20)
    Alors   m ← m + x
            c ← c + 1
    FinSi
Fait
Si c > 0
Alors
  Ecrire (c)
  Ecrire (m / c)
FinSi
```

e)

```
Var a : entier
a ← 0
Tant Que a > 0 Faire
    a ← a - 1
Fait
Ecrire (a)
```

f)

```
Var x : entier
x ← 5
Tant Que x > 0 Faire
    x ← x - 2
Fait
Ecrire (x)
```

g)

```
Var s, f, x : entier
Lire (f)        /* (f prendra la valeur -1) */
Lire (x)        /* (x prendra les valeurs 7, 9, 3 et -1) */
s ← 0
Tant Que x <> f Faire
    s ← s + x
    Lire (x)
Fait
Ecrire (x)
```

h)

```
Var c, x : entier
x ← 5
c ← 0
Tant Que x < 10 Faire
    x ← x + 1
    c ← c + 1
Fait
Ecrire (x)
```

i)

```
Var a : entier
a ← 1
Tant Que a > 1 Faire
    a ← a + 1
Fait
Ecrire (a)
```

j)

```
Var n, co, x : entier
co ← 0
Lire (n)
Tant Que co < n Faire
    Lire (x)
    co ← co + 1
Fait
Ecrire (co)
```

k)

```
Var s, x, co : entier
co ← 0
s ← -1
Lire (x)
Tant Que x <> s Faire
    co ← co + 1
    Lire (x)
Fait
Ecrire (co)
```

l)

```
Var s, x, co, n : entier
s ← 0
n ← 3
Lire (x)
co ← 1
Tant Que(x <> s) et (co < n) Faire
    co ← co + 1
    Lire (x)
Fait
Ecrire (co)
```

m)

```
Var s, x, co, n : entier
s ← 0
n ← 3
Lire (x)
co ← 1
Tant Que(x <> s) ou (co < n) Faire
    co ← co + 1
    Lire (x)
Fait
Ecrire (co)
```

n)

```
Var s, a : entier
s ← 0
Répéter
    Lire (a)
    s ← s + a
Jusqu'à (s=0)
Ecrire (s)
```

o)

Récrire 1k) avec une boucle Répéter ... Jusqu'à.

p)

Récrire 1l) avec une boucle Répéter ... Jusqu'à.

q)

Récrire 1m) avec une boucle Répéter ... Jusqu'à.

Exercice 4.2 [m]

Etant donné le nombre d'élèves d'une classe et la suite des notes obtenues en anglais par les n élèves, on demande de concevoir un algorithme qui donne pour résultat le nombre et le pourcentage de notes inférieures à 4 obtenues par ces élèves.

Exercice 4.3 [m]

Etant donné le nombre d'élèves d'une classe et la suite des notes obtenues par ces élèves lors d'un contrôle de chimie, on demande d'écrire un algorithme qui calcule et affiche la moyenne de ces notes ainsi que le nombre d'élèves effectivement présents à ce contrôle, sachant que la note d'un élève absent lors du contrôle aura une valeur négative et ne devra pas être prise en compte lors du calcul de la moyenne. L'algorithme affichera pour la facilité de l'utilisateur le numéro d'ordre des élèves.

Exercice 4.4 [m]

On demande de concevoir un algorithme qui donne pour résultat le nombre de valeurs égales à zéros contenues dans une suite de n chiffres donnés (un chiffre est un nombre compris entre 0 et 9).

Exercice 4.5 [m]

Etant donné une suite de n nombres, on demande d'écrire un algorithme qui soit affiche le message [pas de valeur égale à 0] si la suite ne contient aucune valeur égale à 0, soit, s'il en trouve au moins une, en indique la quantité et la position de la dernière valeur égale à 0 rencontrée.

Exercice 4.6 [m]

Un client achète 5 articles différents dans un magasin. Les quantités de chaque article ainsi que les prix étant saisis au clavier, on demande d'écrire un algorithme qui calcule et affiche les montants correspondant à chaque article ainsi que le montant total à payer.

Exercice 4.7 [m]

Etant donné le salaire horaire et le nombre d'heures prestées chaque jour de la semaine (5 jours) par ouvrier, on demande de concevoir un algorithme qui détermine le nombre d'heures de prestations normales, le nombre d'heures supplémentaires et le salaire de la semaine de cet ouvrier, sachant que les heures supplémentaires (au-delà de 40 heures par semaine) sont payées 50% plus cher que les heures de prestations normales.

Exercice 4.8 [m]

Le contrôleur de gestion d'une chaîne de supermarchés souhaite analyser chaque semestre l'évolution du montant des ventes réalisées par rapport aux prévisions faites. Il pose donc le problème suivant. Etant donné pour chaque mois du semestre le montant des ventes réalisées et le montant des prévision faites, on demande de concevoir un algorithme qui calcule et affiche, pour chaque mois, l'écart positif ou négatif, exprimé en % des prévisions de vente, entre ventes et prévisions; en outre, cet écart sera suivi d'un « * » s'il est fortement erroné, c'est-à-dire > 10%, positivement ou négativement.

Exercice 4.9 [m]

On a compté, un samedi entre 17 et 18 h, le nombre de voitures circulant sur un tronçon de l'autoroute Aigle-Lausanne et enregistré pour chacune d'elles sa vitesse.
On vous demande de concevoir un algorithme qui, étant donnée le nombre de voitures comptées et la vitesse de passage de chacune d'elles, exprimée en km/h, calcule et affiche la vitesse moyenne des voitures à cet endroit et le pourcentage de voitures en infraction sachant que la vitesse maximale autorisée sur le tronçon est de 120 km/h.

Exercice 4.10 [m]

Le commentateur sportif d'une station de radio souhaite donner aux auditeurs, lors de l'émission sportive du dimanche soir, des statistiques concernant le week-end. A cet effet, il souhaite qu'on conçoive à son intention l'algorithme suivant.
Etant donné le nombre de rencontres effectivement disputées (ce nombre est supposé être > 0) et les résultats de chaque rencontre donnés au moyens de 2 nombres entiers : d'abord, le nombre de buts inscrits par l'équipe visitée et, ensuite, le nombre de buts inscrits par l'équipe visiteuse, on demande d'écrire un algorithme qui calcule et affiche le nombre de victoires remportées par les équipes visitées, le nombre de victoires remportées par les équipes visiteuses, le nombre de rencontres qui se sont terminées par un match nul, le nombre moyen de buts inscrits par rencontre. On notera que l'équipe visitée (visiteuse) remporte la victoire lorsque le nombre de buts qu'elle a inscrits est strictement supérieur au nombre de buts inscrits par l'équipe visiteuse (visitée). La rencontre se termine par un match nul si les deux équipes ont inscrit le même nombre de buts.

Exercice 4.11 [m]

Etant donné une suite de n nombres, on demande d'écrire un algorithme qui calcule la somme des nombres de cette suite et affiche le plus grand des nombres de cette suite.

Exercice 4.12 [m]

Etant donné une suite de nombres (différents de 0) dont la fin est, par convention, signalée par un zéro, on demande de concevoir un algorithme qui détermine de combien de nombres la suite est constituée (zéro final non compris).

Exercice 4.13 [m]

Etant donné une suite de nombres dont la fin est définie par une sentinelle préalablement choisie par l'utilisateur, on demande de concevoir un algorithme qui détermine de combien de nombres la suite est constituée (sentinelle non comprise).

Exercice 4.14 [m]

Etant donné, pour chaque école d'une commune, le numéro de code de l'école (numéro > 0), le nombre d'élèves inscrits en section primaire de cette école et le nombre d'élèves inscrits en section secondaire de cette école, on demande de concevoir un algorithme qui détermine, pour l'ensemble de la commune, les nombres d'élèves inscrits en section primaire et secondaire ainsi que les pourcentages d'inscrits dans chaque section. La fin du relevé se termine par un numéro inférieur à zéro.

Exercice 4.15 [m]

On veut compter des jetons colorés. On demande de concevoir un algorithme qui lise des entiers représentant à chaque fois une couleur et affiche ensuite le nombre total de jetons rouges, verts, jaunes et bleus. On convient lors de la saisie que 1 = rouge, 2 = vert, 3 = jaune, 4 = bleu, 0 = fin de saisie. Un message d'erreur est affiché si un autre entier est entré.

Exercice 4.16 [m]

On demande d'écrire un algorithme qui lise des temps entrés en heures, minutes et secondes et affiche le temps total en heures, minutes et secondes. La fin de la saisie est indiquée par l'entrée de -1 pour le nombre d'heures. Exemple d'entrée : 3 0 0 1 10 23 0 59 59 -1, qui donne un total de 5 heures, 10 minutes et 22 secondes.

Exercice 4.17 [m]

On demande d'écrire un algorithme qui lise des temps entrés en heures, minutes et secondes et affiche le temps total en heures, minutes et secondes. La fin de la saisie est indiquée par l'entrée d'un temps non valable (heures, minutes ou secondes non valables). Le temps non valable n'est pas comptabilisé. Exemple d'entrée : 3 0 0 1 10 23 0 59 59 -1, qui donne un total de 5 heures, 10 minutes et 22 secondes.

Exercice 4.18 [m]

On demande de construire un algorithme qui, étant donné le solde initial d'un compte bancaire et la suite des montants déposés ou retirés par le titulaire du compte au cours d'une période, (les montants déposés sont des nombres positifs tandis que les montants retirés sont des nombres négatifs), calcule et affiche en fin de période le solde final du compte bancaire, le montant moyen des dépôts effectués ou le message [il n'y a pas eu de dépôt] selon le cas, le montant moyen des retraits effectués ou le message [il n'y a pas eu de retrait] selon le cas. Il est convenu que la fin de la période sera signalée par un montant déposé ou retiré égal à zéro.

Exercice 4.19 [m]

Lors d'une enquête d'opinion préalable aux élections législatives, on demande à un échantillon d'électeurs s'ils pensaient voter pour le candidat X. Les seules réponses possibles des électeurs interrogés étaient :

- ne sait pas encore qui voter (code correspondant : 0)
- vote le candidat X (code correspondant 1)
- ne vote pas le candidat X (code correspondant 2)

Etant donné les réponses codées des électeurs interrogés, on demande d'écrire un algorithme qui donne les pourcentages respectifs d'électeurs hésitants, d'électeurs acquis au candidat X, d'électeurs acquis aux autres candidats. (On supposera que l'utilisateur terminera le dépouillement de cette enquête en communiquant un code négatif).

Exercice 4.20 [d]

En vue de contrôler la qualité des ampoules électriques produites par une usine, un expérimentateur tire au hasard n ampoules dans le stock d'ampoules fabriquées en un jour par cette usine. Il test ensuite chacune des n ampoules sélectionnées.

On demande d'écrire un algorithme qui, étant donné le nombre d'ampoules testées et pour chaque ampoule, le résultat du test (0 = ampoule qui ne fonctionne pas, 1 = ampoule qui fonctionne), calcule et affiche à chaque fois qu'une ampoule est testée, le pourcentage d'ampoules défectueuses parmi celles qui ont déjà été testées et à la fin de l'expérimentation, le pourcentage d'ampoules qui fonctionnent parmi les n ampoules testées. On s'assurera que la valeur saisie comme résultat est toujours 0 ou 1.

Exercice 4.21 [d]

Etant donné une suite de n chiffres (0 à 9), on demande d'écrire un algorithme qui, soit, affiche le message [pas de zéros] si la suite ne contient aucun zéro, soit, s'il trouve au moins un zéro, indique le nombre de zéros que contient cette suite et la position du premier 0 rencontré.

Exercice 4.22 [d]

Un instituteur demande de concevoir un algorithme qui permette à ses jeunes élèves de vérifier et de perfectionner leurs connaissances des tables de multiplication de nombres entiers compris entre 1 et 10. Ses souhaits sont les suivants :

- l'algorithme proposera 20 exercices tirés au hasard,
- si l'élève donne une réponse fausse à un exercice, la chose lui sera signalée par le message [erreur !] et une seconde chance de répondre correctement lui sera donnée. Si l'élève se trompe encore, le message [erreur !] réapparaîtra et le résultat correct sera affiché.

L'exercice sera noté. La notation se réalise en fonction des paramètres suivants :

- toute réponse correcte au premier coup rapporte 1 point,
- toute réponse correcte au deuxième coup rapporte 1/2 point,
- toute réponse définitivement fausse ne rapporte aucun point.

Exercice 4.23 [d]

Dans le cadre d'une enquête portant sur l'évolution du coût de la vie, on a relevé chaque mois le prix moyen du panier de la ménagère acheté dans les grandes surfaces. Le but de l'enquête est d'étudier l'évolution de ce prix moyen au cours de la période étudiée. A cet effet se pose le problème suivant. Etant donné le nombre de mois recensés, on demande de concevoir un algorithme qui :

- affiche pour chaque mois recensé, sauf le premier,
    - le message [croissant], si le prix moyen d'un mois est supérieur au prix moyen du mois précédent,
    - le message [décroissant], si le prix moyen d'un mois est inférieur au prix moyen du mois précédent,
    - le message [en stagnation], si le prix moyen d'un mois est égal au prix moyen du mois précédent,
- détermine, à la fin de la période recensée, le nombre de mois où le prix moyen a été croissant, décroissant, en stagnation

Exercice 4.24 [d]

Le comité organisateur d'une compétition de ski de randonnée voudrait qu'on écrive pour lui un algorithme qui lui permette de déterminer le vainqueur de l'épreuve. Avant le début de l'épreuve, l'utilisateur communique le nombre de concurrents et le temps de parcours maximum imposé (en heures, minutes, secondes).
A chaque prestation d'un concurrent, on enregistre son numéro de dossard, son heure de départ et son heure d'arrivée (en heures, minutes, secondes) et le algorithme détermine le temps de parcours (en heures, minutes, secondes) du concurrent.
Lorsqu'un concurrent a pris le départ mais n'est pas arrivé dans les temps, en raison notamment d'une chute, l'heure d'arrivée sera conventionnellement égale à 0 heure, 0 minute et 0 seconde et il se verra attribuer le temps de parcours maximum imposé.
Si aucun concurrent n'est arrivé dans les temps, on affichera le message [aucune arrivée dans les temps]. Dans le but de simplification, on supposera qu'il ne peut y avoir plusieurs vainqueurs.

Exercice 4.25 [d]

Etant donné un nombre entier tiré au hasard (entre 1 et 100) et inconnu de l'utilisateur on demande de concevoir un algorithme qui permette à l'utilisateur de deviner ce nombre. Les règles du jeu sont les suivantes :

- l'utilisateur a le droit de proposer un maximum de 10 nombres,
- à chaque proposition erronée, sauf la 10^{e}, il lui sera indiqué si le nombre à deviner est plus grand ou plus petit que le nombre proposé,
- la solution sera communiquée à l'utilisateur si celui-ci ne l'a pas devinée après dix essais.

Exercice 4.26 [d]

Un professeur est supposé avoir classé les élèves de sa classe par ordre décroissant de leur résultat global de fin d'année. Etant donné le nombre d'élèves de la classe (nombre >= 2) et pour chaque élève, son numéro et le résultat obtenu (exprimé en nombre de points sur 100), on demande de concevoir un algorithme qui

- affiche le message [incorrect] dès qu'un résultat est supérieur au précédent et arrête son exécution, sans prendre en considération les autres résultats,
- affiche le message [correct] si la série des résultats communiqués est bien en ordre décroissant.

Exercice 4.27 [d]

Etant donné un nombre entier tiré au hasard (entre 1 et 100) et inconnu de l'utilisateur on demande de concevoir un algorithme qui permette à l'utilisateur de deviner ce nombre. Les règles du jeu sont les suivantes :

- à chaque proposition erronée, il lui sera indiqué si le nombre à deviner est plus grand ou plus petit que le nombre proposé.
- si l'utilisateur a un comportement illogique, c'est-à-dire s'il propose un nombre plus petit que sa proposition précédente alors qu'on lui a signalé que la solution était plus grande ou inversement, ce comportement irrationnel lui est signalé et le jeu est arrêté.

Exercice 4.28 [d]

Etant donné un nombre n > 1, on demande de concevoir un algorithme qui dise si n est un nombre premier. Un nombre premier n'est divisible que par 1 et lui-même. Il suffit de vérifier s'il existe un nombre entier entre 2 et la racine carrée de n qui soit un diviseur de n.

Exercice 4.29 [d]

Etant donné un montant de n francs, on demande de concevoir un algorithme qui affiche comment obtenir cette somme au moyen de pièces de 5, 2 et 1 francs et en utilisant le moins possible de pièces (par exemple, 27 Frs = 5×5 Frs + 1×2 Frs). L'algorithme ne doit pas utiliser les opérateurs MOD ou DIV.

Exercice 4.30 [d]

Etant donné deux nombres entiers a et b, on demande de concevoir un algorithme qui affiche leur plus grand commun diviseur (PGCD). Le PGCD de deux nombres peut être trouvé ainsi : si a = b, alors le PGCD est a. Sinon, PGCD(a, b) = PGCD(a - b, b) si a > b, et PGCD(b – a, a) si a < b.

Exercice 4.31 [d]

Un moyen pour calculer la racine carrée d'un nombre réel a et de faire évoluer deux nombres par approximations successives : on part de 1 et de a, et on fait la moyenne m_1 (= (a + 1) / 2). On remplace alors les deux nombres par m_1 et (a / m_1), et on fait leur moyenne m_2. On continue ainsi jusqu'à obtenir une valeur approximant la racine cherchée avec suffisamment de précision. (cela revient à dire que $m_{i+1} = (m_i + a/m_i)/2$)
On demande de concevoir un algorithme qui demande un nombre réel a et calcule sa racine carrée avec une précision de 0.01 (on s'arrête dès que la différence absolue entre m et a/m est < 0.01).

Exercice 4.32 [d]

Etant donné un nombre écrit en binaire, entré par l'utilisateur en tant que suite de 1 et de 0 et dont la fin est indiquée conventionnellement par -1, on demande de concevoir un algorithme qui convertisse ce nombre binaire en un nombre décimal et l'affiche.

Exercice 4.33 [d]

On demande de concevoir un algorithme qui, ayant lu un nombre décimal inférieur ou égal à 255 (< 2^8), affiche ce nombre en binaire avec 8 chiffres (par exemple, 19 donnera 00010011).

Série 5 : tableaux

Matière couverte

Déclaration des variables
Affectation
Sortie, Entrée
Conditions
Boucles
Tableaux

Remarques :

- Le terme *tableau* est utilisé ici aussi bien dans le sens de vecteur que de tableau multidimensionnel.
- Quand cela n'est pas précisé, on suppose que le premier indice des tableaux vaut 1.

Exercices

Exercice 5.1 [f]

On demande d'exécuter les algorithmes suivants et de déterminer les valeurs qui seront affichées à l'écran ainsi que leur signification.

a)

```
var a, n, i : entier
var x [1..10] de entier
n ← 4
a ← 0
Pour i Allant de 1 à n à pas de 1 Faire
    x[i] ← a
Fait
```

b)

```
var a, n, i : entier
var x [1..10] de entier
n ← 4
a ← 1
Pour i Allant de 1 à n à pas de 1 Faire
    x[i] ← a
    a ← a + 2
Fait
```

c)

```
var k, n : entier
var x [1..10] de entier
n ← 3
Pour k Allant de 1 à n Faire
    Lire (x[k])
Fait
```

d)

```
var co, n, i, z : entier
var x [1..100] de entier
n ← 5
co ← 0
Pour i Allant de 1 à n Faire
    Lire (z)
    Si z = 0
    Alors
      co ← co + 1
      x[co] ← i
    FinSi
Fait
```

e)

```
var a, co : entier
var x [1..100] de entier
Lire (a)
co ← 0
Tant Que a >= 0 Faire
    co ← co + 1
    x[co] ← a
    Lire (a)
Fait
```

On supposera que les valeurs successivement introduites par l'utilisateur en réponse à l'opération répétée *lire (a)* sont : 9, 12, 8, 14 et -999.

f)

```
var a, inf, i, n : entier
var x [1..100] de entier
Lire (n)
inf ← 0
Pour i Allant de 1 à n Faire
    Lire (a)
    Si a <= 10
    Alors
      inf ← inf + 1
      x[inf] ← a
    FinSi
Fait
```

On supposera que les valeurs successivement introduites par l'utilisateur en réponse à l'opération répétée *lire (a)* sont : 12, 9 et 7.

Exercice 5.2 [m]

On demande d'écrire un algorithme qui, dans un tableau de 10 éléments,

- donne la valeur 7 à tous les éléments,
- donne la valeur 7 au quatrième élément du tableau et 0 aux autres éléments,
- donne la valeur 7 aux premier et dernier éléments du tableau, et la valeur 0 à tous les autres éléments

Exercice 5.3 [m]

Etant donné une suite de 10 nombres, on demande d'écrire un algorithme capable :

- d'inscrire cette suite dans un tableau,

puis

- d'afficher chaque élément du tableau ainsi constitué en le faisant précéder de la valeur de son indice.

Exercice 5.4 [m]

Etant donné une suite de nombres de longueur a priori inconnue mais dont la fin est indiquée par une "sentinelle" choisie par l'utilisateur, on demande d'écrire un algorithme capable

- d'inscrire cette suite dans un tableau,

puis

- d'afficher chaque élément du tableau ainsi constitué.

Exercice 5.5 [m]

Etant donné une suite de 10 nombres on demande d'écrire un algorithme capable

- d'inscrire cette suite dans un tableau,

puis

- d'afficher chaque élément du tableau, en ayant pris soin, au préalable, d'intervertir les valeurs du premier et du dernier élément du tableau.

Exercice 5.6 [m]

Etant donné une suite de n nombres, on demande d'écrire un algorithme capable

- d'inscrire cette suite dans un tableau,

puis

- d'afficher ensuite les nombres positifs contenus dans le tableau.

Exercice 5.7 [m]

Etant donné une suite de n nombres, on demande d'écrire un algorithme capable d'afficher cette suite de nombres dans l'ordre inverse où ils ont été communiqués par l'utilisateur.

Exercice 5.8 [m]

Etant donné une suite de n nombres différents, on demande d'écrire un algorithme qui

- enregistre ces n nombres dans un tableau,
- détermine la valeur maximum que contient ce tableau, ainsi que la position de l'élément du tableau contenant cette valeur maximum.

Exercice 5.9 [m]

Etant donné n plus petit que 100, écrire un algorithme qui calcule les n premiers termes de la suite de Fibonacci en les enregistrant dans un tableau et les affiche. La suite de Fibonacci est 1, 1, 2, 3, 5, 8, 13…, où chaque terme est la somme des deux précédents.

Exercice 5.10 [m]

Etant donné n et deux vecteurs d'entiers de taille n entrés par l'utilisateur, écrire un algorithme qui calcule leur produit scalaire. Par exemple, le produit scalaire de [2 5 7] et de [3 4 -1] est $2\times3 + 5\times4 + 7\times(-1) = 19$.

Exercice 5.11 [m]

Etant donné une suite de n nombres, écrire l'algorithme qui enregistre cette suite dans un tableau, puis trouve les 2 nombres de la suite qui ont le plus grand écart entre eux et affiche cet écart.

Exercice 5.12 [m]

Etant donné un entier n et une matrice carrée $n\times n$ d'entiers, écrire un algorithme qui enregistre cette matrice dans un tableau, puis calcule la trace de cette matrice (la trace est la somme des éléments sur la diagonale où le numéro de ligne est égal au numéro de colonne).

Exercice 5.13 [m]

Etant donné le nombre d'élèves d'une classe et la note obtenue par chacun d'eux lors d'une interrogation, on demande d'écrire un algorithme qui affiche la moyenne de la classe et, pour chaque élève, son numéro et l'écart de sa note par rapport à la moyenne de la classe. Les élèves sont numérotés de 1 au nombre d'élèves de la classe.

Exercice 5.14 [m]

Etant donné la situation des n comptes des clients d'une agence bancaire en début de journée, on demande d'écrire un algorithme qui enregistre, tout au long de la journée, les opérations de dépôt et de retrait faites par ses clients au cours de la journée et fournit, en fin de journée, la situation de tous les comptes des clients. On conviendra que :

- les comptes débiteurs ont un solde négatif et les comptes créditeurs ont un solde positif ou nul,
- les comptes sont répertoriés par un numéro allant de 1 à n,
- les opérations de dépôt seront identifiées par la valeur conventionnelle 1, celle de retrait par la valeur -1,
- les retraits supérieurs à 25'000 Frs sont refusés,
- la fin de la journée est identifiée par une opération = 0.

Exercice 5.15 [m]

Une chaîne de montage monte un seul type de voiture en 12 coloris différents (numérotés de 1 à 12). On demande d'écrire un algorithme qui permette :

- d'enregistrer en début de journée l'état du stock de voitures fabriquées pour chaque coloris (pour chaque coloris, l'ordinateur affichera le numéro du coloris et l'utilisateur introduira la quantité disponible correspondante),
- de mettre à jour l'état de stock chaque fois qu'une voiture terminée quitte la chaîne de montage (et entre donc dans le stock disponible) ou qu'une voiture est expédiée chez un revendeur (et sort donc du stock disponible). L'utilisateur communiquera pour chaque mouvement de stock, la nature du mouvement et le numéro du coloris de la voiture
- en fin de journée, d'afficher le nouvel état du stock (numéro de coloris et quantité disponible). On fera précéder d'un astérisque (*) tout numéro de coloris dont le stock est épuisé.

On conviendra que l'utilisateur communiquera la nature du mouvement de stock selon les conventions suivantes :

- entrée dans le stock : type de mouvement = 1,
- sortie du stock : type de mouvement = -1.

La fin de la journée et donc la fin des mouvements de stock sera signalée conventionnellement par un type de mouvement égal à 0.

Exercice 5.16 [m]

Etant donné une suite de chiffres (0 à 9) de longueur quelconque dont la fin est indiquée par -1, on demande d'écrire un algorithme qui détermine le nombre d'occurrences de chaque chiffre dans cette suite.

Exercice 5.17 [m]

Soit un jeu classique de 32 cartes à jouer comportant des cartes de l'as au 7. Etant donné une main de 8 cartes extraites de ce jeu, on demande d'écrire un algorithme qui détermine le nombre d'occurrences de chaque type de carte dans la main. On conviendra que :

- les cartes du 7 au 10 sont représentées par leur valeur;
- un valet est représenté par 11
- une dame est représentée par 12
- un roi est représenté par 13
- un as est représenté par 14

Exercice 5.18 [m]

Le service marketing d'un grand magasin voudrait mieux connaître le profil d'âge de sa clientèle. A cet effet, on a relevé pendant plusieurs jours la catégorie d'âge des clients entrant dans le magasin. Ces catégories d'âge sont codées comme suit :

âge	**code**
18 à 25 ans	1
26 à 35 ans	2
36 à 45 ans	3
46 à 55 ans	4
56 à 65 ans	5
66 et plus	6

Etant donné la suite des codes d'âge ainsi relevés, on demande d'écrire un algorithme qui donne pour résultat le pourcentage de clients de chaque catégorie. Il est convenu que la fin du relevé sera signalée par un code de catégorie d'âge égal à 0.

Exercice 5.19 [m]

Le service marketing d'une chaîne de magasins désirerait mieux connaître la composition de son chiffre d'affaire et, plus spécialement, l'apport relatif des ventes classées selon leur importance. A cet effet, les ventes ont été réparties en 7 catégories selon leur montant :

catégorie 0	montant < 1000
catégorie 1	1000 <= montant < 2000
catégorie 2	2000 <= montant < 3000
catégorie 3	3000 <= montant < 4000
catégorie 4	4000 <= montant < 5000
catégorie 5	5000 <= montant < 6000
catégorie 6	montant >= 6000

Etant donnée la suite des ventes réalisées pendant une période, on demande de concevoir un algorithme qui détermine :

- le montant moyen des ventes réalisées
- la part relative en % que représente chaque catégorie de ventes dans le chiffre d'affaire total

On conviendra que la fin du relevé sera signalée par un montant de vente négatif ou nul.

Exercice 5.20 [d]

Etant donnée une suite de n nombres entiers et un nombre entier x quelconque, on demande d'écrire un algorithme qui :

- enregistre ces n nombres dans un tableau
- vérifie si le nombre x appartient au tableau et affiche, selon le cas, l'indice du premier élément où le nombre x a été enregistré ou le message [le nombre recherché n'appartient pas à la suite].

Exercice 5.21 [d]

Dans un championnat de jeu de dames regroupant plusieurs joueurs, chaque joueur joue exactement une partie contre chaque adversaire. L'organisateur aimerait avoir la liste des parties qui doivent être jouées. Par exemple, s'il y a 3 joueurs A, B et C, les parties sont [A contre B], [A contre C] et [B contre C].
On demande d'écrire un algorithme qui demande d'entrer les noms des joueurs et affiche ensuite la liste des parties devant être jouées. On suppose qu'il y a au maximum 100 joueurs et que l'utilisateur entre un nom vide pour indiquer la fin de la saisie.

Exercice 5.22 [d]

Le dirigeant du service marketing d'un grand magasin désire se faire une idée précise de la répartition des ventes selon les jours de la semaine. A cet effet, il dispose des chiffres de vente journaliers recensés pendant une certaine période. Ce relevé indique pour chaque journée recensée le code du jour de la semaine concernée (nombre entier entre 1 et 6) et le chiffre de vente correspondant.
Ce dirigeant désire qu'on réalise pour lui un algorithme qui permette d'obtenir, sur base des données fournies par le relevé :

- le montant moyen des ventes de chaque jour de la semaine,
- la part en pour-cent que représente le montant moyen des ventes de chaque jour par rapport au montant moyen des ventes par semaine.

On conviendra que :

- dans le relevé fourni, les jours de la semaine ne se suivent pas forcément du lundi au samedi, le magasin pouvant être fermé certaines journées,
- le montant moyen des ventes par semaine est égal à la somme des montants moyens des ventes de chaque jour de la semaine,
- l'utilisateur n'aura pas à communiquer lors de l'exécution de l'algorithme le nombre de journées que comprend le relevé.

Exercice 5.23 [d]

On a réalisé une enquête sur les dépenses faites à l'occasion de la rentrée scolaire par des ménages ayant des enfants en âge d'école.
Les résultats de cette enquête donnent, pour chaque ménage interrogé :

- le nombre d'enfants en âge d'école de ce ménage (1, 2 .., 10),
- le montant des dépenses de ce ménage.

Etant donné les résultats de cette enquête, on demande d'écrire un algorithme qui calcule et affiche la dépense moyenne par enfant pour les ménages qui ont 1, 2, 3, ..., 10 enfants.
Il est convenu que la fin du relevé sera signalée en introduisant un nombre d'enfants égal à zéro.

Exercice 5.24 [d]

On demande d'écrire un algorithme qui permette de jouer au jeu de Marienbad. Au départ le jeu comporte un certain nombre de tas d'allumettes, chaque tas pouvant contenir un nombre différent d'allumettes. Chaque joueur doit prendre à son tour un nombre quelconque d'allumettes dans un seul tas. Le joueur qui sera obligé de prendre l'allumette restante a perdu. On convient de concevoir l'algorithme de telle sorte que :

- le jeu se joue à deux joueurs,
- le nombre de tas que compte le jeu sera communiqué lors de l'exécution de l'algorithme,
- pour chaque tas, le nombre d'allumettes qu'il comporte sera communiqué lors de l'exécution de l'algorithme,
- avant chaque intervention d'un joueur, on affichera l'état du jeu, à savoir le nombre d'allumettes restant dans chaque tas et le numéro du joueur qui doit jouer,
- le joueur communiquera le numéro du tas dans lequel il compte prélever des allumettes ainsi que le nombre d'allumettes prélevées.

Exercice 5.25 [d]

Etant donné une suite de n nombres, on demande d'écrire un algorithme qui enregistre ces n nombres dans un tableau de sorte que tous les nombres positifs ou nuls soient enregistrés en début de tableau et tous les nombres négatifs en fin du tableau et affiche ensuite le contenu du tableau ainsi constitué.

Exercice 5.26 [d]

Soit une année à 2 semestres et une classe, qui comporte 20 élèves qui suivent une formation composée de 10 matières. Chaque élève reçoit une note par matière, par semestre.
On demande d'écrire un algorithme qui :

1- remplit le tableau,
2- pour chaque élève, calcule sa moyenne au semestre 1, puis sa moyenne d'année,
3- pour chaque matière, calcule la moyenne de classe au semestre 2 puis la moyenne annuelle,
4- calcule la moyenne générale (égale à la moyenne de toutes les notes individuelles).

Exercice 5.27 [d]

On demande d'écrire un algorithme qui tient à jour une liste de noms stockée dans un tableau. L'algorithme lit une commande, qui peut être

- "afficher" pour afficher les noms, précédés de leur indice de tableau,
- "ajouter" pour lire un nom et l'ajouter à la liste,
- "effacer" pour lire un indice et effacer le nom correspondant,
- "quitter" pour terminer.

L'algorithme écrit un message d'erreur si la commande entrée n'est pas reconnue. On suppose qu'il n'y a jamais plus de 100 noms.

Exercice 5.28 [d]

Etant donné deux entiers m et n, et une matrice n×m qui sera stockée au préalable dans un tableau, on demande d'écrire un algorithme qui donne la somme de tous les éléments de la matrice, la somme des éléments qui sont au bord de la matrice (1re et dernière colonnes, et 1re et dernière lignes) et enfin la somme des éléments qui ne sont pas au bord de la matrice.

Exercice 5.29 [d]

Etant donné un entier n et une matrice carrée n×n d'entiers, écrire un algorithme qui enregistre cette matrice, puis indique s'il s'agit ou non de la matrice identité (des 1 sur la diagonale où le numéro de ligne est égal au numéro de colonne, des 0 partout ailleurs).

Exercice 5.30 [d]

On dispose d'un échiquier sur lequel sont placés des pièces du jeu. On voudrait savoir combien de pièces sont placées sur les cases blanches.
On demande d'écrire l'algorithme qui demande pour chacune des 64 cases (blanche ou noire) s'il y a une pièce sur celle-ci, stocke le résultat dans un tableau et indique finalement le total des de pièces qui sont sur une case blanche.On partira de l'idée que la première case est une case blanche.

Exercice 5.31 [d]

On veut déterminer les nombres premiers inférieurs à un n donné, au moyen du crible d'Eratosthène. Pour cela, on utilise un tableau de booléens de taille n et l'on "marque" dans le tableau tous les indices qui sont multiples de 2, 3, 4, 5... . Il suffit de marquer les multiples des nombres inférieurs à la racine carrée de n.
Les indices du tableau qui n'ont pas été marqués sont les nombres premiers recherchés, qu'il ne reste plus qu'à afficher.
On demande d'écrire un algorithme/algorithme qui lit un entier n et affiche la liste des nombres premiers inférieurs à n.

Exercice 5.32 [d]

Etant donné n, m et une matrice de n lignes × m colonnes, on demande d'écrire un algorithme qui affiche les éléments de cette matrice qui sont à la fois le plus grand de leur ligne et le plus petit de leur colonne. S'il n'existe pas de tel élément, le message [aucun élément satisfaisant] est affiché.

Exercice 5.33 [d]

Etant donné n et une matrice carrée n×n d'entiers, on demande d'écrire un algorithme qui écrive toutes les valeurs de la matrice qui ne s'y trouvent qu'une seule fois.

Exercice 5.34 [d]

Etant donné m, n, o et deux matrices d'entier données de dimensions m×n et n×o, écrire l'algorithme qui calcule et affiche le produit des deux (matrice de taille m×o).

Série 6 : procédures & fonctions

Matière couverte

Conditions
Boucles
Tableaux
Procédures et fonctions

Remarques :

- Le terme *tableau* est utilisé ici aussi bien dans le sens de vecteur que de tableau multidimensionnel.
- Quand cela n'est pas précisé, on suppose que le premier indice des tableaux vaut 1.

Exercices

Exercice 6.1 [f]

On demande d'exécuter les algorithmes suivants et de déterminer les valeurs qui seront affichées à l'écran.

a)

```
/*  début de l'algorithme principal   */
Var a, b : entier

a ← 1
b ← 2
modifier(a, b)
Ecrire (a, b)          /*  fin de l'algorithme principal   */

                       /*  début des procédures et fonctions   */
Procédure modifier (a : entier,  var b : entier)
    a ← a + 10
    b ← b + 20
Fin Procédure          /*  fin des procédures et fonctions   */
```

b)

```
/*  début de l'algorithme principal   */
var n, m, k, prod : entier
m ← 4
n ← 3
Pour k allant de 1 à m
Faire
    inconnue (n, prod)
    Ecrire (k, " ", prod)
Fait                   /*  fin de l'algorithme principal   */

                       /*  début des procédures et fonctions   */
Procédure inconnue (n : entier,  var p : entier)
    Var y, j : entier
    p ← 1
    Pour j allant de 1 à n
    Faire
          Lire (y)
          p ← p * y
    Fait
Fin Procédure          /*  fin des procédures et fonctions   */
```

On supposera que les valeurs successivement introduites par l'utilisateur en réponse à l'opération répétée *LIRE (y)* sont :

```
2     3     9
6     5     2
7     8     1
3     6     3
```

c)

```
          /*  début de l'algorithme principal   */
var sent, som, co : entier
sent ← 0
somme_var (sent, som, co)
Ecrire (som, " ", co)
          /*  fin de l'algorithme principal   */


         /*  début des procédures et fonctions   */
Procédure somme_var (f : entier;  var s, c : entier)
    var x : entier
    c ← 0
    s ← 0
    Lire (x)
    Tant que x <> f
    Faire
          s ← s + x
          c ← c + 1
          Lire (x)
    Fait
Fin Procédure
          /*  fin des procédures et fonctions   */
```

On supposera que les valeurs successivement introduites par l'utilisateur en réponse à l'opération répétée *Lire (x)* sont :
7 -3 8 0

d)

```
          /*  début de l'algorithme principal   */
Const MAX_NB c'est 100

Var i, k, l, n : entier
Var y [1..MAX_NB] de entier

Pour i allant de 1 à 4
Faire
    y[i] ← i * 2
Fait
k ← 1
l ← 4
échange (l, k, y)
n ← 4
affiche_tab (n, y)
          /*  fin de l'algorithme principal   */


         /*  début des procédures et fonctions   */
Procédure affiche_tab (nt : entier,  tx [1..MAX_NB] de entier)
    Var i : entier
    Pour i allant de 1 à nt
    Faire
          Ecrire (i, " ", tx[i])
    Fait
Fin Procédure

Procédure échange (i, j : entier,  var tx [1..MAX_NB] de entier)
    var t : entier
    t ← tx[i]
    tx[i] ← tx[j]
    tx[j] ← t
Fin Procédure
          /*  fin des procédures et fonctions   */
```

e)

```
/*  début de l'algorithme principal   */
Const MAX_NB c'est 100

Var a, b, k, l, n : entier
Var y [1..MAX_NB] de entier

Pour i allant de 1 à 4  Faire
   y[i] ← i * 2
Fait
k ← 1
l ← 4
échange (k, l, y)
a ← 2
b ← 4
échange (a, b, y)
n ← 4
affiche_tab (n, y)   /*  fin de l'algorithme principal   */

                     /*  début des procédures et fonctions   */
Procédure affiche_tab (nt : entier, tx [1..MAX_NB] de entier)
   Var i : entier
   Pour i allant de 1 à nt  Faire
         Ecrire (i, " ", tx[i])
   Fait
Fin Procédure

Procédure échange (i,j : entier, var tx [1..MAX_NB] de entier)
   var t : entier
   t ← tx[i]
   tx[i] ← tx[j]
   tx[j] ← t
Fin Procédure        /*  fin des procédures et fonctions   */
```

f)

```
/*  début de l'algorithme principal   */
Const MAX_NB c'est 100

Var i, j, m, ind : entier
Var y [1..MAX_NB] de entier

y[1] ← 10
y[2] ← 9
y[3] ← 5
y[4] ← 15
i ← 1
j ← 4
minimum (y, i, j, m, ind)
Ecrire (m, " ", ind)       /*  fin de l'algorithme principal   */

                           /*  début des procédures et fonctions   */
Procédure minimum (x [1..MAX_NB] : entier, ideb, ifin : entier,
                                                   var min, imin : entier)
   var i : entier
   min ← x[ideb]
   imin ← ideb
   Pour i allant de ideb + 1 à ifin  Faire
         Si x[i] < min
         Alors min ← x[i]
               imin ← i
         FinSi
   Fait
Fin Procédure           /*  fin des procédures et fonctions   */
```

Exercice 6.2 [f]

Ecrire une fonction qui prend en paramètres 6 entiers h1, m1, s1, h2, m2 et s2 représentant des durées et qui retourne -1 si la première durée est plus petite que la seconde, 0 si les durées sont identiques et 1 si la première durée est plus grande que la seconde.

Exercice 6.3 [f]

Ecrire une fonction qui prend en paramètres un tableau de caractères de taille 100, et le nombre effectif n de caractères existant dans celui-ci, et qui retourne le nombre de voyelles contenues dans le tableau.

Exercice 6.4 [f]

Ecrire une fonction qui prend en paramètres un tableau d'entiers et son nombre d'éléments n, et retourne vrai s'il contient plus de nombre pairs que de nombres impairs.

Exercice 6.5 [f]

Ecrire un algorithme qui calcule l'addition de la factorielle de trois nombres entrés au clavier.

Exercice 6.6 [f]

Ecrire un algorithme qui affiche les entiers compris entre x et y (lus au clavier, x et y dans un ordre quelconque, x et y inclus) dans l'ordre croissant. Après chaque lecture de x, y, on affiche les entiers, et on s'arrête quand le x entré au clavier est -1.

Exercice 6.7 [f]

Ecrire un algorithme qui tire au hasard 10 fois 2 nombres, puis demande à un élève d'effectuer la multiplication, puis l'addition des deux nombres. Pour chaque calcul, l'élève aura deux chances. Pour chaque réponse juste au 1^{er} coup, il aura 1 point, pour chaque réponse juste au 2^{e} coup, il aura 0.5 point. L'algorithme affichera la note (= le total des points) en addition, la note en multiplication, et la moyenne des deux notes.

Exercice 6.8 [f]

Ecrire un algorithme qui lit 10 suites de 3 nombres (compris entre 1 et 10), et affiche combien de ces suites ont été rentrées dans l'ordre croissant.

Exercice 6.9 [f]

On considère que l'on a à disposition un tableau de 10 noms (chaînes de caractères). On veut effectuer des recherches de noms pour savoir si une personne est enregistrée ou non. Ecrire une fonction qui effectue la recherche d'un nom (passé en paramètre) dans un tableau.
Ecrire l'algorithme utilisant la fonction déjà écrite et qui lit des noms et affiche au fur et à mesure si le nom est enregistré dans le tableau ou non. La lecture s'arrête quand l'utilisateur entre 'zzzzzz'.

Exercice 6.10 [f]

On demande de rédiger une procédure qui, à partir d'un tableau de n nombres réels, calcule et affiche pour chaque nombre enregistré dans le tableau, l'indice, le nombre et la part en pour-cent de ce nombre par rapport à la somme des nombres. On supposera que le premier indice du tableau est 1.

Exercice 6.11 [m]

On demande d'écrire une procédure qui calcule le maximum des valeurs enregistrées dans un "morceau" de tableau ainsi que l'indice de l'élément contenant ce maximum. Les indices délimitant les éléments dont on calcule le maximum seront passés en paramètres de la procédure.

Exercice 6.12 [m]

On demande d'écrire une fonction dont le résultat sera l'indice de la plus grande des valeurs enregistrées dans un "morceau" de tableau. Les indices délimitant les éléments dont on calcule le maximum seront passés en paramètres de la fonction.

Exercice 6.13 [m]

On demande d'écrire une procédure qui inverse le contenu d'un tableau à une dimension.

Exercice 6.14 [m]

Etant donné un tableau de caractères de taille n, écrire une fonction qui dise si les caractères dans le tableau forment un palindrome (les caractères sont les mêmes lus dans un sens ou dans l'autre, par exemple 'A', 'b', 'c', 'D', 'c', 'b', 'A').

Exercice 6.15 [m]

On demande de rédiger le texte des sous-algorithmes dont les spécifications sont les suivantes :

a) affiche_tab
paramètres d'entrée : nt, tx
résultat : -
objectif : afficher l'indice et les valeurs des éléments d'indice 1 à nt du tableau tx.

b) lecture_tab
paramètres d'entrée : nt
résultat : tx
objectif : lire une suite de nt nombres entiers donnés par l'utilisateur et les ranger dans le tableau tx, de l'élément d'indice 1 à l'élément d'indice nt.

c) init_tab
paramètres d'entrée : nt
résultat : tx
objectif : initialiser à 0 tous les éléments d'indice 1 à nt du tableau tx.

d) moyenne
paramètres d'entrée : somme, n
résultat : moy
objectif : sur la base de la somme de n nombres entiers, si n <> 0, ranger dans moy la moyenne de ces nombres et l'afficher.

e) somme_fixe
paramètres d'entrée : nt
résultat : somme
objectif : lire une suite de nt nombres entiers communiqués par l'utilisateur et ranger dans som la somme de ces nombres.

f) somme_var
paramètres d'entrée : s
résultat : somme, nt
objectif : lire une suite de nombres entiers communiqués par l'utilisateur et dont la fin est signalée par une sentinelle (s), calculer la somme de ces nombres (som) et la longueur (nt) de cette suite.

Pour résoudre les problèmes suivants, on pourra s'inspirer des procédures spécifiées à l'exercice n°15.

Exercice 6.16 [m]

Etant donné la liste des différents achats effectués par chaque membre d'un club de lecture au cours de l'année écoulée, on demande d'écrire un algorithme qui affiche la moyenne des achats annuels des membres ainsi que la liste des membres dont le total des achats effectués au cours de l'année a été supérieur ou égal à la moyenne des achats annuels des membres.

On conviendra que
- tous les membres du club sont identifiés par un code numérique dont les valeurs ne sont pas nécessairement consécutives,
- l'utilisateur du algorithme communiquera, lors de chaque exécution du algorithme, le nombre de membres du club,
- le nombre d'achats effectués est variable d'un membre à l'autre,
- la fin de la liste des montants des achats effectués par un membre particulier sera signalée par -1,
- si un membre n'a effectué aucun achat au cours de l'année, il interviendra tout de même dans le calcul de la moyenne des achats.

Exemple d'exécution :

Nombre de membres ? 5

code	*Achats*				
101	100	500	300	-1	
572	250	300	450	400	-1
221	200	700	800	-1	
315	-1				
132	600	550	450	900	-1

Moyenne des achats annuels : 1300 F
Les membres suivants ont effectué au cours de l'année des achats dont le total s'élève au moins à 1300 F :
572
221
132

Exercice 6.17 [m]

Plusieurs équipes ont participé à un tournoi amical de tennis en double mixte. Pour chacune des équipes, on dispose des résultats suivants :
- le nombre de rencontres gagnées,
- le nombre total de jeux gagnés au cours des rencontres jouées.

L'organisateur du tournoi nous demande d'écrire à son intention un algorithme capable d'établir le classement final du tournoi. A cet effet, il pose le problème suivant :
Etant donnés
- le nombre d'équipes participant au tournoi ;
- pour chaque équipe, son résultat sous la forme : nombre de rencontres gagnées, nombre total de jeux gagnés ;
on demande d'écrire un algorithme qui affichera par ordre croissant de numéro d'équipe le classement obtenu par chacune des équipes.

On conviendra que :
- chaque équipe est identifiée par un numéro,
- si deux équipes ont le même nombre de rencontres gagnées, on regarde le nombre total de jeux gagnés,
- si deux ou plusieurs équipes ont gagné le même nombre de rencontres et de jeux, elles sont classées ex-æquo.

Le dialogue utilisateur-ordinateur pourra se présenter ainsi, les réponses de l'utilisateur étant soulignées :

Nombre d'équipes :5
Résultats des 5 équipes :

n° d'équipe	rencontres gagnées	jeux gagnés
1	3	41
2	5	63
3	1	8
4	5	69
5	6	84

Le classement est le suivant :

n° d'équipe	classement
1	4
2	3
3	5
4	2
5	1

Exercice 6.18 [m]

Le Scrabble est un jeu où chaque joueur dispose, au début d'une partie, d'un certain nombre de lettres prises dans un "pot commun" et valant un certain nombre de points, (par exemple, la lettre A vaut 1 point, la lettre K vaut 10 points, etc.). Le jeu consiste, pour chaque joueur, à essayer chacun à son tour de placer, en respectant certaines règles, une partie ou toutes les lettres dont il dispose sur un damier de façon à former des mots corrects. Chaque fois qu'un joueur réussit à placer une ou plusieurs lettre(s) sur le damier, il obtient un nombre de points égal à la somme des valeurs des lettres placées et prend un nombre équivalent de nouvelles lettres dans le "pot commun".
Quand, étant donné les lettres dont il dispose et l'état du damier, un joueur n'arrive à placer aucune lettre, il "passe son tour". La partie se termine quand aucun joueur n'est capable d'ajouter au moins une lettre au damier ou lorsque le "pot commun" est épuisé.

Les membres d'un club de Scrabble demandent d'écrire à leur intention un algorithme qui, lors d'une partie de Scrabble, leur rende après chaque intervention d'un joueur, le service suivant.

Etant donné
- le numéro du joueur qui vient d'intervenir dans le jeu,
- la liste des points qu'il a obtenus pour chaque lettre placée lors de cette dernière intervention,

on demande d'écrire un algorithme qui :
- calcule et affiche le total des points obtenus par ce joueur pour cette dernière intervention,
- affiche le total des points accumulés depuis le début de la partie par chacun des joueurs.

On admettra que :
- le nombre de participants à une partie de Scrabble, même s'il varie d'une partie à l'autre, est évidemment connu en début de partie et est au maximum de 6,
- chacun des joueurs est identifié par un numéro variant de 1 au nombre de participants à la partie (par exemple s'il s'agit d'une partie à 3 joueurs, ceux-ci sont identifiés par les numéros 1, 2 et 3),
- le nombre de lettres placées lors de chaque intervention d'un joueur est variable, ce qui amènera à clôturer la liste des points obtenus lors d'une intervention par une valeur conventionnelle, à savoir zéro,
- en début de partie et donc d'exécution du algorithme, on ignore combien d'interventions comptera la partie. En conséquence, il est convenu de signaler la fin de la partie en communiquant un numéro de joueur conventionnel qui arrêtera l'exécution du algorithme.

Le dialogue utilisateur-ordinateur pourra se présenter ainsi, les réponses de l'utilisateur étant soulignées :

Nombre de participants ? 3

Joueur n° ? 1
Points ?
2
4
8
0
Total : 14
Totaux depuis début partie :

numéro joueur	total points
1	14
2	0
3	0

Joueur n° ? 2
Points ?
2
2
3
4
0
Total : 11
Totaux depuis début partie :

numéro joueur	total points
1	14
2	11
3	0

Joueur n° ? 3
Points ?
4
6
0
Total : 10
Totaux depuis début partie :

numéro joueur	total points
1	14
2	11
3	10

Joueur n° ? 1
Points ?
7
0
Total : 7
Totaux depuis début partie :

numéro joueur	total points
1	21
2	11
3	10

Joueur n° ? 3
Points ?
3
4
0
Total : 7
Totaux depuis début partie :

numéro joueur	total points
1	21
2	11
3	17

Joueur n° ? -1

Exercice 6.19 [m]

Les organisateurs d'une compétition de patinage artistique demandent d'écrire un algorithme qui, après la prestation de chaque concurrent identifié par un numéro de dossard, leur rend le service suivant.

Etant donné :
- le numéro du dossard du concurrent qui vient d'effectuer sa prestation,
- les points qui lui ont été attribués par un jury de 6 membres,

on demande de concevoir un algorithme qui

- calcule et affiche la cote finale de ce concurrent en effectuant la moyenne arithmétique des points qui lui ont été attribués par chacun des 6 membres du jury,
- affiche dans l'ordre de leur passage la liste des concurrents ayant déjà terminé leur prestation en rappelant pour chacun d'eux le numéro de leur dossard et la cote finale qui leur a été attribuée.

On tiendra compte du fait que :

- les numéros de dossard des concurrents ne sont pas nécessairement consécutifs et ne correspondent pas nécessairement à l'ordre dans lequel les concurrents effectuent leur prestation,
- le nombre de concurrents n'est pas connu en début de compétition (et donc en début d'exécution du algorithme) car un abandon de dernière minute est toujours possible.

Le dialogue utilisateur-ordinateur se présentera ainsi, les réponses de l'utilisateur étant soulignées :

Numéro de dossard ? 5
membre jury points attribués
1 8.9
2 8.2
3 8.6
4 8.7
5 9.3
6 9.1

cote finale : 8.80

liste des concurrents ayant terminé leur prestation :
numéro de dossard cote finale
5 8.80

Numéro de dossard ? 3
membre jury points attribués
1 9.1
2 9.2
3 9.5
4 9.6
5 8.9
6 9.6

cote finale : 9.30

liste des concurrents ayant terminé leur prestation :
numéro de dossard cote finale
5 8.80
3 9.30

Numéro de dossard ? -1

Exercice 6.20 [m]

Une chaîne de télévision veut organiser une grande soirée "Elections communales". Les personnalités politiques locales participeront à différents débats, régulièrement interrompus par l'affichage de résultats électoraux de plus en plus complets. Six partis sont en compétition : ils sont numérotés de 1 à 6. Chacun des bureaux de vote éparpillés dans la ville transmettra dès que possible à la chaîne de télévision le résultat de son dépouillement (le nombre de voix récoltées par chaque parti et le nombre de bulletins blancs ou nuls).

On vous demande de construire un algorithme qui

• après chaque réception des résultats du dépouillement d'un bureau de vote, réponde au problème suivant :

étant donné
- le numéro du bureau de vote,
- le nombre de bulletins blancs ou nuls comptés dans ce bureau de vote et le nombre de voix obtenues par chacun des six partis,

on demande
- l'affichage du nombre de bureaux de vote d'où l'on a déjà reçu des résultats,
- pour l'ensemble des bureaux de vote d'où l'on a déjà reçu des résultats, l'affichage du score obtenu jusque là par chaque parti, exprimé en nombre de voix et en pourcentage du nombre total de bulletins dépouillés (bulletins blancs ou nuls compris),
- à la fin de la soirée "Elections communales", le algorithme doit afficher le nombre maximum de voix récoltées par un parti et le numéro du parti ou des partis (en cas d'ex-æquo) qui a (ont) obtenu(s) ce nombre de voix.

On tiendra compte du fait que le nombre de bureaux qui parviendront à achever leur dépouillement avant la fin de l'émission est inconnu a priori. Il est donc convenu de marquer la fin de l'émission en introduisant un numéro de bureau nul.

Un conseil : considérez les électeurs qui ont voté blanc ou nul comme des électeurs du parti 0 (parti fictif).

Exercice 6.21 [m]

Les membres du Conseil d'une Faculté doivent procéder tous les 3 ans à l'élection du Doyen de la Faculté.
Cette élection se déroule ainsi :
- on établit au préalable la liste des professeurs qui peuvent être élus à cette fonction; ces professeurs sont appelés candidats;
- chaque membre du Conseil vote pour un et un seul candidat;
- à l'issue du 1er tour de vote, le candidat qui a obtenu le plus de voix est élu doyen (règle simplifiée); si par hasard 2 ou plusieurs candidats ont obtenu le score le plus élevé, on procède à un nouveau vote.

Etant donné
- le nombre de candidats à la fonction de doyen,
- la suite des votes émis par les membres du Conseil facultaire,

on vous demande de concevoir un algorithme qui, à l'issue du premier tour,
- calcule et affiche les nombres de voix obtenus par chaque candidat,
- ensuite,
  soit détermine et affiche le numéro du candidat élu Doyen si aucun autre candidat n'a obtenu autant de voix que lui,
  soit affiche le message "procéder à un nouveau vote" dans les autres cas.

On conviendra que
- les candidats sont numérotés de 1 au nombre de professeurs éligibles,
- le vote d'un membre du Conseil est représenté par un numéro identifiant un candidat,
- le nombre de membres du Conseil participant au vote ne sera pas communiqué par l'utilisateur en début d'exécution de l'algorithme, la fin du dépouillement du vote étant signalée par un numéro de candidat égal à 0.

Exercice 6.22 [d]

On dispose d'une liste de codes (nombres entiers positifs) qui, en principe, sont tous différents. On demande de concevoir un algorithme qui vérifie, à chaque code introduit, si ce code est bien unique. La fin de la liste des codes sera indiquée par un code nul.

Exercice 6.23 [d]

Supposons que l'on dispose de l'ensemble des procédures suivantes dont les spécifications sont précisées :

minimum
paramètres d'entrée : tx, ideb ,ifin
résultat : tmin, imin
objectif : renvoyer dans tmin la plus petite valeur des éléments du tableau (tx) allant de l'indice ideb à l'indice ifin et ranger dans imin l'indice de l'élément du tableau contenant ce minimum (tx est un tableau de nombres entiers)

lecture_var
paramètres d'entrée : s
résultat : tx, nb
objectif : lire une suite de nombres entiers communiqués par l'utilisateur et dont la fin est signalée par une sentinelle (s); ranger cette suite de nombres dans le tableau tx de l'indice 1 à l'indice nb et déterminer la longueur nb de la suite.

échange
paramètres d'entrée : tx, i, j
résultat : tx
objectif : échanger les valeurs des éléments d'indices i et j du tableau d'entiers tx

affiche_tab
paramètres d'entrée : nt, tx
résultat : -
objectif : afficher à l'écran la valeur des éléments d'indice 1 à nt du tableau tx.

On demande d'écrire un algorithme qui, étant donné une suite de nombres entiers communiqués par l'utilisateur et terminé par une sentinelle désignée par lui

- range ces nombres **dans un tableau**
- trie ce tableau par ordre croissant
- affiche les valeurs des éléments du tableau ainsi trié. Il n'est pas demandé de rédiger le texte des procédures.

Exercice 6.24 [d]

Etant donnés les 6 numéros gagnants d'un jeu de loto (simplifié), et le nombre de joueurs participant à ce jeu, on demande de construire un algorithme qui prend connaissance des 6 numéros proposés par chacun des joueurs et affiche le nombre de gagnants de chaque rang. Le rang d'un gagnant est déterminé en tenant compte du nombre de numéros corrects proposés par le joueur : le gagnant est de rang 1, 2, 3, 4, 5 ou 6 selon qu'il a proposé respectivement 6, 5, 4, 3, 2 ou 1 numéro(s) correct(s).

Exercice 6.25 [d]

Ecrire une fonction qui prend en paramètres un tableau d'entiers et sa taille n, et retourne le deuxième nombre plus petit.

Exercice 6.26 [d]

On voudrait pouvoir transformer un tableau à deux dimensions en un tableau à une dimension. On demande donc d'écrire une fonction qui prenne en paramètre un tableau d'entiers de taille n lignes × m colonnes ainsi que l'entier n et l'entier m, et qui renvoie un tableau à 1 dimension de longueur (n×m) rempli avec les valeurs du tableau initial prises ligne par ligne.
Ecrire une première fois cette fonction avec deux boucles `Pour` imbriquées, puis une fois la même fonction avec une seule boucle `Pour`.

Exercice 6.27 [d]

a) Etant donné un échiquier sur lequel sont placés des pièces du jeu, on voudrait pouvoir dire s'il est possible de déplacer le roi blanc. Les rois peuvent se déplacer sur une des 8 cases avoisinantes, pour autant qu'elle existe et qu'elle ne soit pas déjà occupée ou soit occupée par une pièce adverse (dans ce dernier cas, le roi mange la pièce adverse et prend sa place; on ne se préoccupe pas des autres restrictions).
Ecrire une fonction qui prend en paramètres un tableau 8×8 d'entiers (0 = case vide, 1 = pièce blanche, 2 = pièce noire) ainsi que le numéro de ligne et celui de la colonne du roi blanc, et renvoie *vrai* si le roi peut se déplacer, *faux* sinon.

b) Une reine peut se déplacer dans les mêmes 8 directions que les rois, mais d'autant de cases qu'elle le veut, jusqu'à rencontrer le bord de l'échiquier, une pièce de la même couleur (elle s'arrête avant) ou une pièce adverse (elle peut manger cette pièce et prendre sa place). Si l'on désire cette fois savoir si la reine blanche peut être jouée, quelles sont les modifications à apporter à la fonction déjà écrite ?

c) Un fou se déplace comme une reine, mais seulement en diagonale. Pour savoir si un fou blanc donné peut être joué, quelles sont les modifications à apporter à la fonction ?

Exercice 6.28 [d]

Etant donné un échiquier sur lequel sont placés des pièces du jeu, on voudrait pouvoir dire s'il est possible pour la reine blanche de *manger une pièce adverse*. Une reine peut se déplacer horizontalement, verticalement ou selon les diagonales, d'autant de cases qu'elle le veut, jusqu'à rencontrer le bord de l'échiquier, une pièce de la même couleur (elle s'arrête avant) ou une pièce adverse (elle peut manger cette pièce et prendre sa place).
Ecrire une fonction qui prend en paramètres un tableau 8×8 d'entiers (0 = case vide, 1 = pièce blanche, 2 = pièce noire) ainsi que le numéro de ligne et celui de la colonne de la reine blanche, et renvoie vrai si la reine peut manger une pièce adverse, faux sinon.

Exercice 6.29 [d]

On possède un parallélépipède rectangle composé de m×n×o briques de LEGO. Certaines briques sont rouges, d'autres sont jaunes, vertes ou encore bleues. On désire compter le nombre de briques différentes.
Ecrire la procédure qui prend en paramètre 3 entiers m, n et o et un tableau m×n×o d'entiers, et qui affiche le nombre de briques de chaque couleur. On décide que, dans le tableau, 0 signifie rouge, 1 jaune, 2 vert et 3 bleu.

Exercice 6.30 [d]

On possède un cube composé de 5×5×5 = 125 briques de LEGO. Certaines briques sont rouges, d'autres sont jaunes, vertes ou encore bleues. On désire savoir si l'on peut voir une des briques rouges du cube, c'est-à-dire s'il en existe une qui soit sur le bord du cube.
Ecrire la fonction qui prend en paramètre un tableau 5×5×5 d'entiers et qui retourne VRAI s'il existe une brique rouge visible. On décide que, dans le tableau, 0 signifie rouge, 1 jaune, 2 vert et 3 bleu.

Exercice 6.31 [d]

Etant donné deux vecteurs de réels de taille identique n, écrire une fonction qui indique si l'un est un multiple de l'autre (par exemple, [3 6 -6] est multiple de [2 4 -4]). Par définition, un vecteur qui n'a que des valeurs nulles est toujours multiple d'un autre.

Série 7 : structures

Matière couverte

Conditions
Boucles
Tableaux
Procédures et fonctions
Structures

Exercices

Exercice 7.1 [f]

Proposer une structure permettant de stocker les caractéristiques physiques d'un chien.

Exercice 7.2 [f]

Proposer une structure permettant de stocker les caractéristiques d'un ordinateur.

Exercice 7.3 [f]

Soit une petite entreprise de peinture qui a 10 fournisseurs et 100 clients. Proposer les structures de données qui permettront de stocker les informations nécessaires à l'envoi du courrier leur étant destiné.

Exercice 7.4 [m]

Ecrire un algorithme qui enregistre les renseignements concernant les naissances de bébés dans une maternité : nom et prénom du bébé, nationalité, date de naissance, heure de naissance et poids.
L'algorithme devra ensuite :
- afficher le nom et le prénom des bébés qui pèsent moins de 2 kg.
- compter combien de bébés sont de nationalité suisse.
- afficher le nom de ceux qui sont nés au mois de mars.

Exercice 7.5 [m]

Ecrire un algorithme qui enregistre les renseignements concernant 100 voitures : numéro d'immatriculation, marque de la voiture, type de la voiture, puissance, nombre de places assises et couleur.
L'algorithme devra ensuite :
- afficher le numéro d'immatriculation des voitures rouges.
- compter combien de marques différentes ont été enregistrées.

Exercice 7.6 [m]

Soit une entreprise qui vend des verres en plastique et qui désire conserver les coordonnées de ses clients.
Proposez les structures de données nécessaires pour stocker pour chaque client :

- son numéro de client,
- son nom,
- son prénom,
- son adresse,
- la date de sa dernière commande.

Ecrire un algorithme qui, étant donné un numéro de client, supprime ce client.
Ecrire un algorithme qui, étant donné les renseignements nécessaires, ajoute un client.

Exercice 7.7 [m]

Soit une classe de 15 élèves qui suivent un cours de techniques de programmation. Pour chaque élève, on désire mémoriser son nom et son prénom ainsi que la liste des cinq notes qu'il a obtenues.
Proposer les structures nécessaires.
Pour chaque élève, on désire calculer et stocker sa moyenne. Quel changement cela implique-t-il ?
Ecrire un algorithme, qui étant donnée la structure initiale et son contenu (nom, prénom, notes), calcule la moyenne de chaque élève.
Ecrire un algorithme qui affiche le nom de l'élève ayant la moins bonne note, celui de l'élève ayant la meilleure note.

Exercice 7.8 [m]

Un botaniste désire comparer la taille moyenne de 10 espèces différentes de plantes. Il ramène un jour des échantillons de celles-ci, dont il désire stocker la taille afin de calculer la taille moyenne de chaque espèce.
Proposer une structure qui puisse stocker le nom de l'espèce et les mesures des échantillons correspondants. Il y a entre 0 et 100 échantillons de chaque espèce.
Ecrire la procédure qui prend en paramètre un tableau de taille 10, représentant les 10 espèces, et qui indique pour chacune d'elle son nom et la taille moyenne des échantillons.

Exercice 7.9 [d]

Soit une course de moto avec n coureurs. Chaque coureur possède un nom, un prénom et un temps de course en heures, minutes et secondes.
Proposer les structures nécessaires.

On a un tableau représentant les n coureurs, le premier ayant le dossard 1, le second le dossard 2, etc., c'est-à-dire que le numéro de dossard correspond à l'indice du tableau.
Pour chaque coureur, on veut pouvoir savoir facilement qui est arrivé juste après lui. On modifie donc la structure de sorte qu'elle comporte un champ `Suivant` indiquant le numéro de dossard du coureur lui succédant à l'arrivée. On décide par convention que le dernier coureur a comme valeur `-1` pour ce champ, puisque personne n'est arrivé après lui.

Ecrire une procédure qui prend en paramètres un entier n, un tableau de n coureurs et le numéro de dossard du premier arrivé, et qui affiche les noms des coureurs dans l'ordre d'arrivée en précisant leur place et leur temps de course. On suppose que les champs `Suivant` ont déjà été correctement remplis pour chaque coureur du tableau.

Série 8 : types énumérés

Matière couverte

Conditions
Boucles
Tableaux
Procédures et fonctions
Structures
Types énumérés

Exercices

Exercice 8.1 [f]

Ecrire les types utilisés pour décrire l'état civil d'une personne.

Exercice 8.2 [m]

Ecrire les types utilisés pour décrire les résultats d'un examen de maths d'une classe de 30 personnes. Les résultats possibles de cet examen sont : pas encore corrigé, réussi, échoué, triche de la part de l'étudiant, ou absence de l'étudiant.

Exercice 8.3 [m]

Ecrire les types utilisés pour décrire un stock d'ampoules, qui ont chacune une certaine puissance en watts, et qui peuvent être neuves ou utilisées, et être fonctionnelles, cassées ou non testées.

Exercice 8.4 [m]

Ecrire les types utilisés pour décrire les deux grilles d'une bataille navale, dont chacune des 12×12 cases peut être vide, occupée par un bateau et touchée ou occupée par un bateau mais pas encore découverte.

Exercice 8.5 [m]

Ecrire les types utilisés pour décrire 100 personnes, ayant chacune un nom, un état civil, un sexe, une taille, une couleur de cheveux, une langue (français, allemand, italien, anglais ou "autre") et un canton de résidence qui peut-être Vaud, Valais, Genève, Neuchâtel ou Fribourg.

Exercice 8.6 [d]

Ecrire un algorithme qui :

- prenne la description de 20 personnes (sexe et âge), stocke le tout avec en plus pour chaque personne la tranche d'âge correspondant à son âge ("bébé" pour les 2 ans et moins, "enfant" pour les 16 ans et moins, "adulte" sinon),
- demande ensuite à l'utilisateur une tranche d'âge et un sexe et calcule le nombre de personnes correspondant à ces 2 critères.

Exercice 8.7 [d]

Ecrire un algorithme qui lise la description de n meubles (genre, matériau et poids).
Un meuble peut être une chaise, une table, une étagère ou un canapé. Le meuble est en bois, en métal, ou en plastique.
L'utilisateur indique ensuite un matériau, et on indique alors le poids moyen pour chaque type de meuble de ce matériau.

Série 9 : découpage en sous-algorithmes

Matière couverte

Conditions
Boucles
Tableaux
Procédures et fonctions
Structures
Types énumérés

Exercices

Dans cette série, on n'écrit que l'algorithme principal (y compris les constantes et structures de données) et l'en-tête des procédures et fonctions. Certaines procédures ou fonctions peuvent parfois utiliser elles-mêmes d'autres procédures ou fonctions, qu'on indiquera dans la mesure du possible.

Exemple

Ecrire un algorithme qui demande 10 fois un mois et une année et indique le nombre de jours dans ce mois.

Solution :

```
Const NB_DEMANDES c'est 10


Procédure lireMoisAnnée(var mois, annee : entier)
        /* … on n'écrit pas le corps de la procédure */

/* cette fonction est utilisée par  calculerNbJours */
Fonction estBissextile(annee : entier) : booléen
        /* … on n'écrit pas le corps de la fonction */

Fonction calculerNbJours(mois, annee : entier) : entier
        /* … on n'écrit pas le corps de la fonction */

Procédure afficher(mois, annee, nbJours : entier)
        /* … on n'écrit pas le corps de la procédure */


/* algorithme principal */
Var mois, annee, nbJours, i : entier
Pour i allant de 1 à NB_DEMANDES Faire
        lireMoisAnnée(mois, annee)
        nbJours ← calculerNbJours(mois, annee)
        afficher(mois, annee, nbJours)
Fait
```

Exercice 9.1 [f]

Ecrire un algorithme qui demande au plus 1000 notes entières entre 1 et 100, affiche la moyenne et calcule combien il y a de 1, de 2, etc. puis affiche visuellement ce résultat, par exemple avec un astérisque par note :

```
  1  ***
  2  *
  3  ******
 …
100  ***
```

Exercice 9.2 [m]

Ecrire un algorithme qui lise la date du jour, puis les noms, prénoms, dates de naissance et dates d'engagement des employés d'une entreprise, puis qui indique :

- quel est l'employé le plus jeune,
- celui qui est dans l'entreprise depuis le plus longtemps,
- les gens qui ont leur anniversaire ce jour ou le lendemain.

Exercice 9.3 [m]

Ecrire un algorithme qui gère les fiches de salaires des employés d'une entreprise, une par employé et par mois sur une année.
Une fiche de salaire comporte les heures mensuelles, le salaire brut et le salaire net.
On doit pouvoir :

- ajouter un employé,
- supprimer un employé,
- modifier un employé,
- afficher la fiche de salaire d'un employé indiqué par son nom et prénom pour un mois donné,
- afficher le total des heures et des salaires bruts et nets pour un employé indiqué par son nom et prénom.

Exercice 9.4 [m]

Ecrire un algorithme qui permette de jouer au jeu du pendu. Le mot à deviner est entré au départ par une autre personne. On suppose que le joueur a le droit à 10 essais infructueux. On indique à chaque fois les lettres déjà utilisées, et le mot avec les lettres trouvées.

Exercice 9.5 [m]

Ecrire un algorithme qui permette de jouer aux tours de Hanoï. On demande à l'utilisateur le nombre de disques au départ (entre 1 et 10).
A chaque fois, on indique la position des disques et on demande au joueur quel déplacement il veut effectuer (piquet de départ et piquet d'arrivée). Si le déplacement est légitime (pas de disque au-dessus d'un disque plus petit), alors on affiche à nouveau la position des disques, sinon on affiche un message d'erreur. Le jeu s'arrête quand les disques sont tous passés du piquet 1 au piquet 2.

Exercice 9.6 [m]

Ecrire un algorithme qui demande le nom de 2 joueurs et leur fait faire une partie de tic-tac-toe (morpion). On vérifie que les coups sont valables (pas sur une case déjà occupée) et on arrête quand un joueur a gagné (on indique alors le nom du gagnant) ou que la grille est pleine (on indique alors qu'il y a match nul).
On affiche la grille de cette manière :

```
. O .
X . .
O . X
```

Exercice 9.7 [d]

Modifier l'algorithme précédent de sorte que l'on puisse aussi jouer contre l'ordinateur ou que l'ordinateur joue contre lui-même.

Exercice 9.8 [d]

Ecrire un algorithme qui permette de jouer au jeu du Rubik's Cube. C'est un dé à 6 faces, composé de 3×3×3=27 sous-cubes, qui divise chaque face en 9 facettes avec chacune leur couleur (rouge, vert, bleu, blanc, jaune ou orange). On peut faire tourner n'importe quelle face composée de 9 sous-cubes dans le sens des aiguilles d'une montre ou dans le sens contraire, ce qui réarrange les places des 9 facettes de la face en question ainsi que les 12 facettes sur le bord de cette face (3 facettes sur 4 côtés).

L'algorithme part du cube ayant chaque face d'une même couleur, puis fait 100 mouvements au hasard pour mélanger les facettes. Le algorithme affiche alors le cube, demande à l'utilisateur quel mouvement il veut faire (face à bouger, puis sens), effectue le mouvement et réaffiche le cube. On recommence, jusqu'à que toutes les faces soient de nouveau d'une même couleur, et on affiche alors combien de mouvements il a fallu à l'utilisateur pour résoudre le cube.