# UPJV - Amiens
# Licence Professionnelle SILDA
# Algorithmique
# Exercices

Gilles Dequen

## 1 Échauffements

**Exercice 1** *Premier algorithme*

Écrire un algorithme qui effectue la saisie d'un entier, affiche son carré puis son cube.

**Exercice 2** *Lire une séquence d'instructions*

Quelles seront les valeurs de A et B après la suite d'instructions suivante ?

```
Entier A, B;

A = 1;
B = A+1;
A = 3;
```

**Exercice 3** *Lire une séquence d'instructions*

Quelles seront les valeurs de A, B et C après la suite d'instructions suivante ?

```
Entier A, B, C;

A = 1;
B = 5;
C = A-B;
A = 2;
C = A+B;
```

**Exercice 4** *Lire une séquence d'instructions*

Quelles seront les valeurs de A, B et C après la suite d'instructions suivante ?

```
Entier A, B, C;

A = 2;
A = A+2;
B = A*2+A-2;
C = 4;
C = B-C;
```

```
C = C+A-B;
A = B-C*A;
A = (B-A)*C;
B = (A+C)*B;
```

**Exercice 5** *Calcul du discriminant*

Écrire un algorithme qui calcule le discriminant d'un polynôme du second degré $ax^2 + bx + c = 0$ et qui affiche sa valeur. a, b et c sont de type entier et sont saisis par l'utilisateur.

**Exercice 6** *Division entière*

Écrire un algorithme qui calcule le reste de la division entière d'un nombre A par un nombre B et affiche sa valeur. A et B sont saisis par l'utilisateur. Vous n'utiliserez pas l'opérateur % pour le calcul du reste.

**Exercice 7** *Lecture d'algorithmes et parenthésage*

Pour chaque algorithme, quelle est la valeur de la variable Résultat en supposant qu'on saisit, chaque fois, la valeur 10 pour la variable X ?

```
Programme Algorithme1 {
 Réel X, Résultat;
 X = Lire();
 Résultat = X-1/X*(20+X);
 Afficher(Résultat);
}

Programme Algorithme2 {
 Réel X, Résultat;
 X = Lire();
 Résultat = (X/2)+(X*(X-1));
 Afficher(Résultat);
}
```

**Exercice 8** *Manipulation de variables*

Écrire un algorithme permettant d'échanger la valeur de 2 variables A et B préalablement saisies par l'utilisateur. Est-il possible de faire cet échange sans passer par une variable intermédiaire ?

**Exercice 9** *Permutation circulaire*

Écrire un algorithme qui, après avoir permis la saisie de 3 variables réelles X, Y et Z, effectue une permutation circulaire de leurs valeurs respectives (i.e. Z = Y, X = Z, Y = X).

**Exercice 10** *Comprendre les principes de l'affectation*

On considère une variable entière X déjà lue au clavier. Combien faut-il d'instructions élémentaires (affectation et une seule opération) pour calculer $X^{16}$ sans jamais utiliser l'opérateur d'exponentiation ? Idem pour calculer $X^{10}$.

**Exercice 11** *Affectation*

Écrire un algorithme permettant de lire une valeur réelle X et de modifier X de telle sorte que l'on obtienne $X^3 + X^2 + X + 1$. Vous n'utiliserez que des instructions élémentaires (sans l'opérateur d'exponentiation).

**Exercice 12** *Premiers programmes*

Écrire un programme qui calcule et affiche le prix HT d'un article à partir du prix TTC et du pourcentage de TVA qui le concerne. Les variables TTC et TVA sont saisies au clavier par l'utilisateur.

**Exercice 13** *Premiers programmes*

Écrire un algorithme saisissant 3 variables entières puis qui calcule et affiche leur moyenne.

**Exercice 14** *Premiers programmes*

Soient M et N deux variables entières saisies au clavier contenant respectivement m et n. Ecrire une suite d'instructions permettant d'obtenir dans M la valeur $m - n$ et dans N la valeur $m + n$.

**Exercice 15** *Décomposition d'une somme d'argent*

Écrire un algorithme qui à partir d'une somme d'argent donnée, la décompose en un nombre minimal de billets de 20, 10 et 5 Euros et 50, 20, 10, 5, 2 et 1 centimes d'Euros. La somme initiale sera saisie par l'utilisateur.

**Exercice 16** *Premiers programmes*

Écrire un algorithme prenant un temps en secondes que l'on transcrira en jours, heures, minutes et secondes.

**Exercice 17** *Premiers programmes*

En se basant sur l'exercice précédent, ecrire un algorithme permettant de faire la différence entre 2 horaires exprimés en heures, minutes et secondes.

## 2 Instructions Conditionnelles

**Exercice 18** *Quelques notions de logique*

a) Quelle sera la valeur logique des expressions booléennes suivantes lorsque :
- $A = VRAI \ \ \&\& \ \ B = VRAI$
- $A = FAUX \ \ \&\& \ \ B = VRAI$
- $A = VRAI \ \ \&\& \ \ B = FAUX$
- $A = FAUX \ \ \&\& \ \ B = FAUX$

b) Simplifier les expressions booléennes suivantes :
- $A \ \ \&\& \ \ (A \ \ || \ \ B)$
- $A \ \ \&\& \ \ (!A \ \ || \ \ B)$
- $A \ \ || \ \ (!A \ \ \&\& \ \ B)$
- $A \ \ \&\& \ \ C \ \ \&\& \ \ (B \ \ || \ \ C)$

**Exercice 19** *Savoir interpréter une condition*

Donner les valeurs des variables A, B et C à la sortie de ce bloc d'instructions :

```
Si (C-B == B) {
 A = A + 1;
 C = C + B;
 B = A;
}
Sinon {
 B = A;
 A = A - 1;
 C = C * B;
}
```

a) pour $A = 2, B = 3, C = A * B$
b) pour $A = 1, B = 5, C = 3$
c) pour $A = -3, B = A * A, C = B - 5$
d) pour $A = 8, B = 3, C = A - 2$
e) pour $A = 10, B = 1, C = -B + A^2$

**Exercice 20** *Expression d'une condition*

X, Y, Z et T sont quatre variables numériques d'un environnement donné. Exprimez le prédicat relatif aux conditions décrites ci-dessous (exemple : "Les valeurs de X et Y sont supérieures à 3" aura pour réponse "$X > 3 \ \ \&\& \ \ Y > 3$").

a) Les valeurs de X, Y et Z sont identiques
b) Les valeurs de X, Y et Z sont identiques et différentes de T
c) La valeur de X est comprise (strictement) entre les valeurs de Y et de T et la valeur de Y est inférieure à celle de T
d) La valeur de X est comprise (strictement) entre les valeurs de Y et de T
e) Parmi les valeurs de X, Y et Z, 2 d'entres elles au moins sont identiques
f) Parmi les valeurs de X, Y et Z, 2 d'entres elles seulement sont identiques
g) Parmi les valeurs de X, Y et Z, 2 d'entres elles au plus sont identiques

**Exercice 21** *Mini-tri*

Écrire un algorithme qui a pour but de saisir 2 entiers et les afficher ordonnés dans l'ordre croissant.

**Exercice 22** *Mini-tri*

Écrire un algorithme effectuant la saisie de 3 nombres entiers. Vous concevrez un algorithme affichant ces 3 nombres dans l'ordre croissant. Vous utiliserez exclusivement des conditions élémentaires (i.e. ne faisant pas intervenir d'opérateurs logiques).

**Exercice 23** *Mini-tri*

Même question qu'à l'exercice précédent en faisant intervenir des opérateurs logiques && et ||.

**Exercice 24** *Polynôme du second degré*

Écrire un algorithme qui calcule le discriminant Delta ($b^2 - 4ac$) d'un polynôme du second degré ($ax^2 + bx + c = 0$) et qui en fonction de son signe, calcule la ou les racines réelles s'il y a lieu (a, b et c sont de type réel et sont saisies au clavier par l'utilisateur).

**Exercice 25** *Le temps plus une seconde*

Écrire un algorithme qui saisit un temps exprimé en jours, heures, minutes et secondes. À ce temps, vous ajouterez une seconde et vous l'afficherez en jours, heures, minutes et secondes. Il vous est interdit de passer par une phase de conversion en Secondes.

**Exercice 26** *Nombre de jours de congés*

Dans une entreprise, le calcul des jours de congés payés s'effectue de la manière suivante :

- Si une personne est entrée dans l'entreprise depuis moins d'un an, elle a droit à deux jours de congés par mois de présence, sinon 28 jours au moins.
- Si c'est un cadre et s'il est agé d'au moins 35 ans et si son ancienneté est supérieure à 3 ans, il lui est accordé 2 jours supplémentaires.
- S'il est agé d'au moins 45 ans et si son ancienneté est supérieure à 5 ans, il lui est accordé 4 jours supplémentaires.

Écrire un algorithme qui calcule le nombre de jours de congés à partir de l'âge, l'ancienneté et l'appartenance au collège cadre d'un employé.

# 3 Instructions répétitives

**Exercice 27** *Boucles, échauffements*

Afficher tous les entiers compris entre 1 et 10.

**Exercice 28** *Boucles, échauffements*

Pour 'N' saisi au clavier, afficher les N premiers nombres impairs positifs. Vous écrirez un algorithme utilisant une boucle définie ("Pour"), et également deux versions traitant le même problème utilisant respectivement une boucle indéfinie "Tant Que" et "Répéter ... Jusqu'à".

**Exercice 29** *Boucles, échauffements*

Écrire un algorithme qui lit au clavier un nombre entier n, puis qui affiche tous les nombres entiers positifs strictement inférieurs à n.

**Exercice 30** *Boucles, échauffements*

Écrire un algorithme qui calcule $X^n$ sans l'exponentiation.

**Exercice 31** *Boucles, échauffements*

Écrire un algorithme qui calcule A * B sans effectuer de multiplication.

**Exercice 32** *Boucles, échauffements*

Écrire un algorithme qui calcule A / B sans effectuer de division.

**Exercice 33** *Boucles, échauffements*

Écrire un algorithme qui compte le nombre de valeurs divisibles par 7 parmi les valeurs comprises entre 2 entiers M et N. M et N seront saisies au clavier.

**Exercice 34** *Conversions décimal ↔ binaire*

Écrire un algorithme qui affiche le nombre de digits à '1' d'un nombre entier (Base 10) lorqu'il est représenté sous son format binaire (Base 2).

**Exercice 35** *Conversions décimal ↔ binaire*

Écrire un algorithme qui affiche un nombre entier (Base 10) sous son format binaire (Base 2).

**Exercice 36** *Suite de Fibonacci*

Les nombres de Fibonacci sont caractérisés par la série suivante :
$Fib(0) = 0$
$Fib(1) = 1$
$Fib(n) = Fib(n-1) + Fib(n-2)$
Écrire un algorithme qui calcule et affiche le N-ième terme de cette suite. N sera saisi au clavier.

**Exercice 37** *Factorielle*

Réaliser une fonction retournant la factorielle de son paramètre.
Rappel :

$$n! = \prod_{i=1}^{n} i = 1 \times 2 \times \ldots \times (n-1) \times n$$

**Exercice 38** *PGCD*

On souhaite calculer le PGCD de 2 nombres entiers suivant la méthode d'Euclide.
Exemple pour 35 et 14 :
35 - 14 = 21 (on mémorise 21 et 14)
21 - 14 = 7 (on mémorise 14 et 7)
14 - 7 = 7 (on mémorise 7 et 7)
7 - 7 = 0 (on mémorise 7 et 0)
le PGCD est alors 7
Réaliser un algorithme réalisant ce calcul.

**Exercice 39** *Moyenne*

Écrire un algorithme qui permet de saisir au clavier des notes, calculer et afficher leur somme ainsi que leur moyenne. Les notes sont fournies au clavier avec un dialogue du type :
note 1 : 12
note 2 : 15.25
note 3 : 13.5
note 4 : 8.75
note 5 : -1
Le nombre de notes n'est pas connu à priori et l'utilisateur peut en fournir autant qu'il le souhaite. Pour signaler qu'il a terminé, on convient qu'il fournira une note fictive négative.

**Exercice 40** *Petit jeu*

On supppose qu'il existe une fonction Aléat() qui renvoie un nombre compris entre 0 et 1 (inclus). Écrire un algorithme qui permet à l'utilisateur de deviner un nombre choisi au hasard compris entre 100 et 2000.

**Exercice 41** $\pi$

$\pi$ peut être calculé à l'aide de la série suivante : $4(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \ldots)$ Écrire une fonction qui prend en paramètre le rang de calcul pour l'approximation de $\pi$ et qui retourne la valeur approchée au rang $n$ de $\pi$.

**Exercice 42** *Série harmonique*

Écrire un algorithme permettant de calculer la somme des $n$ premiers termes de la série harmonique, i.e. la somme $1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \ldots + \frac{1}{n}$.

**Exercice 43** $e$

Calculez la valeur du nombre $e$ en considérant

$$e = \sum_{n=0}^{+\infty} \frac{1}{n!}$$

**Exercice 44** *Nombre premier*

Écrire un algorithme qui, à partir d'un nombre entier strictement positif, affichera "VRAI" ou "FAUX" selon que le nombre est premier ou non. Pour mémoire, les nombres premiers inférieurs à 100 sont : 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97.

**Exercice 45** *Fonction mathématique et recherche du 0 par dichotomie*

Écrire un algorithme qui calcule le zéro d'une fonction $f(x)$ sur l'intervalle $[a, b]$, avec une précision $\varepsilon$. La fonction $f$ et les réels $a, b$ et $\varepsilon$ sont donnés. Soit $f(x)$ une fonction continue sur l'intervalle $[a, b]$, où elle ne s'annule qu'une seule et unique fois. Pour trouver ce zéro, on procède par dichotomie, c'est-à-dire que l'on divise l'intervalle de recherche en 2 à chaque étape. Soit $m$ le milieu de $[a, b]$. Si $f(m)$ et $f(a)$ sont de même signe, le zéro recherché est dans l'intervalle $[m, b]$ et dans l'intervalle $[a, m]$ sinon.

**Exercice 46** *Nombres parfaits*

Écrire un algorithme qui affiche la suite de tous les nombres parfaits inférieurs ou égaux à un nombre donné (saisi!) noté $n$. Un nombre est dit parfait s'il est égal à la somme de tous ses diviseurs stricts. Par exemple, 28 est parfait car 28=1+2+4+7+14. Pour information, la liste des nombres parfaits inférieurs à 10000 est : 6, 28, 496, 8128.

## 4 Tableaux

**Exercice 47** *Un premier tableau*

Écrire un algorithme qui remplit un tableau avec les 100 premiers entiers naturels. Ajoutez 1 à toutes les valeurs de rang pair de ce tableau. Soutrayez 1 à toutes les valeurs de rang impair. Affichez l'intégralité de ce tableau.

**Exercice 48** *Recherche d'élément dans un tableau*

On considère un tableau T contenant déjà des éléments de type entier.

a) Écrire un algorithme de recherche séquentielle d'un élément dans un tableau.

b) Supposons que le tableau initial est trié dans l'ordre croissant, l'algorithme de recherche d'un élément (si il existe) devra tirer partie de cette spécificité. Vous utiliserez une approche dichotomique.

**Exercice 49** *Insertion d'éléments*

On considère un tableau T préalablement trié. Écrire un algorithme permettant d'insérer un élément dans T. T doit bien entendu resté trié. On suppose qu'il existe une case vide à la fin du tableau.

**Exercice 50** *Fusion de tableaux*

On considère 2 tableaux de 100 éléments T1 et T2. Réaliser dans un tableau T3 une fusion de ces 2 tableaux. Réécrivez cet algorithme en prenant en compte cette fois la suppression des doublons.

**Exercice 51** *Fusion de tableaux triés*

On considère 2 tableaux de 100 éléments T1 et T2 triés dans l'ordre croissant. Réaliser dans un tableau T3 une fusion de ces 2 tableaux avec suppression des doublons. T3 doit bien entendu garder la caractéristique d'avoir ses éléments triés dans l'ordre croissant.

## 5 Fonctions et Procédures

**Exercice 52** *Lecture d'algorithmes*

On considère la fonction suivante :

```
Entier TOTO(entier A, entier B) {
 Entier C;
 Booléen S;

 S = FAUX;
 Si (A * B < 0)
  S = VRAI;
 C = 0;
 A = Valeur_Absolue(A);
```

```
 B = Valeur_Absolue(B);
 TantQue (A >= B) {
  C = C+1;
  A = A-B;
 }
 Si (S)
  C = -C;
 Retourne C;
}
```

a) Indiquez les différentes valeurs affichées si l'on trouve dans le programme les lignes suivantes :
- A = 25 ; B = 4 ; C = 3 ; Afficher(A, B, C, TOTO(14, 3)) ;
- A = 25 ; B = 4 ; C = 3 ; Afficher(A, B, C, TOTO(A, B)) ;
- A = 25 ; B = 4 ; C = 3 ; Afficher(A, B, C, TOTO(A, C)) ;
- A = 25 ; B = 3 ; C = 2 ; Afficher(TOTO(TOTO(A, B), C)) ;

b) Que fait cette fonction ?

**Exercice 53** *Calcul dans une matrice*

On considère une matrice 20x20, contenant des entiers. On s'intéresse à une case référencée par L et C désignant respectivement son numéro de ligne et son numéro de colonne. On souhaite calculer la somme des valeurs contenues dans les cases qui possèdent un coté ou un coin commun avec la case référencée par L et C.

a) Décrivez en langage naturel l'algorithme et les différents cas à gérer si l'on souhaite traiter ce problème dans une matrice normale.

b) Effectuer la même démarche en envisageant une matrice dont les bords se touchent (les bords haut et bas se touchent ainsi que les bords gauche et droit).

c) Écrire les algorithmes correspondants sous forme de fonctions.

**Exercice 54** *Opérations matricielles*

Soient 2 matrices carrées M1 et M2 de taille 5x5. On cherche à écrire une série de fonctions permettant d'additionner et multiplier 2 matrices.

a) Écrire une fonction de saisie de matrice 5x5.

b) Écrire les 2 fonctions effectuant l'addition et la multiplication de ces 2 matrices.

c) Modifier vos fonctions afin de pouvoir traiter des matrices de taille $T \times T$ ($2 \geq T > 100$) et écrivez le programme principal appelant ces fonctions.

**Exercice 55** *Max - Min*

Écrire un algorithme effectuant la saisie (aléatoire ou au clavier) de 100 nombres entiers (compris entre 0 et 1000) dans un tableau T et qui isole la valeur maximale existant dans T.

a) À partir de cet algorithme, imaginez un traitement permettant de trier dans l'ordre décroissant les nombres contenus dans T.

b) Adaptez ce traitement pour trier les nombres contenus dans T dans l'ordre croissant.

## 6 Divers

**Exercice 56** *Lecture d'algorithmes*

```
SansRésultat F(entier t (modifiable),
               entier T1[10] (modifiable), entier T2[10]) {
 Entier i, m;
```

```
 t = 1;
 m = 0;
 Pour i allant de 1 à 10 {
  Si (T2[i] > m) {
   m = T2[i];
   t = 1;
  }
  T1[t] = m;
  t = t+1;
 }
}
```

a) Faîtes tourner l'algorithme dans les 2 cas suivants :
   T2 : 2 4 9 0 8 9 7 3 1 4
   T2 : 7 3 10 9 10 9 8 10 10 3
b) Que fait cet algorithme ?

**Exercice 57** *Matrices creuses*

Les matrices ont une utilisation essentielle en calcul numérique. Leurs utilisations sont multiples notamment en astronomie, physique, calcul de trajet, ...Les matrices creuses ont une particularité qui les distingue des autres. Elles contiennent une très grande proportion de 0 (ce qui signifie que très peu de cases sont significatives). La matrice à considérer sera de taille $nlignes \times mcolonnes$.

a) Réalisez une compression de la matrice. Pour cela vous ignorerez les '0' et vous représenterez la matrice sous la forme de trois tableaux : Ligne, Colonne, Valeur pour toute valeur différente de 0.
b) Vous réaliserez la procédure de décompression.
c) Imaginez un algorithme permettant d'additionner 2 matrices sous forme compressée. Vous ferez de même pour l'opération de multiplication.

**Exercice 58** *Racine carrée*

Il est possible de calculer la racine carrée d'un nombre en utilisant un algorithme particulier sans avoir recours à une fonction mathématique Racine_Carrée prédéfinie. Soit n un nombre dont on veut extraire la racine carrée. On construit une suite de nombres $x_i$ dont le premier terme vaut 1 et dont le terme général a pour expression :

$$x_i = \frac{\frac{n}{x_{i-1}} + x_{i-1}}{2}.$$

Cette suite converge vers $\sqrt{n}$. Expliciter cet algorithme et le traduire sous forme d'une fonction prenant en paramètre n ainsi que le nombre d'itérations voulues.

**Exercice 59** *Tri à bulle*

On considère un tableau unidimensionnel de N nombres entiers (saisis par l'utilisateur). On se propose de trier les éléments de ce tableau dans l'ordre croissant en utilisant la méthode suivante : on balaye le tableau du début à la fin et on compare chaque élément avec son suivant. Si l'élément d'indice le plus élevé est le plus petit des 2, on échange la place de ces 2 valeurs dans le tableau. En répétant ce balayage un certain nombre de fois, le tableau sera trié. Écrire correctement cet algorithme en expliquant pourquoi il permet effectivement d'obtenir un tableau trié.

**Exercice 60** *Crible d'Ératosthène*

Le crible d'Ératosthène permet de déterminer les nombres premiers inférieurs à une certaine valeur N. On place dans un tableau unidimensionnel T les nombres entiers compris entre 1 et N. L'algorithme consiste, pour chaque élément T[i], à rechercher parmi tous les suivants (indice i+1 à N) ceux qui sont des multiples et les éliminer (par exemple les remplacer par des 0 ou -1). Lorsque tout le tableau a subi ce traitement, seul les nombres premiers n'ont pas été éliminés du tableau. Ecrire l'algorithme de ce crible (Pour N fixé).

**Exercice 61** *Nombres premiers*

Imaginer un algorithme permettant de stocker dans un tableau les N premiers nombres premiers sans utiliser la méthode du crible.

**Exercice 62** *Triangle de Pascal*

Le triangle de Pascal contient les valeurs $C_n^k = \frac{n!}{k! \times (n-k)!}$. Il correspond également aux coefficients du binôme de Newton, coefficients que l'on retrouve dans le développement $(a+b)^n$. On construit ce tableau à 2 dimensions en calculant chaque nouvelle ligne à partir de la précédente en utilisant l'expression $T_{i,j} = T_{i-1,j} + T_{i-1,j-1}$. Expliquer et écrire l'algorithme permettant le calcul du triangle de Pascal jusqu'à un niveau donné.

# 7 Récursivité

**Exercice 63** *Factorielle*

Réaliser une fonction récursive retournant la factorielle de son paramètre.
Rappel :

$$n! = \prod_{i=1}^{n} i = 1 \times 2 \times \ldots \times (n-1) \times n$$

**Exercice 64** *Conversion décimal ↔ binaire*

Écrire un algorithme qui convertit un nombre entier (Base 10) en son format binaire (Base 2). Vous réaliserez aussi l'opération inverse. Vous utiliserez une approche récursive pour les deux cas.

**Exercice 65** *Suite de Fibonacci*

Les nombres de Fibonacci sont caractérisés par la série suivante :
$Fib(0) = 0$
$Fib(1) = 1$
$Fib(n) = Fib(n-1) + Fib(n-2)$
Écrire une fonction récursive réalisant ce calcul.

**Exercice 66** *PGCD*

On souhaite calculer le PGCD de 2 nombres entiers suivant la méthode d'Euclide.
Exemple pour 35 et 14 :
35 - 14 = 21 (on mémorise 21 et 14)
21 - 14 = 7 (on mémorise 14 et 7)
14 - 7 = 7 (on mémorise 7 et 7)
7 - 7 = 0 (on mémorise 7 et 0)
Le PGCD est alors 7.
Vous utiliserez une approche récursive.