# *Travaux dirigés M1207*

R&T

## *Exercice 1 : Algorithme mystère*

Dessiner l'ordinogramme puis écrire la trace d'exécution avec les entrées 6, 5 et 4 de l'algorithme mystère et en déduire ce qu'il fait.

```
Nom : mystère
Rôle : à deviner
Entrée : a,b,c: entier
Sortie : a,b,c: entier
Déclaration: d: entier
Début
    if a>b then
            d = a
            a = b
            b = d
    end
    if b>c then
            d = c
            c = b
            b = d
    end
    if a>b then
            d = b
            b = a
            a = d
    end
Fin
```

## *Exercice 2 : Algorithme*

Dessiner l'ordinogramme de l'algorithme suivant, puis effectuer sa trace d'exécution avec l'entrée 4. Que calcule cet algorithme ?

```
Nom : boucle
Rôle : inconnu
Entrée : x: entier
```

Sortie : s: entier
Déclaration: n: entier, b: boolean
Début

```
    n = x
    s = 0
    b = false
    while b or n > 0 do
                s = s+x
                n = n-1
                print(n)
    end
```

Fin

## *Exercice 3 : Expressions*

Calculer le type et la valeur des expressions suivantes (on réalisera des arbres d'évaluation pour chaque expression).

Préciser les décisions de typage et les conventions de priorité utilisées.

- `2 + 3 * 4`
- `2 or 3`
- `2.0 + 3 * 4`
- `3 / 2 / 2`
- `true and (false or true)`
- `(2<3) and (4>5)`
- `2 < 3 <= 4`

## *Exercice 4 : Expressions mathématiques*

Traduire les expressions mathématiques suivantes en pseudo-code algorithmique :

- $z = \frac{\frac{2a-3}{2}+4}{d}$
- $w = b^2 - 4ac$
- $\phi^2 = \phi + 1$

## *Exercice 5 : Moyenne de trois notes*

1. Écrire un algorithme qui, à partir de trois notes et trois coefficients calcule la moyenne pondérée.
2. Réaliser une trace d'exécution avec les notes 9, 12, et 7 assorties des coefficients 1, 1 et 2.
3. Réaliser une trace d'exécution avec les notes 9, 20, et 1 et les coefficients 1, 1 et -2.
4. Réaliser une trace d'exécution avec les notes 10, 10, et 10 et les coefficients 0, 0 et 0.

## Exercice 6 : Instructions

Les bouts de code suivants sont des instructions. Identifier leurs composants (variables, opérateurs, noms de fonctions...) et si cela a du sens les types de ces composants.

- `s = a%4 == 0`
- `b = x+3 < y/2`
- `print("a or b")`
- `print(a or b)`

## Exercice 7 : Le temps plus une seconde

Écrire un algorithme qui prend en entrée un horaire donné sous la forme `heure`, `minute`, `seconde` et retourne l'heure qu'il sera une seconde plus tard.

## Exercice 8 : Équivalence de blocs

Pour chacun des cas suivants, les deux schémas sont-ils sémantiquement équivalents ?

1. 

```
if condition == vrai then
   actionsA
else
   actionsB
end
```

```
if condition then
   actionsA
else
   actionsB
end
```

2. 

```
if cond then
   resultat = expBoolA
else
   resultat = expBoolB
end
```

```
resultat = (cond and expBoolA) or
           ((not cond) and expBoolB)
```

3. 

```
if cond then
   actions1
else
   actions2
end
```

```
if (not cond) then
   actions2
else
   actions1
end
```

4. 

```
if condition1 then
   actions1
elseif condition2 then
   actions2
else

end
```

```
if condition1 then actions1 end
if condition2 then actions2 end
```

## *Exercice 9 : Année bissextile*

Écrire un algorithme qui teste si une année est bissextile. On rappelle qu'une année bissextile comporte 366 jours au lieu de 365 pour les années non bissextile. On rappelle également qu'une année est bissextile si elle est divisible par 4 sauf les années de fin de siècle qui ne sont pas divisibles par 400. Par exemple, 2000, 2008, 1896 sont bissextiles mais 1900, 2100, 2011 ne le sont pas.

En utilisant les équivalences de l'exercice 4 (celles qui sont vraies), simplifier l'algorithme.

## *Exercice 10 : Mois de 30 jours*

Écrire un algorithme qui prend en entrée le numéro du mois et dit s'il possède 30 jours. Faire plusieurs versions en utilisant divers schémas.

## *Exercice 11 : Division euclidienne*

Écrire un algorithme prenant en entrée deux entiers $a$ et $b$ et donne en sortie le quotient et le reste de la division euclidienne de $a$ par $b$.

Réaliser un ordinogramme de cet algorithme.

Réaliser une trace d'exécution de cet algorithme pour 14 et 4.

## *Exercice 12 : Conversion décimal-binaire*

Écrire un algorithme prenant en entrée un nombre décimal et donnant en sortie sa représentation en binaire.

L'algorithme devra utiliser la méthode des divisions successives. On peut supposer que le quotient de la division euclidienne de `a` par `b` est donné par l'expression `quotient(a, b)`.

On pourra utiliser une chaîne de caractères 0 et 1 pour stocker le nombre binaire.

## *Exercice 13 : Décomposition d'une somme d'argent*

Écrire un algorithme prenant en entrée une somme d'argent et donnant en sortie le nombre de pièces et billets de 1, 2, 5 et 10 euros constituant la décomposition de cette somme utilisant le mois possible de pièces et billets. On n'utilisera pas la division pour cet exercice.

Réaliser un ordinogramme de cet algorithme.

Réaliser une trace d'exécution de cet algorithme pour 19 euros.

Cet algorithme fonctionne-t-il avec les dollars américains (pièces de 1, 10, 25 cents) ? L'appliquer à la somme de 30 cents.

## *Exercice 14 : Racine carrée*

Écrire un algorithme calculant la partie entière de la racine carrée d'un nombre entier, c'est-à-dire le plus grand entier inférieure ou égal à la racine carrée.

Réaliser un ordinogramme de cet algorithme.

Réaliser une trace d'exécution de cet algorithme pour l'entrée 13.

## *Exercice 15 : Factorielle*

Écrire un algorithme calculant la factorielle d'un nombre entier. On rappelle que

$$n! = n \times (n-1) \times (n-2) \times \cdots \times 2 \times 1.$$

Réaliser un ordinogramme de cet algorithme.

## *Exercice 16 : Puissance*

Écrire un algorithme qui pour les entrées $x$ et $n$ retourne la valeur de $x^n$.
Réaliser la trace d'exécution de cet algorithme pour les valeurs 7 et 3.

## *Exercice 17 : Intégration*

Écrire un algorithme qui prend en entrée des réels $a$ et $b$ avec $a < b$ et un entier $n$ et qui calcule une approximation de l'intégrale d'une fonction $f$ donnée entre $a$ et $b$ par la méthode des trapèzes en découpant l'intervalle $[a, b]$ en $n$ parties.

On rappelle que l'aire d'un trapèze rectangle de grandes bases et petites bases $b$ et $B$ et de hauteur $h$ est donnée par la formule

$$\mathcal{A} = \frac{(B+b)h}{2}.$$

Réaliser l'ordinogramme de cet algorithme.
On pourra supposer que la fonction $f$ est définie comme suit :

```
function f(x)
    -- calcule un polynome
    -- entrée : x : réel
    -- sortie : resultat : réel
    resultat = 1 + x * (2 + x * 3)
    return resultat
end
```

## *Exercice 18 : Nombres premiers*

Un nombre premier est un nombre ayant exactement 2 diviseurs (1 et lui-même). Écrire un algorithme testant si un nombre donné en entrée est ou non premier.
Réaliser un ordinogramme de cet algorithme.
Réaliser une trace d'exécution pour la valeur 15.

## *Exercice 19 : Suite de Fibonacci*

La suite de Fibonacci est définie par récurrence par

$$\begin{cases} u_0 &= 1 \\ u_1 &= 1 \\ u_{n+2} &= u_n + u_{n+1} \end{cases}$$

Écrire un algorithme donnant le $n$-ième terme de la suite de Fibonacci.
Écrire un algorithme donnant le tableau des $n$ premiers termes de la suite de Fibonacci.
Écrire un algorithme donnant le tableau de tous les termes de la suite inférieurs à $n$.

## Exercice 20 : Expressions booléennes

Soient `x`, `y`, `z` et `t` quatre variables numériques. Traduire en expressions booléennes les phrases suivantes :

- les valeurs de `x` et de `y` sont toutes les deux supérieures à 3 ;
- les valeurs de `x`, `y` et `z` sont identiques mais différentes de celle de `t` ;
- la valeur de `x` est strictement comprise entre les valeurs de `y` et `t` ;
- la valeur de `x` est strictement comprise entre les valeurs de `y` et de `t` et la valeur de `y` est inférieure à celle de `t` ;
- parmi les valeurs de `x`, `y` et `z`, deux valeurs au moins sont identiques ;
- parmi les valeurs de `x`, `y` et `z`, deux valeurs exactement sont identiques ;
- parmi les valeurs de `x`, `y` et `z`, deux valeurs au plus sont identiques ;

## Exercice 21 : Représentation des entiers

On supposera dans cet exercice que les nombres entiers signés sont représentés sur 32 bits. Les entiers signés peuvent être positifs ou négatifs, il y a autant d'entiers signés positifs ou nul que d'entiers signés strictement négatifs.

- Combien de nombres entiers différents peut-on représenter ?
- Que vaut le plus grand entier signé (nommé `MAX_INT`) ?
- Que vaut le plus petit entier signé (nommé `MIN_INT`) ?
- Que se passe-t-il si on ajoute 1 à `MAX_INT` ?

## Exercice 22 : Représentation des réels

Les nombres réels sont en général représentés sur 32 bits (format simple précision) à l'aide d'1 bit de signe, 8 bits d'exposant et 23 bits de mantisse. Pour cet exercice, on se contentera d'un format 16 bits (1 bit de signe, 4 bits d'exposant et 11 bits de mantisse). Dans ce format, si l'on nomme $s$ le signe, $e$ l'exposant et $m$ la mantisse le nombre réel représenté vaut

$$(-1)^s 1, m \times 2^{e-7}$$

- Calculer la représentation en 16 bits des nombres `0.3`, `0.6` et `0.9`.
- Calculer la somme des représentations de `0.3` et de `0.6`.
- Quelle est la valeur de l'expression `0.3 + 0.6 == 0.9` ?
- Quel est le plus petit réel positif que l'on peut représenter dans ce format ?
- Quel est le plus grand réel positif que l'on peut représenter dans ce format ?