

Travaux dirigés

Notation en virgule flottante

Mathématiques pour l'informatique
—IMAC 2—

► Exercice 1. *Rappel sur les entiers*

1. Écrire en base 10 les nombres binaires suivants :

(a) 01011

(b) 10110

Que constatez-vous ?

2. Faites les opérations suivantes :

(a) $01010 + 10011$

(b) $01011 + 10110$

Vérifier en base 10.

► Exercice 2. *Un peu de code*

Écrire en langage C++ une fonction permettant d'afficher le contenu binaire d'une variable du type `unsigned char`. Vous pouvez charger le template `printUchar.cpp` sur le site de l'enseignant.

► Exercice 3. *Nombres flottants*

1. Écrire en base 10 les nombres à virgule flottante (IEEE 754) suivants :

(a) 1 10000010 010000000000000000000000

(b) 0 01111110 000000000000000000000000

2. Télécharger puis compiler le programme `seeFloat.cpp` sur la page de l'enseignant. En déduire la notation en virgule flottante (IEEE 754) des nombres suivants :

(a) 22

(b) 22.5

(c) *inf*

► **Exercice 4. Calculs faciles**

Récupérer le programme "caluclsFaciles.cpp" ou bien recopier les lignes suivantes sur votre éditeur puis compiler. Que constatez-vous ? Quelle est votre explication ?

```
#include <iostream>
#include <iomanip>

int main()
{
    std::cout << "0.1 + 0.2    = " << 0.1 + 0.2 << std::endl;
    std::cout << std::setprecision(20) << "0.1f + 0.2f = " << 0.1f + 0.2f << std::endl;
    std::cout << std::setprecision(20) << "0.1  + 0.2  = " << 0.1  + 0.2  << std::endl;
    std::cout << std::setprecision(20) << "0.1L + 0.2L = " << 0.1L + 0.2L << std::endl;

    std::cout << std::endl;
    std::cout << std::setprecision(20) << "0.1f - 0.1L = " << 0.1f - 0.1L << std::endl;

    return 0;
}
```

► **Exercice 5. Et en C/C++ ?**

1. Qu'est-ce que je risque en écrivant dans mon programme les lignes suivantes :

```
float a = ... ;
float b = ... ;

if(a == b) ... ;
```

Qu'est-ce que je devrais écrire à la place ?

2. Suis-je en danger si j'écris :

```
int a = ... ;
float b = a/3;
```

3. Mon programme compile-t-il avec la ligne suivante :

```
float a = -5.0/0;
std::cout << " a " << a << std::endl;
```

4. Est-ce que je peux mourir si ma vie dépend de la fiabilité de ce calcul ⁽¹⁾ :

⁽¹⁾ question de J. Chaussard

```
float a = powf(2.0,40);
std::cout << " a = " << a << std::endl;
```

```
int b = a;
std::cout << " b = " << b << std::endl;
```

5. Je vais manger dès que la boucle est finie, qu'est-ce qu'on mange ?

```
for(float a=0.0; a<1000000000; a+=0.00000001)
    ... ;

std::cout << " à table ! " << std::endl;
```

► Exercice 6. Suites

Soit la suite définie par :

$$\begin{cases} u_0 &= \frac{1}{3} \\ u_{n+1} &= 4u_n - 1 \end{cases}$$

1. calculer manuellement les 5 premiers termes.
2. faites un programme C++ calculant les 100 premiers termes avec des `float` et les 550 premiers termes avec des `double`.

► Exercice 7. Multiplication à la russe

La multiplication à la russe $a \times b$ consiste à répéter l'opération consistant à diviser a par 2 et à multiplier b par 2 jusqu'à ce que $a = 1$, dans quel cas $a \times b = b$. Deux cas sont à considérer dans ce processus itératif : si a est pair, tout se passe bien, si a est impair, sa division par 2 génère un résidu qu'il faudra additionner au résultat final. Cette approche était utilisée sur les premières calculatrices.

Par exemple 13×320 :

opération	a	b	résidu
	13	320	
13/2=6 reste 1	6	640	320
6/2=3 reste 0	3	1280	-
3/2=1 reste 1	1	2560	1280

résultat : $13 \times 320 = 2560 + (320 + 1280) = 4160$

Il apparait clairement qu'il est préférable de choisir pour a la plus petite des deux valeurs à multiplier.

Coder la multiplication à la russe en C++.

► **Exercice 8. Série de Bâle**

Il s'agit de sommer numériquement la série suivante :

$$u = \sum_{i=1}^{\infty} \frac{1}{i^2} = \frac{\pi^2}{6}$$

1. Calculez cette somme en utilisant des `float`, en commençant la somme par $i = 1$.

$$u \simeq \sum_{i=1}^{i_{\max}} \frac{1}{i^2}$$

2. Faites le même calcul par valeurs décroissantes en commençant la somme par i_{\max} . Que constate-t-on?

► **Exercice 9. Écart type**

Il est possible de calculer l'écart type d'un ensemble de données par la formule suivante :

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N \left(x_i - \frac{1}{N} \sum_{j=1}^N x_j \right)^2$$

En développant le terme au carré, on trouve une formule équivalente :

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N x_i^2 - \frac{1}{N^2} \left(\sum_{i=1}^N x_i \right)^2$$

Implémenter ces 2 méthodes en utilisant des `float` afin de voir à partir de combien de digit les résultats obtenus diffèrent.

► **Exercice 10. Entier positifs et négatifs**

Un entier signé `a` s'exprime sous la forme `a = s.m` où `s` est un bit de signe et `m` la magnitude de `a`. L'évaluation de `a` se fait sous la forme `a = m - s × 231`.

Exemples :

- `a = 5` : `0.0000000 00000000 00000000 00000101` soit `5 - 0 × 231`
- `a = -5` : `1.1111111 11111111 11111111 1111011` soit `(231 - 5) - 1 × 231`

Montrer que pour un `int a`, on peut calculer la valeur `-a` par la formule :

$$-a = \bar{a} + 1$$

où \bar{a} est le complément à 2 de `a` (on remplace les 1 par des 0 et réciproquement).



► Exercice 11. Opérateur modulo

L'opération $u \bmod m$ avec $m = 2^k$ (où k est un entier positif) est souvent utilisée en analyse numérique. Le but de l'exercice est de montrer que pour ce calcul, il suffit de ne garder que les k premiers bits de poids faible de u (les bits de poids faibles sont les bits correspondant aux plus petites puissances de 2).

1. Que fait l'opération $a \ll 1$ sur un entier non signé a ?
2. Que fait l'opération $a \gg 1$ sur un entier non signé a ?
3. Que font les opérations suivantes sur un entier non signé a ?
 - $a \ll k$
 - $a \gg k$
4. Que fait l'opération $(a \gg k) \ll k$ sur un entier non signé a ?
5. Montrer que l'opération $a - ((a \gg k) \ll k)$ sur un entier non signé a correspond à l'opération $a \bmod 2^k$. En déduire que pour calculer $a \bmod 2^k$, il suffit juste de garder les k 1er bits de poids faibles de a .

► Exercice 12. Racine carrée

1. Imaginez un programme estimant la racine carrée d'un nombre à virgule flottante.
2. Allez voir sur [wikipedia](https://fr.wikipedia.org/wiki/Racine_carr%C3%A9e).