Calcul flottant : exceptions et erreurs d'arrondi

Le chargé de TD vous indiquera en début de séance où trouver les programmes utilisés dans ce TP.

Exceptions

Exercice 1

1. Expliquer les valeurs obtenues pour les opérations $\sqrt{-1}$, zero/zero, un/zero et big $\times$ big à l'exécution du programme test1-exceptions (mode IEEE sans récupération d'erreur, big vaut 1e+300).

2. Pour la norme IEEE, à quelles opérations correspondent les 2 opérations invalides et le dépassement de capacité (*overflow*) signalés à la suite des calculs (mode IEEE avec récupération d'erreur) ?

3. Quelles exceptions non signalées interviennent dans les calculs ?

4. Quel est l'intérêt du *trap* utilisé dans la troisième partie du programme (mode IEEE avec *trap* des erreurs) ?

Exceptions de la norme IEEE : les exceptions interviennent lorsque le résultat d'une opération flottante est indésirable ou incertain. Elles peuvent être ignorées, dans ce cas une action par défaut est exécutée comme renvoyer une valeur spéciale. Quand un *trap* est activé pour une exception, une erreur est signalée lors de cette exception. Voici la liste des exceptions flottantes :

- *underflow* : le résultat est trop petit pour être représenté par un flottant normalisé. Il sera soit représenté par un flottant dénormalisé, soit par zéro.

- *overflow* : le résultat est trop grand pour être représenté par un flottant. Il sera représenté par l'infini.

- *inexact* : cette exception intervient quand le résultat d'un calcul flottant est inexact, i.e. quand le résultat est arrondi. Le résultat arrondi est la valeur finale obtenue.

- *invalid* : cette exception intervient quand un calcul invalide est demandé. Le résultat est NaN (Not a Number).

- *divide-by-zero* : cette exception intervient quand un flottant est divisé par zéro. La valeur infini est renvoyée.

Exercice 2

1. Quelle est la limite de $f(x) = \frac{1+x^2}{\sqrt{1+x^3}}$ quand x tend vers $+\infty$?

2. Exécuter le programme test2-exceptions. Expliquer pourquoi $f(x)$ calculée en simple précision donne 0 quand x est supérieur à 1e+13 et inférieur à 1e+18.

Modes d'arrondi

Exercice 3
Les programmes test1 à test4 calculent les trois valeurs suivantes (big = 1e+250) :

- $(\frac{1}{big} \times big)^{100}$
- $\sqrt{\frac{1}{2}}$
- $sin(-\pi)$

Ces programmes utilisent tous un mode d'arrondi différent *nearest*, *positive-infinity*, *negative-infinity* ou *zero*. Déterminer quel mode d'arrondi utilise chaque programme en fonction des résultats obtenus pour les trois calculs précédents.

Mode d'arrondi de la norme IEEE :

- *nearest* : le mode par défaut, le résultat est arrondi à la valeur la plus proche. Arrondi au plus près.
- *positive-infinity* : dans ce mode, le résultat est arrondi à la valeur la plus proche par valeur supérieure. Arrondi par excès.
- *negative-infinity* : dans ce mode, le résultat est arrondi à la valeur la plus proche par valeur inférieure. Arrondi par défaut.
- *zero* : dans ce mode, le résultat est arrondi à la valeur la plus proche de zéro (la plus petite en valeur absolue). Troncature.

Fiabilité des calculs

Arithmétiques non correctes

Exercice 4
Connaissez-vous un exemple d'arithmétique non correcte ?

Erreurs d'arrondi

Un cas d'école

Exercice 5
Considérer le programme (programme1.c) qui calcule la suite :

$$u_0 = 2$$

$$u_1 = -4$$

$$u_{n+1} = 111 - \frac{1130}{u_n} + \frac{3000}{u_n \times u_{n-1}}$$

1. Exécuter ce programme avec différentes valeurs de n. La suite converge-t-elle ? Si oui, quelle est sa limite ?
2. Démontrer que u_n peut s'exprimer sous la forme $u_n = \frac{4 \times 5^{n+1} - 3 \times 6^{n+1}}{4 \times 5^n - 3 \times 6^n}$. En déduire sa limite.

Les principales causes d'erreurs : absorption et élimination

Exercice 6

Considérer un premier programme qui calcule : $\sum_{i=1}^{n} \frac{1}{i}$ (programme2.c).

1. Exécuter le programme 2 et remplir le tableau suivant :

n	1 000	10 000	100 000	1 M	10 M	100 M
somme						

2. La suite converge-t-elle ? Si oui, quelle est sa limite ?

   Considérer ensuite un deuxième programme (programme3.c).

3. Quelle différence entre les programmes 2 et 3 ?
4. Exécuter le programme 3 et remplir le tableau suivant :

n	1 000	10 000	100 000	1 M	10 M	100 M
somme						

5. La suite converge-t-elle ? Si oui, quelle est sa limite ?
6. Essayer d'expliquer ce phénomène.

Exercice 7

Considérer un programme qui calcule l'exponentielle d'un réel x positif e^{-x} (programme4.c) par la série $\sum_{i=0}^{\infty}(-1)^n(\frac{x^n}{n!})$.

1. Exécuter le programme 4 pour calculer l'exponentielle e^{-20} avec différentes valeurs de n.
2. Quelle est la valeur a priori de l'exponentielle e^{-20} ?
3. Quelle est la valeur de n pour laquelle la série est la plus grande (en valeur absolue) ; quelle est cette valeur ?
4. Convertir cette valeur en hexadécimal. Pour cela, vous pouvez utiliser les convertisseurs décimal/binaire/hexadécimal que vous trouverez sur la page : http://babbage.cs.qc.edu/courses/cs341/IEEE-754.html.
5. Mesurer l'écart entre cette valeur et la valeur suivante en hexadécimal. Convertisser en décimal.
6. En réalité, le résultat exact de e^{-20} est 2.06115...10^{-9}. Essayer d'expliquer ce phénomène.

Une plus grande précision

Exercice 8

Reprendre les différents programmes (programmes 1, 2 et 4) utilisés dans la section *Fiabilité des calculs* et remplacer les *float* par des *double* et les %f dans les *printf* par des %lf. Comparer les résultats.