Architecture des ordinateurs
L3 Informatique 2020-2021
TP 1 - calcul Binôme :

- Chaque binôme rendra une feuille d'énoncé complétée et la déposera sur *e-learning* avec les programmes écrits pour répondre aux questions.
- Le nombre d'étoiles indique la difficulté de l'exercice :
  ★ = facile, ★★ = moyen, ★★★ = difficile.
- Ce TP a pour objectifs de :
  a. comprendre la manière dont les nombres flottants sont codés sur machine ;
  b. comprendre les erreurs que ce codage peut provoquer dans les comparaisons == et =< entre nombres flottant ;
  c. aborder des méthodes pour analyser ces erreurs.

Exercice 1. ★ L'égalité des flottants : $x = x$ ou $x \neq x$?

On s'intéresse ici au test d'égalité sur les flottants.

a. Écrire un programme C qui, pour chaque paire d'entiers (i, j) ci dessous, initialise les trois variables `float` (en C) à

```
a = i/10.0,   b = j/10.0,   c = (i+j)/10.0
```

et affiche si le résultat du test `a+b == c` est vrai. Reportez vos observations ci-dessous

(1,4) : (3,4) :

(1,6) : (3,5) :

(1,7) : (3,7) :

(1,8) : (3,8) :

b. Pour comprendre, faire afficher les valeur de `a+b` et `c` (utiliser une précision suffisante).

c. Peut-on en déduire dans quel(s) cas on peut utiliser l'égalité entre flottants ?

Exercice 2. ★ Accumulation d'erreurs

On s'intéresse aux sommes $\sum_{i=1}^{k} \frac{1}{i}$ et $\sum_{i=k}^{1} \frac{1}{i}$ (attention, c'est l'ordre des calculs qui est important).

a. Calculer ces deux sommes en **utilisant des `float`** et comparer les résultats (on prendra $k = 10^3$, $k = 10^6$ ou $k = 10^8$ par exemple) :

b. À votre avis, quel est le résultat le plus proche de la valeur correcte (vous pouvez vérifier ici https://www.wolframalpha.com/input/?i=sum, ou en faisant le calcul en `double`) :

c. Proposer un explication à l'observation précédente :

Indice : examiner la première et la dernière addition effectuée dans chaque cas.

Exercice 3. ★★ Décodage/encodage de nombres flottants.

On s'intéresse à l'encodage des nombres flottants en C. On pourra se référer à la documentation *754-2008 - IEEE Standard for Floating-Point Arithmetic*, disponible sur *e-learning*, ou procéder expérimentalement.

a. Quelle est la taille mémoire d'un `float`, et quelles tailles font sa mantisse et son exposant ?

b. Donner, en hexadécimal, l'encodage du flottant qui approxime 0.3.

c. Donner la mantisse, l'exposant et la valeur (en notation scientifique décimale) du `float` codé par 0x 3EA0 0000.

d. Vérifiez vos réponses avec https://www.h-schmidt.net/FloatConverter/IEEE754.html

Exercice 4. ★★ Que détecte réellement l'égalité entre flottants ?

On souhaite étudier le taux de réponses "correctes" du test d'égalité de l'exercice précédent.

a. Pour chaque paire (i, j) d'entiers entre 1 et 9, initialisez trois variables `float` (en C) à

```
a = i/10.0,    b = j/10.0,    c = (i+j)/10.0
```

et calculez le pourcentage (approximatif) de cas où `a+b == c` est vrai. Remplissez la première case du tableau ci-dessous.

b. Procéder de même en remplaçant 10.0 par k pour chacune des lignes du tableau, en examinant toutes les paires (i, j) avec $1 \leq i \leq k-1$ et $1 \leq j \leq k-1$. (On continue à utiliser des `float` en C.) Remplissez ainsi la première colonne du tableau.

	`float` C	`double` C	Python
question a		-	-
k=20			
k=32			
k=100			

c. Procédez de même pour les types `double` du C et le type de nombres flottants en Python. Remplissez ainsi les deuxièmes et troisièmes colonnes du tableau.

d. **Quels enseignements (au moins 3) peut-on en tirer ?**

Exercice 5. ★★ Qu'en est-il de l'inégalité ?

a. On va tester de la même manière le comportement du test d'inégalité large ($\leq$) entre flottants. Pour chaque paire (i, j) qui échoue lors du test de l'exercice précédent, commencer par déterminer lequel de $a + b$ et de c est le plus grand, puis ajouter une petite quantité (par exemple 10^{-6}) à la fois à $a + b$ et à c et vérifier que l'ordre n'a pas changé. Donner dans chaque cas le pourcentage de cas (parmi ceux où $a + b \neq c$) où l'ordre reste le même.

	`float` C	`double` C	Python
k=10			
k=20			
k=100			

b. Dans la question précédente, remplacer 10^{-6} par un nombre plus petit pour voir si cela a de l'influence. Que pouvez-vous en dire ?

c. **Comment peut-on tirer profit de ce que vous venez d'observer ?**

Exercice 6. ★★ Tester approximativement l'égalité.

On définit l'**écart absolu** entre deux nombres x et y comme la valeur absolue de leur différnce :

$$|x - y|,$$

et leur **écart relatif** comme leur écart absolu divisée par le minimum de leurs valeurs absolues :

$$\frac{|x - y|}{min(|x|, |y|)}.$$

a. Choisir un triplet i,j,k pour lequel le test d'égalité de l'exercice 4 échoue, et afficher l'écart absolu entre $a + b$ et c.

```
k =              i=              j=              dif_abs=
```

b. Reprendre l'exercice 4a en calculant, pour chaquev entrée du tableau, le maximum de l'écart relatif entre $a + b$ et c, en `C`.

k	dif_abs - `float`	dif_rel - `float`	if_abs - `double`	dif_rel - `double`
10				
20				
100				

Commenter.

Pourquoi s'intéresse-t-on à l'écart relatif et pas simplement à l'écart absolu ? Si vous ne voyez pas, essayez d'ajouter 10^3, 10^6 ou 10^9 à i et à j dans l'expérience ci-dessus.

Qu'a-t-on mis en évidence avec ces questions ?

Exercice 7. ★★★ Détecter l'égalité à troncation près.

Il est envisageable de dire que deux flottants sont *quasi-égaux* si ils ont le même signe et le même exposant ainsi que la même mantisse au bit de poids faible près.

a. Écrire une fonction C (pour vérifier, vous pouvez également l'écrire en Java ou Python, c'est plus simple) qui prend en entrée un flottant (dont on donnera la précision) et qui en extraie le signe, l'exposant et la mantisse.

b. Écrire une fonction qui réalise le test de quasi-égalité.

c. Reprendre l'exercice 1 en remplaçant le test d'égalité (==) par le test de quasi-égalité.

	`float` C	`double` C	Python
k=10			
k=20			
k=32			
k=100			

d. **Comment ce test de quasi-égalité se compare-t-il à un test d'erreur relative ?**