

RENDU D'EXERCICES N°01

Réalisé par:Mr Lougani Faouzi

Module:Algo et calcul scientifique-L3 math-info

(2019/2020)

Exercice 14 :

d'après l'exo 6,2 on sait bien qu'il y a un plus petit positif x tel que l'expression $(1 \otimes x) \otimes x$ en virgule flottante n'est pas égale à 1, en utilisant la précision simple

aussi on sait que d'après le même exercice qu'il existe des nombres à virgule flottante x pour lesquels n'est pas exactement le même x en utilisant la simple précision pour $1 \otimes (1 \otimes x)$

donc l'expression qui ne donnera pas le résultat x , en supposant que x est un nombre à virgule flottante positif est $\text{sqrt}(x) \otimes \text{sqrt}(x)$ (10.2)

exemple illustratif:

$\text{sqrt}(2) * \text{sqrt}(2) = \sqrt{2} * \sqrt{2} = 1,414213562 \times 1,414213562 = 1,999999999$ donc différent de 2

Exercice 15:

l'expression :

$$\text{sqrt}(x) \otimes \text{sqrt}(x) \quad (10.2)$$

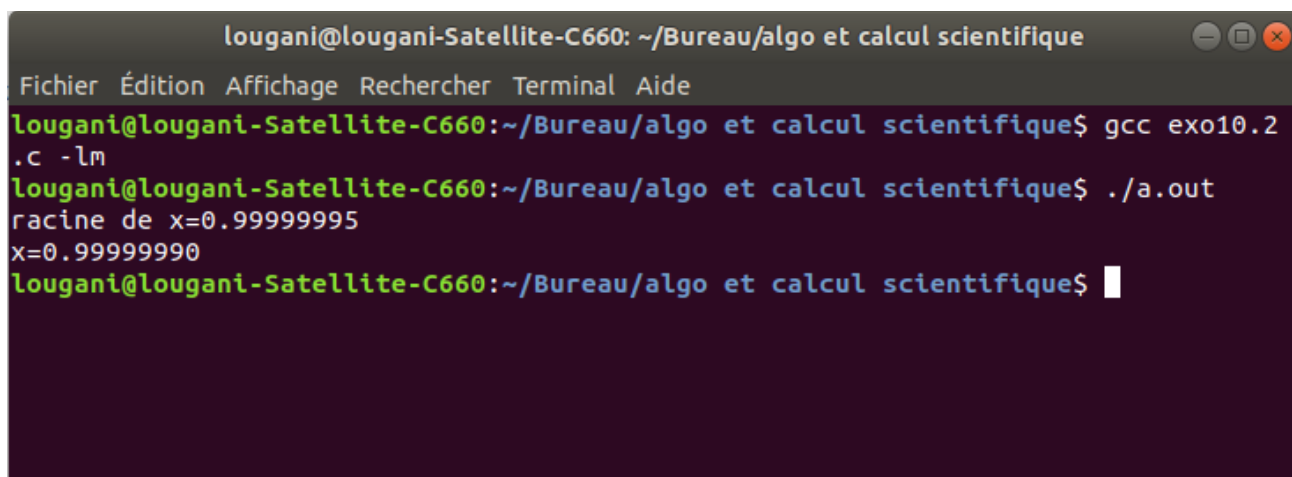
code en c:

```
#include<stdio.h>
#include<math.h>

int main()
{
    double x=0.99999999;
    printf("racine de x=%.8f\n",sqrt(x));
    printf("x=%.8f\n",sqrt(x)*sqrt(x));

    return 0;
}
```

en affichage on aura $x=0.999999990$ or qu'on a $x=0.99999999$ au début donc identique (supprimer le 0 inutile tout à droite)



```
lougani@lougani-Satellite-C660: ~/Bureau/algo et calcul scientifique
Fichier Édition Affichage Rechercher Terminal Aide
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ gcc exo10.2
.c -lm
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ ./a.out
racine de x=0.99999995
x=0.999999990
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$
```

l'expression:

$$\text{sqrt}(x \otimes x) \quad (10.3)$$

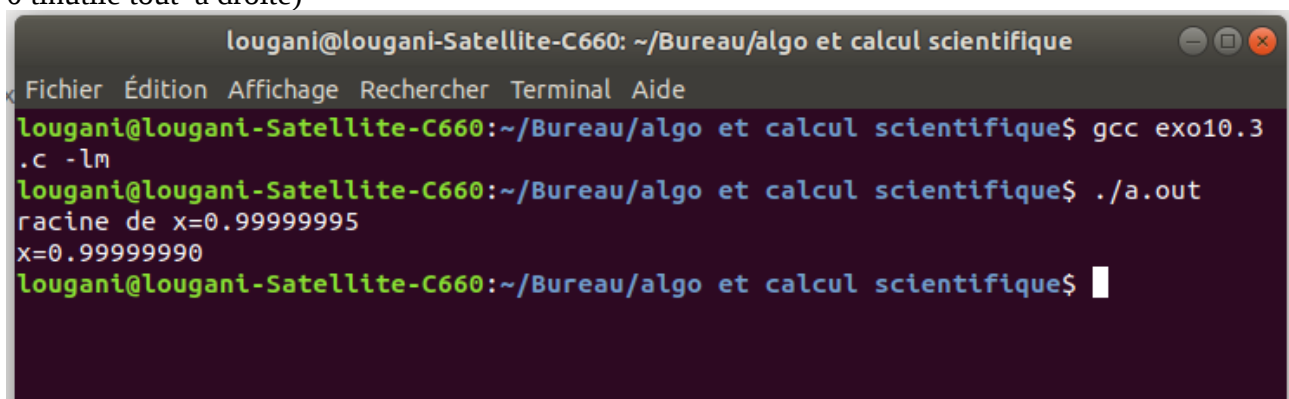
le code en c

```
#include<stdio.h>
#include<math.h>

int main()
{
    double x=0.99999999;
    printf("racine de x=%.8f\n",sqrt(x));
    printf("x=%.8f\n",sqrt(x*x));

    return 0;
}
```

en affichage on aura x=0.99999990 or qu'on a x=0.9999999 au debut donc identique (supprimer le 0 inutile tout à droite)



```
lougani@lougani-Satellite-C660: ~/Bureau/algo et calcul scientifique
Fichier Édition Affichage Rechercher Terminal Aide
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ gcc exo10.3
.c -lm
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ ./a.out
racine de x=0.99999995
x=0.99999990
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$
```

l'expression :

$$\log(\exp(x)) \quad (10.5)$$

code en C :

```
#include<stdio.h>
#include<math.h>

int main()
{
    double x=0.99999999;
    printf("exp de x=%f\n",exp(x));
    printf("x=%f\n",log(exp(x)));

    return 0;
}
```

en affichage on aura x=0.99999990 or qu'on a x=0.9999999 au debut donc identique (supprimer le 0 inutile tout à droite)

```
lougani@lougani-Satellite-C660: ~/Bureau/algo et calcul scientifique
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ gcc exo10.5.c -lm
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ ./a.out
exp de x=2.71828156
x=0.99999990
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$
```

Explication des resultats obtenus:

- un nombre à virgule flottante double précision IEEE 754 64 bits (1 bit pour le signe, 11 bits pour l'exposant et 52 * bits pour la valeur), c'est-à-dire que double a 15 chiffres décimaux de précision.et une plus grande plage de (* 10 ^ + / - 308)
- les fonctions sqrt,log c'est des fonctions mathématiques qui renvoient des valeurs en virgule flottante retournent des doubles donc en passant en parametres des doubles on evite la conversion
- vu que je n'ai pas utilisé des scanf donc , je n ai pas de problemes de format (%lf,%le), pour les printf ils detectent les doubles precisions et les simple precision ,selon le cas elles sont convertis en double avant d'etres passés en parametres .

Exercice 16:

l'expression :

$$\text{sqrt}(x) \otimes \text{sqrt}(x) \quad (10.2)$$

code en c:

```
#include<stdio.h>
#include<math.h>

int main()
{

    float x=0.9999999f;
    float y=sqrt(x);

    printf("racine de x=%.8f\n",sqrt(x));
    printf("x=%.8f\n",y*y);

    return 0;
}
```

en affichage on aura $x=0.9999998$ or qu'on a $x=0.9999999$ au debut donc ce n'est pas identique

```
lougani@lougani-Satellite-C660: ~/Bureau/algo et calcul scientifique
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ gcc exo10.2.c -lm
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ ./a.out
racine de x=0.99999994
x=0.99999988
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$
```

l'expression:

$$\text{sqrt}(x \otimes x) \quad (10.3)$$

code en C :

```
#include<stdio.h>
#include<math.h>

int main()
{
    float x=0.9999999f;
    printf("racine de x=%.8f\n",sqrt(x));
    printf("x=%.8f\n",sqrt(x*x));

    return 0;
}
```

en affichage on aura $x=0.9999998$ or qu'on a $x=0.9999999$ au debut donc ce n'est pas identique

```
lougani@lougani-Satellite-C660: ~/Bureau/algo et calcul scientifique
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ gcc exo10.3.c -lm
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ ./a.out
racine de x=0.99999994
x=0.99999988
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$
```

l'expression:

$$\log(\exp(x)) \quad (10.5)$$

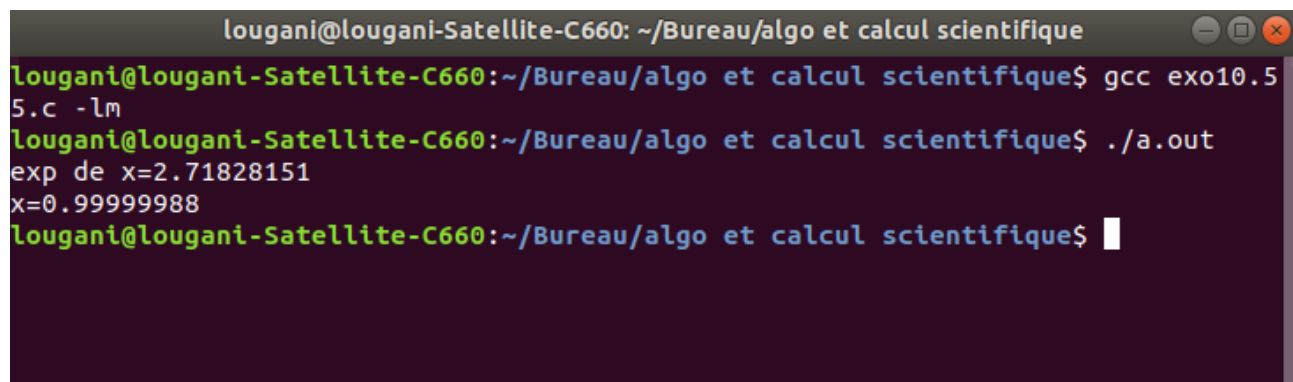
Le code en C:

```
#include<stdio.h>
#include<math.h>

int main()
{
    float x=0.9999999f;
    float y=exp(x);
    printf("exp de x=%.8f\n",y);
    printf("x=%.8f\n",log(y));

    return 0;
}
```

en affichage on aura x=0.9999998 or qu'on a x=0.9999999 au debut donc ce n'est pas identique (le 7eme chiffre)



```
lougani@lougani-Satellite-C660: ~/Bureau/algo et calcul scientifique
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ gcc exo10.5.c -lm
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$ ./a.out
exp de x=2.71828151
x=0.99999988
lougani@lougani-Satellite-C660:~/Bureau/algo et calcul scientifique$
```

Explication des resultats:

- mon choix de comparer les 7 bits pour voir la précision est fondé sur la definition suivante :Le type float, 32 bits de long, a une précision de 7 chiffres. Bien qu'il puisse stocker des valeurs avec une plage très grande ou très petite ($\pm 3,4 \times 10^{38}$ ou $\pm 10^{-38}$), il n'a que 7 chiffres significatifs.
- le stockage dans une variable par exemple $y=\sqrt{x}$ avant de faire la multiplication $y*y$ a permet la converssion du resultat retourné par \sqrt{x} qui par default un double en float afin de faire la multiplication sur des float toujours .
- Un double offre une précision supérieur à un float