1 Durée et rendu attendu

- Durée : 2 heures
- Un rapport **en pdf** avec vos réponses aux questions du sujet, les éventuels inclusions de code et les copies d'écran pertinentes.
- Tout document dans un format autre que pdf ne sera pas considéré.
- Si besoin, un zip avec vos .c **et** makefile.

2 Sujet

Le développement en série de Taylor de la fonction exponentielle $\exp(x)$ est :

$$\exp(x) = 1 + x + x^2/2! + x^3/3! + \dots .$$

Nous allons utiliser ce développement pour calculer des approximations successives de la fonction $\exp$, sans utiliser la version de cette fonction fournie par la bibliothèque mathématique (la `libm` par exemple) excepté comme fonction de *référence* pour des vérifications et des comparaisons.

Le code C de la page suivante effectue le calcul de ce développement et sa comparaison à la fonction de référence `exp()` de `math.h`.

2.1 Compréhension du code

1. Expliquer brièvement quels sont les rôles de `term`, `newsum` et `oldsum`.
2. Quelles sont les conditions qui peuvent conduire à une interruption (une exception) de ce calcul ? Quel sera le traitement dans ce cas ?
3. Expliquer pourquoi le test d'arrêt de la boucle `while` a du sens.
4. Interpréter l'effet d'un tel test sur la précision relative de l'approximation calculée.
5. ($\star$) Quelles sont les conditions nécessaires pour qu'un tel test d'arrêt soit satisfaisant en pratique, *ie.*, la valeur calculée est une approximation satisfaisante de $\exp(x)$?
6. ($\star$) Pour ce traitement, proposer au moins un autre test alternatif adapté au calcul en arithmétique flottante `binary32`.

2.2 Préparatif

1. Ecrire un `makefile` , ou à défaut détailler **chaque fois que nécessaire** dans votre rapport toutes les commandes utilisées pour exécuter le code au fur et à mesure des questions du sujet.

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main()  /* Compute exp(x) from its Taylor series */
{
   int n;
   float x, term, oldsum, newsum;

   printf("Enter x \n");
   scanf("%e", &x);
   n = 0;
   oldsum = 0.0;
   newsum = 1.0;
   term = 1.0;
   /* terminates when the new sum is no different from the old sum */
   while (newsum !=oldsum){
      oldsum = newsum;
      n++;
      term = term*x/n;    /* term has the value (x^n)/(n!) */
      newsum = newsum + term;  /* approximates exp(x) */
      printf("n = %3d   term = %13.6e   newsum = %13.6e \n",
              n,term,newsum);
   }
   printf("From summing the series,     exp(x)=%e \n", newsum);
   printf("Using the standard function, exp(x)=%e \n", exp(x));

   return EXIT_SUCCESS;
}
```

2.3 Première exécution

On exécute ce code pour $x = 0.05$.

2. Fournir les résultats obtenus.
3. Justifier pourquoi l'itération s'arrête.
4. Expliquer pourquoi la valeur affichée de newsum ne varie plus alors que l'itération continue.
5. Proposer une modification ponctuelle et correcte du code pour observer votre justification. Essayer et présenter les résultats ainsi obtenus.

6. Que penser du résultat obtenu par le calcul du développement ?

2.4 Deuxième exécution

On exécute ce code pour $x = 10.0$.

2. Fournir les résultats obtenus.
3. Que penser du résultat obtenu par le calcul du développement ?
4. Que penser de l'évolution de l'itération dans ce cas ?

2.5 D'autres exécutions

1. Effectuer des traitements similaires pour $x = 1, -1, -5, -10$.
2. Rassembler les résultats (finaux) calculés et les résultats de référence dans un tableau selon les 6 valeurs de x considéres jusqu'à présent.
3. Que penser de ces résultats ?

2.6 Un focus

On reprend l'exécution pour $x = -10.0$.

1. Fournir les résultats obtenus.
2. Que penser du résultat obtenu par le calcul du développement ?
3. Comment les itérations se comparent-elles par rapport à celles de $x = 10.0$?
4. Comment expliquer l'évolution de `newsum` dans ce cas ?
5. ($\star$) En observant l'évolution de `newterm`, expliquer pourquoi le résultat obtenu est aussi décevant.

2.7 Un autre focus

On reprend l'exécution pour $x = -5.0$.

1. Fournir les résultats de l'itération.
2. Que penser du résultat obtenu par le calcul du développement ?
3. En s'inspirant de l'analyse précedente, justifier la relative qualité du résultat obtenu.

2.8 Changer le fusil d'épaule

On sait que $\exp(x) = \frac{1}{\exp(x)}$.

1. Utiliser cette propriété pour modifier le code précédent **lorsque c'est opportun**.
2. Exécuter un traitement pour $x = -10.0$ et fournir les résultats obtenus.
3. Que penser du résultat final ainsi obtenu ?
4. Expliquer.

2.9 ($\star$) Bonus

1. Proposer une autre stratégie d'amélioration du calcul de ce développement.