Lundi 21 septembre 2015

Exercice 1. Pour chacun des programmes proposés, les comprendre en détail, faire des modifications, et comprendre les messages émis par le compilateur. Essayer de provoquer une message d'erreur de l'éditeur de liens.

Exercice 2. Écrire un programme qui demande une valeur réelle x et calcule $f(x)$, où f est la fonction implémentée par `mystere` :

```
double mystere (double x)
{
  if (fabs(x) < 1e-10)
    return x;
  double y = mystere (x / 3);
  return y * (3 - 4 * y * y);
}
```

et percer le mystère. La fonction `fabs` renvoie la valeur absolue et est déclarée dans le fichier d'interface `cmath`.

Exercice 3. Les programmes `trapeze` implémentent (surprise !) la méthode des trapèzes pour le calcul d'une intégrale.

$$I(f,a,b,N) = \frac{b-a}{N}\left(\frac{f(a)+f(b)}{2} + \sum_{k=1}^{N-1} f\left(a + k\frac{b-a}{N}\right)\right)$$

`trapeze3.cc` paraît plus simple et plus naturel. Que peut-on lui reprocher ?

Exercice 4. En partant du programme premiers.cc, en écrire un qui compte combien il y a de nombres premiers inférieurs ou égaux à n, où n est donné par l'utilisateur du programme. On peut aussi essayer de factoriser n. Utiliser la commande `time` et les options d'optimisation (par exemple `c++ -O3 -c premier.cpp`) et tenter d'améliorer les performances

Exercice 5. Le *mot de Fibonacci* est la suite infinie de caractères qui commence ainsi :

`abaababaabaababaababaabaababaabaababaababaabaababaababa...`

limite (en quel sens ?) de la suite de mots définie par

$$\begin{aligned} F_0 &= \texttt{b} \\ F_1 &= \texttt{a} \\ F_n &= F_{n-1}F_{n-2} \quad \text{pour } n > 1. \end{aligned}$$

Afficher les n premiers caractères du mot de Fibonacci. On pourra utiliser le type `string`. Quelle est la proportion des `a` dans le mot de Fibonacci ?