

Vitesse d'exécution

On peut mesurer le temps d'exécution pour comparer la performance des algorithmes

```
...  
long T0 = System.currentTimeMillis(); // temps de début  
...  
long dT = System.currentTimeMillis()-T0; // temps (ms) dépassé
```

Vitesse d'exécution 2

Le temps d'exécution dépend de beaucoup de conditions :

- entrée
- ordinateur
- environnement *run-time*
- température
- phase de la lune
- [...]

Ce qui nous intéresse, c'est la dépendance de l'entrée

⇒ abstraction de l'implantation

p.e., calculer nombre d'opérations «primitives»

ceci est une fonction de la taille de l'entrée (disons, N)

question typique : quel est l'effet de doubler N ?

Ordre de grandeur des fonctions

- **constant**
- **logarithmique**
- **linéaire**
- $N \log N$
- **quadratique**
- **cubique**
- **[polynomial]**
- **exponentiel**
- **superexponentiel** (p.e, factoriel)
- **non-calculable** (p.e., fonction de Radó)

Logarithmes

$$\lg x = \log_2 x \text{ et } \ln x = \log_e x \ (x > 0)$$

$$\log(xy) = \log x + \log y; \log(x^a) = a \log x$$

$$2^{\lg n} = n; n^n = 2^{n \lg n}; \log_a n = \frac{\lg n}{\lg a}; a^{\lg n} = n^{\lg a}$$

Factoriel

$$n! = 1 \cdot 2 \cdots n$$

approximation de Stirling :

$$n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

donc $n!$ est superexponentiel

on voit aussi que $\ln(n!) \approx n \ln n - n + \frac{1}{2} \ln n + \ln \sqrt{2\pi}$, essentiellement $n \ln n$