Cours de C++, 1ère année, HE-Arc

Série 9.3

Exercice 1 : Performance des opérations avec `vector` et `list`

En utilisant les déclarations suivantes :

```
typedef vector<int> myArray;
typedef list<int> myList;
```

On demande d'écrire un programme capable de tester l'efficacité des tâches ci-dessous :

1. Insertion de 100'000'000 d'éléments à la fin d'un `vector`
2. Insertion de 100'000'000 d'éléments a la fin d'un `vector` (en utilisant `reserve`)
3. Insertion de 500'000 d'éléments au début d'un `vector` (avec la méthode `insert`)
4. Insertion de 500'000 d'éléments au début d'un `vector` (la méthode `reserve`)
5. Insertion de 500'000 d'éléments à la fin d'une `list`
6. Insertion de 500'000 d'éléments au début d'une `list`
7. Insertion de 500'000 d'éléments à la fin d'une `forward_list`
8. Insertion de 500'000 d'éléments au début d'une `forward_list`

Pour mesurer le temps, vous pouvez utiliser la fonction `clock` de `<ctime>` :

```
#include <ctime>
clock_t clockTicks1 = clock();
// <<<<<<<<<<<   placer ici le code a chronométrer
clock_t clockTicks2 = clock();
cout <<"Difference:" << (float)(clockTicks2 - clockTicks1)/CLOCKS_PER_SEC<<endl;
```

Note: *Sous Windows, pour plus de précision, il existe la classe `PerfChrono` (<windows.h>)*

Quelles sont vos conclusions par rapport aux écarts de performances ?

Est-ce logique ?

Les différents conteneurs possèdent-ils les même méthodes de modification? (voir ici) pour les explications...