Chapter-4 C++標準テンプレートライブラリの利用(上級)～

4.3 可変長配列vectorの特徴

4.3.1 処理速度

可変長配列vectorの代表的な関数の処理速度は次のようになります。

1. 要素の追加。ほとんどの場合でO(1)だが、たまにO(N)
2. 要素へのアクセス時間がO(1)
3. 要素の検索時間がO(N)
4. 要素の削除の時間がO(N)

では、これらについてvectorを簡単に実装したMyVectorクラスを参考に見ていきましょう。

1.要素の追加。ほとんどの場合でO(1)だが、たまにO(N)

MyVector::PushBack()関数を見てください。

2.要素へのアクセス時間がO(1)

MyVector::opertor[]を見てください。

3.要素の検索時間がO(N)

MyVector::Find()関数を見てください。

4.要素の削除時間がO(N)

MyVector::Erase()関数を見てください。

4.3.2 メモリ関係

メモリ関係で、vectorを利用する際は次の点に注意する必要があります。

メモリ使用量が見かけ上の必要メモリ量＊２になる場合がある。

4.3.3 どのようなケースでvectorを使うべき？？？

要素へのアクセス時間はO(1)となっていて、どのコレクションよりも高速。しかし、PushBack()関数による要素の追加で、必要なメモリの倍が確保されることがあったり、新しいメモリ領域にデータのコピーが発生するなどの問題がある。このため、処理時間がO(N)となってしまう場合がある。これがゲームでは致命的な問題となる。特にアクションゲーム。では、どのようなケースでvector使うべきか？配列の上限を決めることができる場合。なので、便利な固定長配列と思って使うべき。 後述するreserve()を利用するとメモリをドカッと確保することができる。ゲームであれば、これを利用して、capacityを超えないように工夫するべき。

4.3.4 std::vectorあれこれ

要素を追加すると、イテレーターが無効になる場合がある。

要素を削除すると、イテレーターが無効になる。

reserve()関数を利用して、メモリを一気に確保すべし。

評価テスト

次の評価テストを行いなさい。
評価テストへジャンプ