第1回　C++講座

今日やること

- 導入

C++

C++は、オブジェクト指向プログラミングを取り入れた、C言語の拡張版。

オブジェクト指向(object-oriented)とは

現実世界のObjectや概念同士の相互作用で、現象をとらえる考え方

・現実世界でまとまったデータはまとめる

・観察対象(object)を「主語」(=class)にしたプログラミング

各objectは、「動詞」(=命令)を持っていて、その動作を行う

オブジェクト指向を支える重要な3つの要素

- カプセル化
- ポリモーフィズム
- 継承

カプセル化

プログラムコード(命令)とコードが扱うデータをひとまとめにして外部からの干渉や誤用から保護する

まとまった概念ごとに自己完結型の「オブジェクト」を作成する

ポリモーフィズム

1つの名前で、複数の関連する目的に使用できる性質

- 関数のオーバーロード

1つの名前で、異なる型の引数をとる関数を代表させることができる
ex)
C:abs(double),labs(long),fabs(float)
C++:abs()

- 演算子のオーバーロード

クラス間の操作を始め、独自の演算子を定義することができる
ex)　carクラスと、truckクラスがあった時、それぞれのオブジェクトの残り燃料の和を求める独自の”+”演算などを新たに定義できる

継承

1つのオブジェクトが、他のオブジェクトの性質を受け継ぐプロセス

・階層的な分類を表現可能

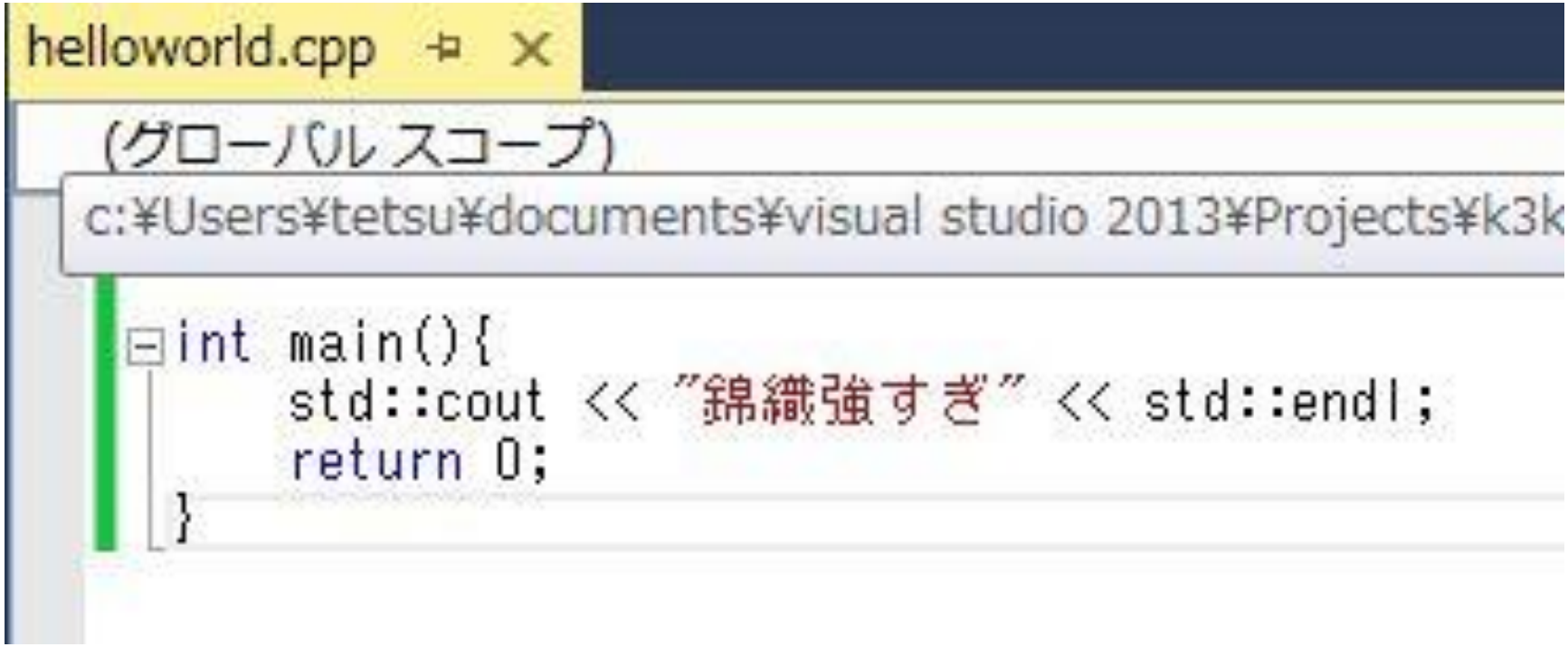
・上位オブジェクトの命令を下位のオブジェクトに対しても実行できる

CとC++の違い

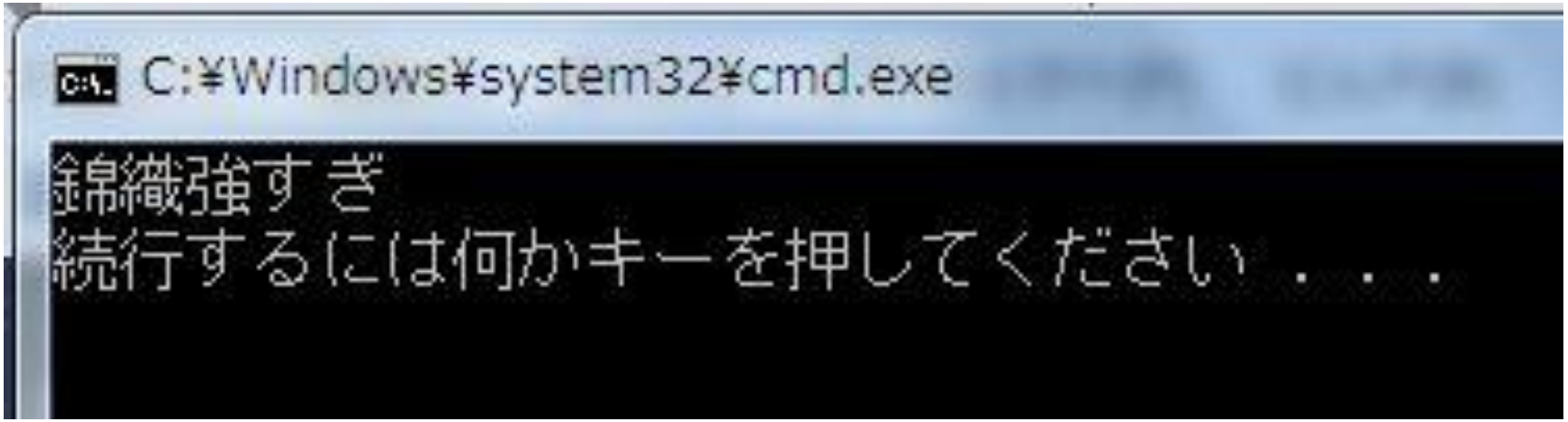
C	C++
printf	cout
scanf	cin
malloc,calloc	new
free	delete

cout

C++では、printf,scanfの代わりにcout,cinを使う

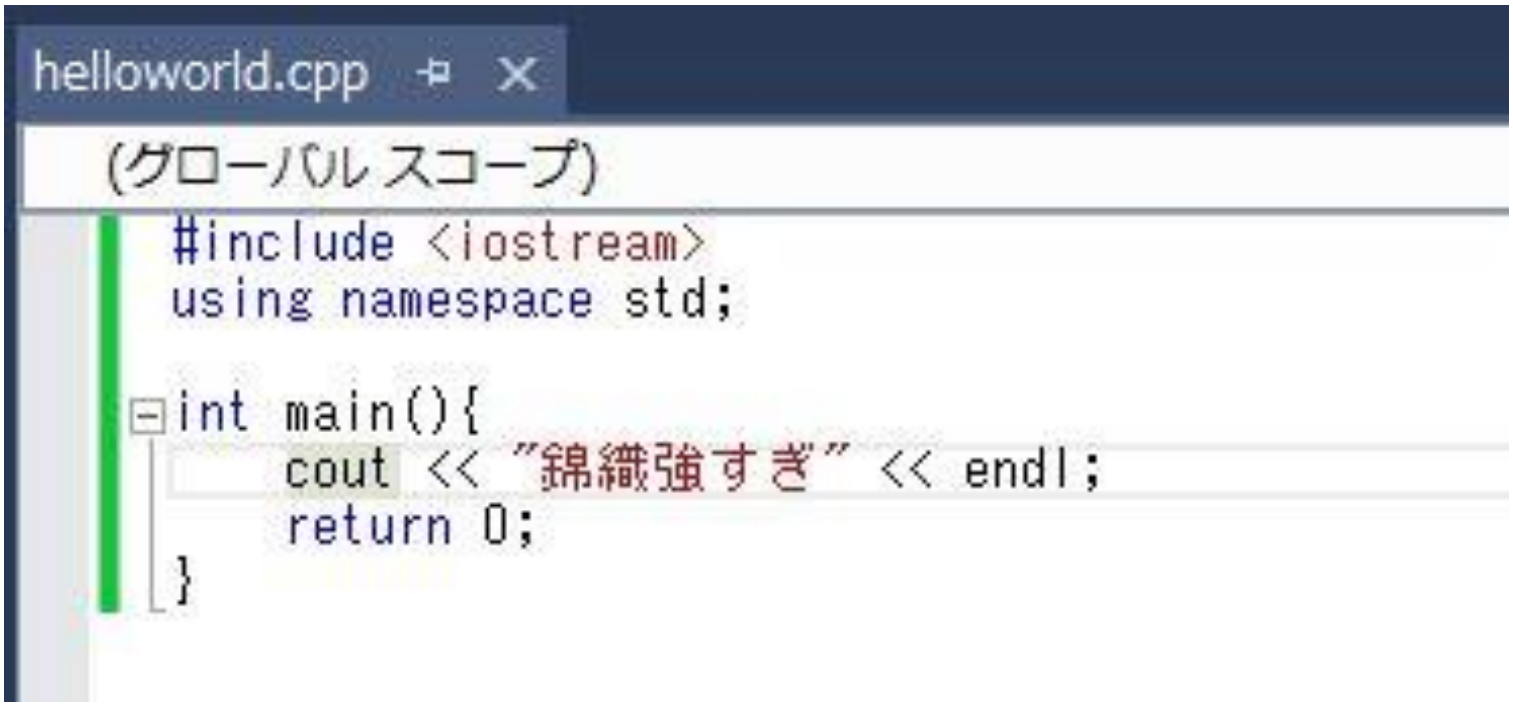


結果



std::(スコープ解決演算子)

std::coutといちいち書くのは面倒！
→using namespace std;と書いておけば省略できる



名前空間(namespace)

using namespace std;とは、「使っているクラス、関数がstdという名前空間の中です」という宣言

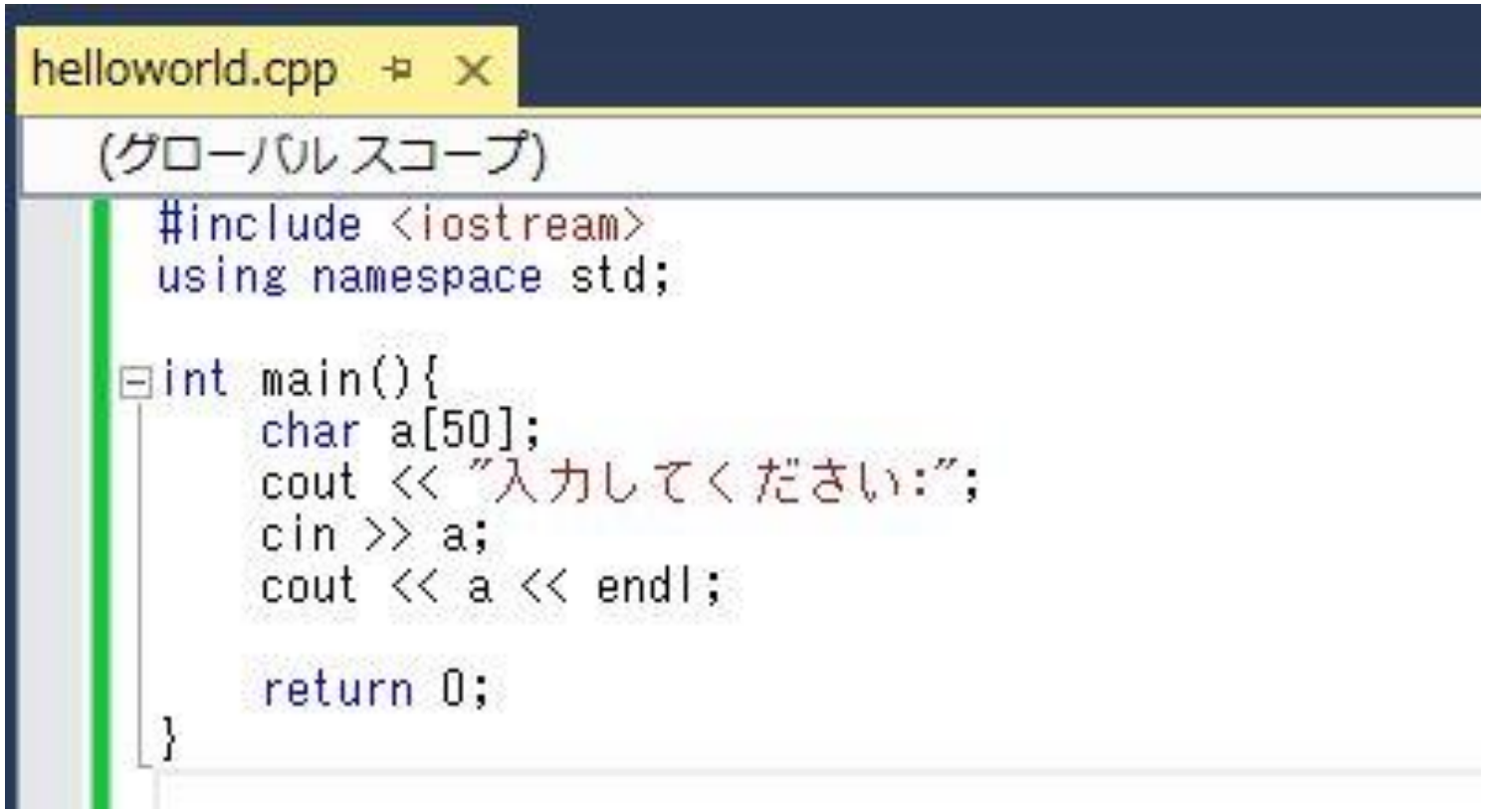
同じ名前の関数、クラスは競合する(コンパイラが区別できない)

→　「名前空間」の導入により、同じ名前によるトラブルを解決する

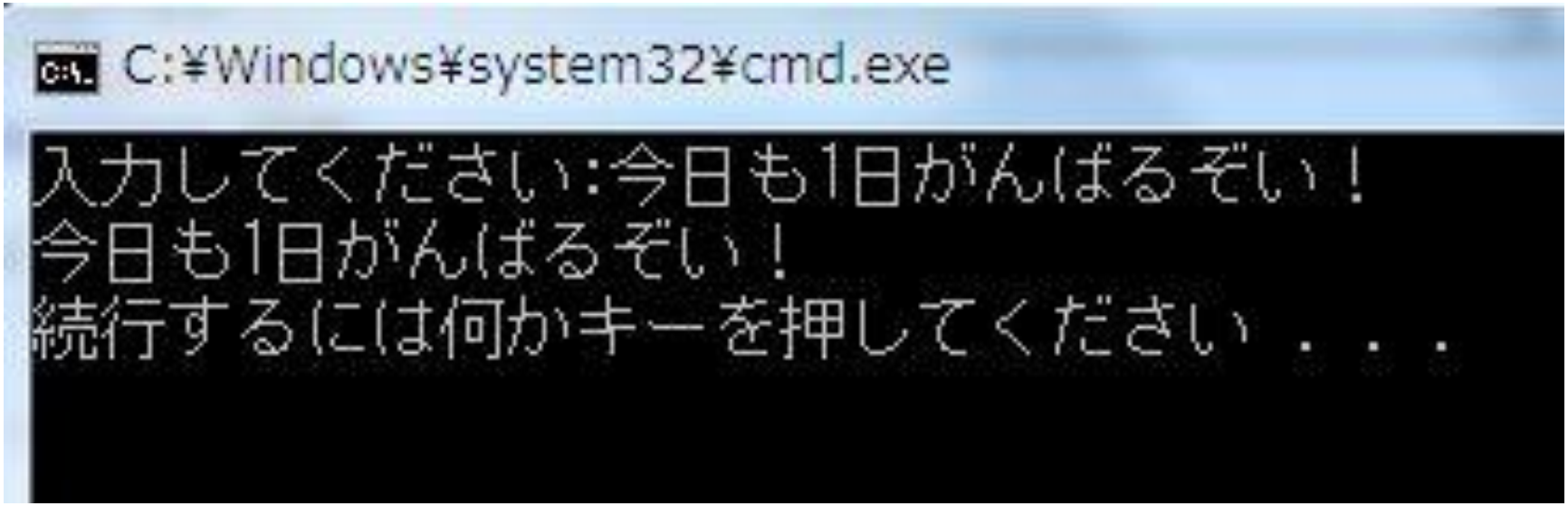
さっきの例を使うと、「stdという名前空間の中にcout関数とかが入ってる」

cin

scanfより簡単だよ！



結果


C:¥Windows¥system32¥cmd.exe
入力してください:今日も1日がんばるぞい！
今日も1日がんばるぞい！
続行するには何かキーを押してください . . .

newとdelete

new…メモリの動的確保
delete…メモリの解放

定義　ポインタ変数 = new 型名;
定義　delete ポインタ変数;

newは、型名のオブジェクトが格納できるメモリを確保し、その先頭のアドレスを返す
deleteは、newによって確保されたメモリを解放する

new,deleteを使うメリット

- オブジェクトに必要なメモリ量を自動的に計算
- 指定した型のポインタを自動的に返す
- 動的に割り当てたオブジェクトを初期化できる
- new,deleteはオーバーロードできる
- 例外処理に対応している

new 初期値の設定

定義
ポインタ変数 = new 型名 (初期値);

ポインタ変数 = new 型名 [要素数];
delete[] ポインタ変数: