**課題1.　リファクタリング**

**ゲームの挙動は一切変更せずに、プログラムを下記の通りに修正してください。**

**・Gameプロジェクトに存在するクラスのメンバ変数を全てprivateにする(メンバ関数はpublic可)**

　→参考：[C++ public private アクセス指定子の使い方 | プログラミングランド (skpme.com)](https://skpme.com/618/)

**・FindGO及びFindGOsの使用禁止**

　→既にFindGO及びFindGOsを使用している箇所は削除して、別の方法で実装する

**・Playerクラスのメンバ変数playerStateの型を列挙型(enum)にして使用する**

　→参考：[C++でのenum(列挙型)の使い方とは？class指定方法や文字列変換方法を紹介！ | .NETコラム (fenet.jp)](https://www.fenet.jp/dotnet/column/language/7003/)

**課題2.　ゲーム要素の追加**

**課題1の条件を引き継ぎつつ、ゲームに以下の要素を追加してください。**

**Game/Game.exeで完成品が起動します。**

**・プレイヤーを追従するエネミーを追加**

　→新しくEnemyクラスを作成する

　→エネミーは3体配置

　→プレイヤー等と同じように、ゲームクリア画面等に遷移する際に全て削除する

　→モデルはAssets/modelData/enemy/enemy.tkmを使用

**・ゲームオーバーの追加**

　→プレイヤーがエネミーに衝突すると、プレイヤーがダウンアニメーション(Assets/animdata/KneelDown.tka)を再生した後にゲームオーバー画面に遷移

　→ゲームクリアと同じように、ゲームオーバーに遷移する際にプレイヤー等が削除する

　→画像はAssets/sprite/gameover.ddsを使用

**・プレイヤーが☆を取得するとエフェクトが発生**

　→エフェクトはAssets/effect/get.efkを使用