課題13

Q01: 全局配置管理（Global Configuration Manager）の実装

（プログラム名：**ex13_1.cpp**）

【背景・用途】

実際のアプリケーション開発では、データ保存先ディレクトリ、タイムアウト値など、プログラム全体で共有したい設定値（全局配置）が多数存在します。
これらを各関数に毎回引数として渡すのは煩雑なため、どこからでも同じ設定を参照・更新できる仕組みを用意するのが一般的です。

本課題では、C++ の **static** と **std::map** を使い、プロセス内で 1 つだけ存在する「設定テーブル」を管理するクラス **AppConfig** を作成します。

【目的】

- 静的メンバ関数（static member function）の基本的な使い方を理解する。
- 関数内 **static** により「プログラム全体で 1 つだけ存在する設定テーブル」を実現できることを理解する。
- **std::map** の基本操作（**operator[]**, **find**, **end**）に慣れる。

【配布コード】

今回の課題では、コードを複数のファイルに分割する必要はありません。
以下のコードをそのままコピーし、**TODO** 部分（**AppConfig::set**, **AppConfig::get**）を実装しなさい。
（それ以外の部分は変更しないこと）

```
C/C++
#include <iostream>
#include <string>
#include <map>

// アプリ全体で共有する設定（キー→値）を管理するクラス
class AppConfig
{
private:
    // 設定を保持する map を返す（関数内 static により1つだけ生成され、参照として返す）
    static std::map<std::string, std::string> &instance()
    {
        static std::map<std::string, std::string> cfg; // 全局唯一配置
        return cfg;
    }
```

```
public:
    // 設定を追加／更新する（存在すれば上書き）
    static void set(const std::string &key, const std::string &value)
    {
        // TODO: instance() から map を取得し、key に対応する値を value に設定する
    }

    // 設定を取得する（存在しない場合は "<unset>" を返す）
    static std::string get(const std::string &key)
    {
        // TODO: instance() から map を取得し、key を探す
        //       見つかった場合は対応する値を返す
        //       見つからない場合は "<unset>" を返す
    }

    // 現在の設定一覧を表示する
    static void display_settings()
    {
        auto &cfg = instance();
        std::cout << "Settings:" << std::endl;
        for (auto it = cfg.begin(); it != cfg.end(); ++it)
        {
            // it->first  : キー
            // it->second : 値
            std::cout << "  " << it->first << ": " << it->second <<
std::endl;
        }
    }
};

int main()
{
    // 初期設定の登録
    AppConfig::set("log.level", "INFO");
    AppConfig::set("data.dir", "/mnt/data");

    // 初期設定一覧の表示
    AppConfig::display_settings();

    // 設定の更新
    AppConfig::set("log.level", "WARN");

    // 個別に取得（存在しないキーは "<unset>"）
    std::cout << "log.level = " << AppConfig::get("log.level") << "\n";
    std::cout << "data.dir  = " << AppConfig::get("data.dir") << "\n";
    std::cout << "timeout   = " << AppConfig::get("timeout") << "\n";
}
```

【実行例】

```
Settings:
  data.dir: /mnt/data
  log.level: INFO
log.level = WARN
data.dir  = /mnt/data
timeout   = <unset>
```