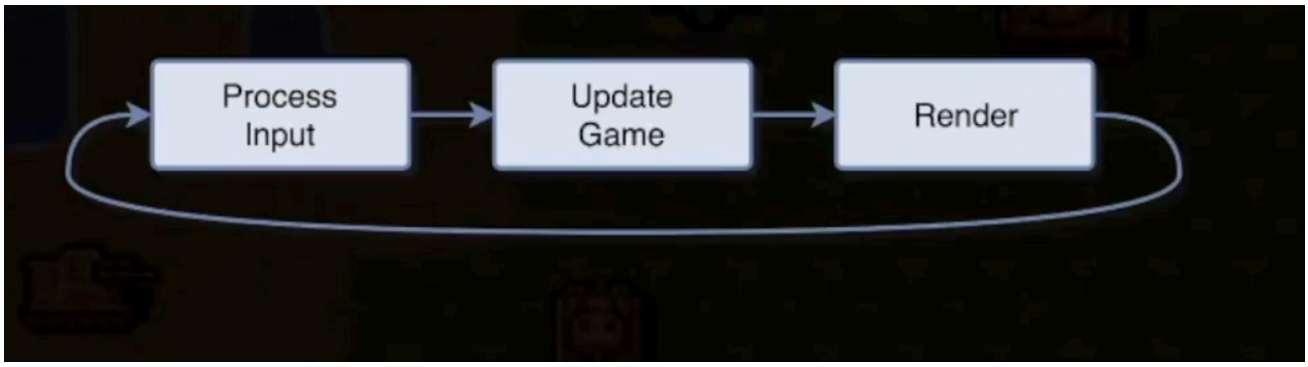
DAY2 GameLoop




```
while(true) {
  game -> processInput();
  game -> update();
  game -> render();
}
```

Troubles

快慢不同的机器 -> 稳定的运行机制

```
SetUp(); // Initialize
Update();
Draw();
```

HeaderGuard

确保整个项目中这个头文件在precomplie时只引用一次

这是 C++ 中的头文件保护（Header Guard）机制，也叫做包含保护符（Include Guard）。让我详细解释一下：

基本语法

```
#ifndef GAME_H     // 如果没有定义 GAME_H
#define GAME_H     // 那么定义 GAME_H
// ... 头文件内容 ...
#endif             // 结束条件编译
```

原理和作用

1. **防止重复包含**
   - 当一个头文件被多次 `#include` 时，这个机制可以防止代码被重复编译
   - 第一次包含：`GAME_H` 未定义，所以定义它并编译内容
   - 后续包含：`GAME_H` 已经定义，就会跳过这个文件的内容
2. **举个例子**：

```
// A.h
#ifndef A_H
#define A_H
struct Point { int x, y; };
#endif

// B.h
#include "A.h"

// C.h
#include "A.h"

// main.cpp
#include "B.h"
#include "C.h"
```

如果没有头文件保护：

- `Point` 结构会被定义两次（通过 B.h 和 C.h）
- 这会导致编译错误（重复定义）

有了头文件保护：

- 第一次通过 B.h 包含 A.h 时，定义 `Point`
- 第二次通过 C.h 包含 A.h 时，因为 `A_H` 已定义，跳过内容

- 避免了重复定义

在现代 C++ 中，也可以使用 `#pragma once`：

```
#pragma once
```