**Ihor Sviatskyi**

**Odessa, Ukraine**

**Mobile:** 0989541848

**Skype:** Ingwar Sv

**E-mail:** [sviatskyi@gmail.com](mailto:sviatskyi@gmail.com)

**Telegram:** @IngwarSv

**PONG**

Игра Pong c использованием библиотеки SDL2

1. Показать знание языка C++ (не С):

* Для следования RAII использованы *std::unique\_ptr<Т>* библиотеки *<memory>* для атрибутов класса (*class Core*), включая объекты библиотеки SDL, для которых реализованы *default deleters*;
* использован функционал шаблонов для функций (*файл cleanUp.h*);
* использован ГПСЧ библиотеки *<random>* (*class Core*);
* использован функционал библиотеки *<chrono>* (*class Timer*) для вычисления и управления FPS: FPS отображается на экране, при увеличение константы *DEF\_SETT::DELAY\_UPDATE* FPS падает;
* использование Uniform инициализации, инициализации атрибутов класса через список инициализации, использование static\_cast<>, *enum class* (*файл Specifications.h*)*,* ссылок, библиотеки *<string>* и т.д.

1. Показать знание ООП и умение его использовать:

* программа реализована на базе объекта класса Core, в котором атрибутами класса являются объекты классов Ball, Paddle, объекты библиотеки SDL2 и ее расширений;
* наследование не применено, наследование – отличный инструмент, но в данной реализации необходимости его использования не возникло.

1. Показать умение использовать STL – контейнеры, итераторы применяю постоянно, знаком и стараюсь задействовать функционал библиотеки <algorithm>, но в данной реализации необходимости их использования не возникло.
2. Аккуратно оформленный структурированный код:

* каждый класс разбит на заголовочный файл и файл реализации;
* глобальные константы вынесены в отдельный файл и заключены в namespace (*файл Specifications.h*);
* используются комментарии.

1. Показать умение подключать и использовать внешние библиотеки:

* подключены и используются дополнительные библиотеки С++, библиотека SDL2 и ее расширений *(SDL\_image, SDL\_ttf, SDL\_mixer),* для следования идиоме RAII библиотеки завернуты в отдельные классы.

1. Показать знание архитектуры игровых движков:

Игра состоит из:

* класса gameManager (*class Core*), который включает:
* модуль ввода InputSystem: *Core::input();*
* модуль GameLogicSystem: *Core::updateF()*;
* модуль RenderSystem: *Core::Render()*;
* звуковой модуль AudioSystem: подключено *SDL\_mixer*.

1. Показать использование паттернов проектирования (не только Singleton):

- использован только Singleton для главного класса программы *(class Core)*, рационально задействовать другие паттерны не смог. В других проектах использовал также Factories, Observer, State, Strategy, практиковался в написании примеров других паттернов.

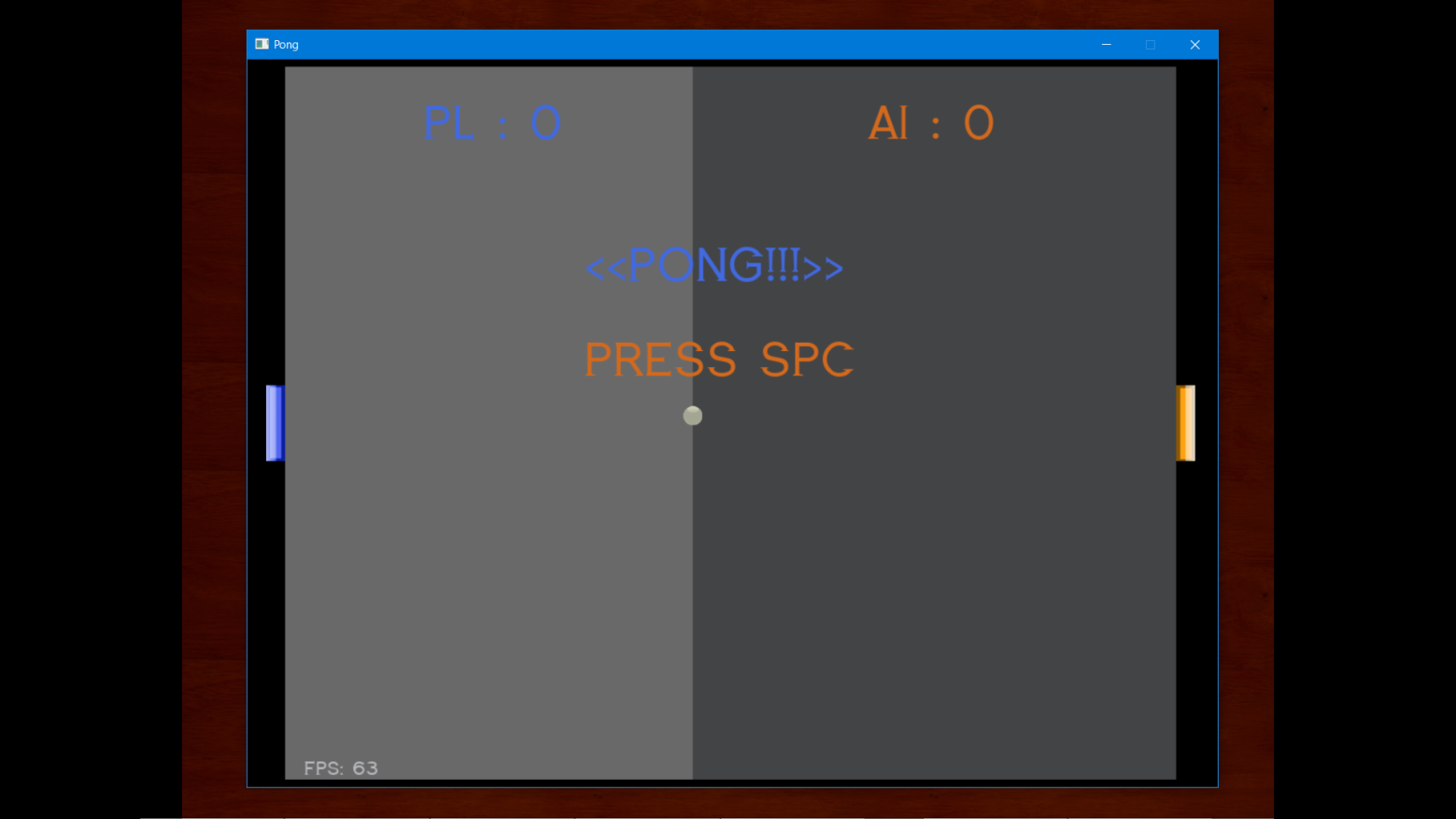
1. Код должен компилироваться либо под Win32 либо под Ubuntu 16.04 x64 или 18.04 x64, чтобы можно было проверить, а также запускаться без дополнительных телодвижений (F5 или же непосредственно запуск из папки с бинарником):

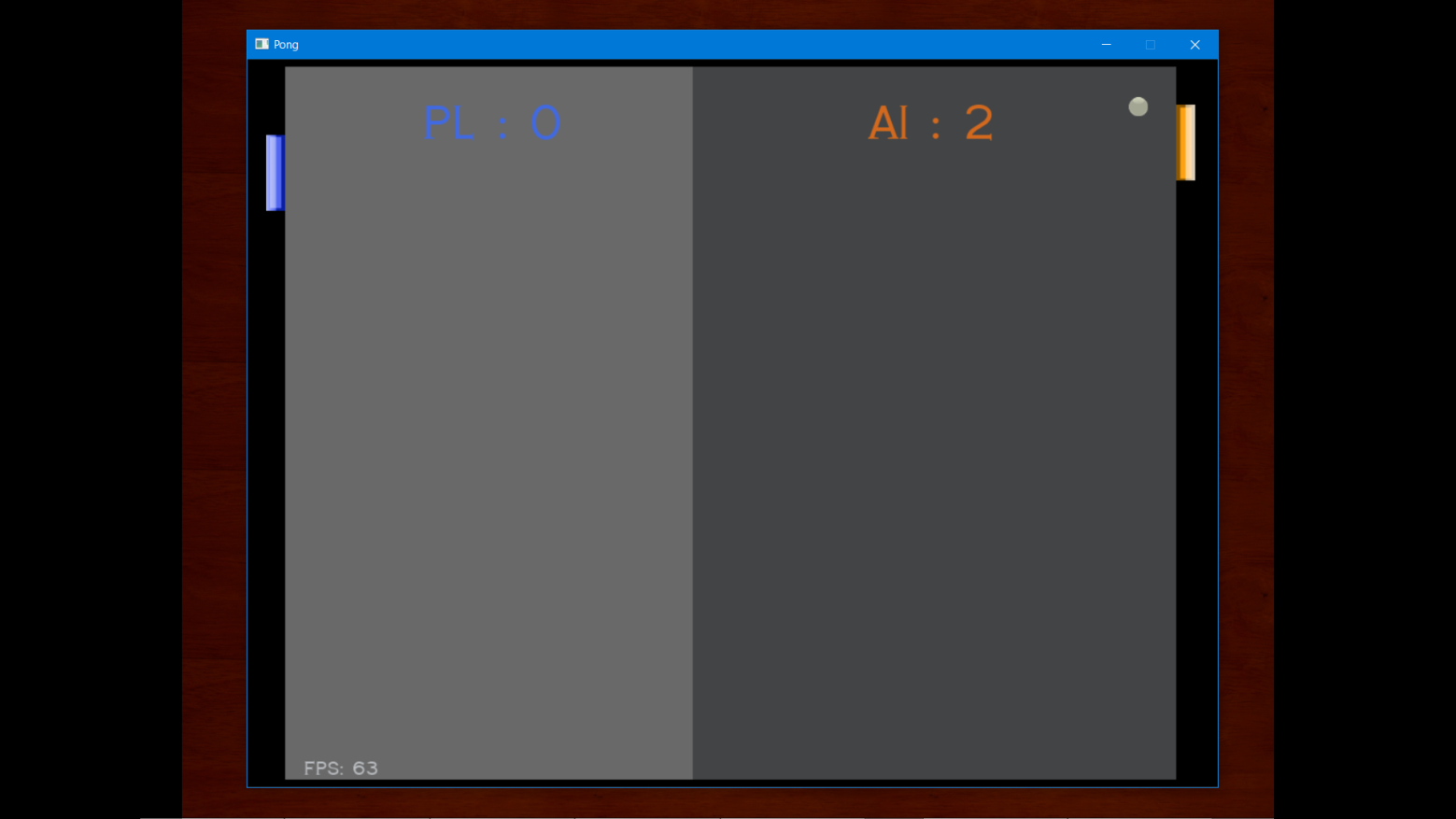
* IDE: Microsoft Visual Studio 2019 Community edition;
* программа запускается в Debug и Release версиях.

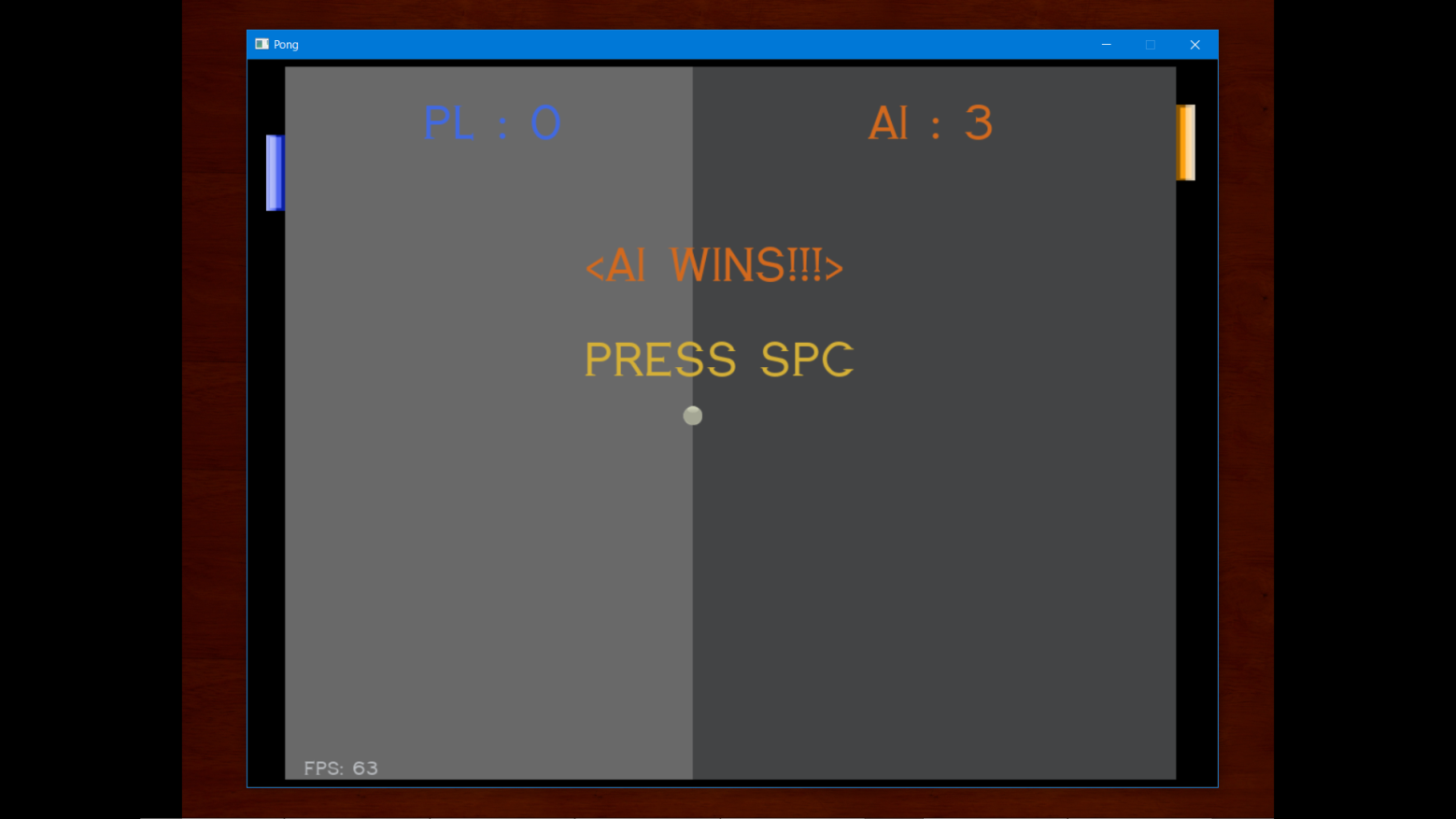
1. Реализована игра против бота.

<https://drive.google.com/file/d/1fjPAFI_qbFu_wNgvKRNbQlFoaw8Mb-Bj/view?usp=sharing>

**Скриншоты**

****

****

****

**Спасибо за внимание!**