Архитектурная часть:

В ПО предусмотрено 2 основных класса: BaseAccount и Player. Player наследует BaseAccount.

BaseAccount содержит в себе структуру Store, которая содержит следующие поля:

- Имя

- Кол-во побед

- Кол-во поражений

- Кол-во ничьих

Поле имени – это статический массив символов на 128 байт. Суммарный вес сохранённого класса этого типа(sba) – занимает объём ровно 140 байт.

Данный тип аккаунтов используется для хранения статистики ПК и Гостевого аккаунта. В эти типы аккаунтов не могут войти обычные игроки, А также, статистика гостевого аккаунта при каждом запуске ПО обнуляется.

Объект типа Player, кроме полей BaseAccount содержит следующие собственные:

- Пароль

- Сохранённый массив

- Логическую переменную отвечающую за тип символа игрока

- Поле с именем последнего сохранённого противника

- Логическая переменная «Ходит первый»

Соответственно поля под пароль и имя прошлого сохранённого противника являются подобными полю «Имя» объекта типа BaseAccount, и имеют размер в 128 байт.

Сохранение с этим типом объекта(spa) – занимает объём ровно в 407 байт.

Сохранённые объекты двух выше перечисленных классов содержаться в директории players.

Функции, отвечающие за логику и графику - находятся в отдельных файлах «Functions.h» и «Functions.cpp».

Техническая документация:

В целях использования графического интерфейса была использована высокоуровневая библиотека API – SFML v 2.4.2. Поэтому для корректной работы ПО – в директории с исполняемым файлом должны находиться соответствующие библиотеки данного API.

Для перемещения между пунктами меню используется id-указатель на конкретную функцию. В случае сбоя в ПО – ему будет присвоено значение -1, и будет выведена в консоль ошибка об аварийном прекращении работы программы.

Функции ПО находятся в «Functions.cpp», и разделены на 4 категории:

- Функции – меню – те функции, которые отвечают за показ меню, и их функционал

- Дополнительные функции меню – те, которые являются дополнениями к основным

- Игровые графические функции.

- Дополнительные игровые функции

Все функции, которые находятся в 1 и 3 группе – возвращают значение типа int, для того чтобы в основной функции main была возможность переходить между окнами.

Функции:

**int main(void)** – основная функция ПО. Осуществляет инициализацию 2 объектов типа BaseAccount и 2 объектов типа Player. Также она обнуляет статистику объекта BaseAccount Guest через соответствующий метод класса. После – происходит инициализация переменной, отвечающей за переход между окнами – iStatus, и в зависимости от её значения рисует соответствующее окно.

Всегда возвращает 0.

ФУНКЦИИ-МЕНЮ

**int DrawMenu(void)** – отвечает за графический показ интерфейса, позволяющего переходить между соответствующими окнами. При выборе кнопки выхода – вызывает функцию exit(0)

**int ComparePvP(const Player& player, const Player& enemy**) – проверяет кто из игроков вошёл. Если оба игрока вошли – переходит в окно игры. Если вошёл только первый – выведет сообщение о том, чтобы продолжить игру, но 2 игрок будет под гостевым аккаунтом, или выйти в меню. Если не вошёл первый игрок – то выведет сообщение о том, чтобы 1 игрок вошёл, и предложит вернуться только в меню. В параметрах принимает ссылки на объекты типа Player.

**int DrawMessage(const char\* message, bool sw, int secRet = 1)** – функция для вывода сообщений. Строка message предназначена для указания текста, логическая переменная sw – указывает на наличие второй кнопки, переменная secRet – указывает на то, куда ведёт вторая кнопка. По умолчанию ей задано id -1, поскольку, не принципиально, куда она должна указывать, если её нету. Используется вышеупомянутой функцией, и функцией написанной ниже:

**int ComparePvCPU(const Player& player)** – проверяет вошёл ли первый игрок. Если да – переходит к игре против ПК. Иначе – выдаёт сообщение о предложении войти 1му игроку, или играть под гостевой записью.

**int SelectPlayer(Player& player, Player& enemy)** – менеджер аккаунтов. Позволяет входить в аккаунт, создавать новые, просматривать статистику аккаунтов. Однако, функция удаления аккаунтов не предусмотрена. Вход и создание аккаунта производятся через консоль. Игрок обязательно должен иметь пароль.

**int Help(void)** – выводит краткую информацию о ПО, его использовании для пользователя, и создателе. Возможен переход только в меню.

**int DrawLog(void)** – выводит список кто с кем играл, и результат игры. Возможен переход между списком, и выход в меню.

ВСПОМОГАТЕЛЬНЫЕ ФУНКЦИИ МЕНЮ

**int CountPlayers(void)** – считает количество сохранённых объектов типа Player(spa) в директории players.

**BaseAccount::Store\* LabelOfPlayers(void)** – возвращает массив типа Store найденных игроков. Размер массива считает самостоятельно благодаря вышеупомянутой функции.

**void LoadAccount(Player& user, Player& second, BaseAccount::Store\* list**) – позволяет произвести загрузку игрока(если он есть в списке, и ввёл пароль). Работает в консольном окне.

**void CreateAccount(BaseAccount::Store\* list)** – позволяет создать новый аккаунт. Проверяет есть ли уже аккаунт с таким именем, если да – то создание аккаунта прерывается. Кроме поля имени нужно обязательно ввести пароль.

**void WriteLog(std::string& str)** – записывает результаты боя в лог.

ИГРОВЫЕ ФУНКЦИИ:

**int PvP(Player& player, Player& enemy)** – показывает игровое окно и позволяет играть в режиме игрок против игрока в случае загрузки обоих аккаунтов. Имеет сохранение. Автоматически производит запись статистики.

**int PvP(Player& player, BaseAccount& guest)** – перегрузка вышеупомянутой функции, однако, для случая, когда не вошёл 2 игрок. 1 игрок для игры в этом режиме должен обязательно войти. Имеет сохранение. Автоматически производит запись статистики.

**int PvCPU(Player& player, BaseAccount& CPU**) – предназначена для игры игрока против ПК. Данный тип перегружаемой функции для вошедшего 1го игрока. 2 игрок играть в данный режим не может. Имеет сохранение. Автоматически производит запись статистики.

**int PvCPU(BaseAccount& Guest, BaseAccount& CPU)** – перегруженная вышеупомянутая функция, для не вошедшего 1го игрока. Не имеет сохранения(из-за отсутствия соответствующих полей в обоих объектах). Автоматически производит запись статистики

Все функции данного подраздела способны только выходить из игры.

ВСПОМОГАТЕЛЬНЫЕ ИГРОВЫЕ ФУНКЦИИ

**bool EditCell(char\* cArray, sf::Vector2i& iMousePos, bool bStep)** – проверяет где произошло нажатие в окне. Если это нажатие на пустой ячейки – меняет её значение в соответствии с ходом.

Возвращает true, если смена значения произошла удачно.

Возвращает false, если невозможно изменить значение.

Принимает в параметрах текущий массив ячеек – cArray, место положения курсора мыши в момент нажатия – iMousePos, и логическую переменную «чей ход» - bStep.

**bool WinCell(char\* cArray, bool bStep)** – осуществляет проверку на победу, и возвражает соответствующие значение логической переменной. Принимает в параметрах текущий массив ячеек – cArray и логическую переменную «чей ход» - bStep.

**bool SelectDifficult(void)** – выбор сложности. Салага – случайные ходы, ветеран – ходы с логикой.

**void LogicCPU(char\* cGame, bool bDifficult, bool bStep)** – логика ПК. В зависимости от сложности – ходы случайные, или с логикой.

Пользовательская документация:

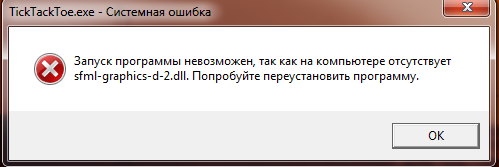
Запуск выполняемого файла должен обязательно выполняться вместе со всеми дочерними файлами.

В случае успешного запуска – появятся 2 окна: консольное окно(подобно командной строке), и окно игры. Если произошла ошибка – будет выведена соответствующая ошибка в консоль, либо в виде сообщения.

Примечание: закрывать консольное окно не нужно после запуска графической оболочки нельзя, так как это приведёт к завершению работы ПО, без сохранения. Для окон графической оболочки предусмотрен возврат в меню и выход из ПО. Закрывать приложение другими способами – не желательно.

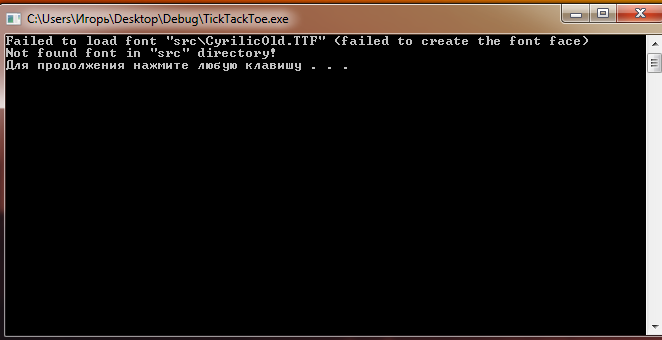
Типы ошибок:

1) Системные:



Возникает в случае отсутствия в директории с ПО соответствующих библиотек. ПО должно быть установлено вместе со всеми дополнительными файлами.

2) Отсутствие файла – для них характерен вывод в консоль, при этом графическая оболочка либо не загружается, либо закрывается. Пример такой ошибки:



Указывает какой файл отсутствует, и в какой директории. В данном случае отсутствует файл «CyrilicOld.TTF» в директории «src».

3) Баги – ошибки, которые не обнаружены на этапе разработки. При их возникновении не выводиться ни каких сообщений, однако они влияют на механику работы ПО. В случае их возникновения писать на электронную почту [igor.nebosenko@gmail.com](mailto:igor.nebosenko@gmail.com).

После запуска ПО будет показано окно-меню. Вы можете выбрать режим игры, однако, желательно войти в свой аккаунт для сохранения своей статистики. Для этого нажмите кнопку «Выбор игроков», далее – выберите «Создать аккаунт», и в консольном окне введите имя игрока и пароль. Если игрок с таким именем уже существует – программа не позволит создать пользователя с таким же именем. Для входа нажмите кнопку под соответствующей учётной записью(1 игрок слева вверху, 2 – справа вверху), после чего программа предложит ввести имя и пароль. Если имя введено неверно – предложение ввести пароль не появиться, а графическая оболочка станет вновь активной.

Список существующих игроков находиться ниже, где игроки расположены в алфавитном порядке, отображено количество их побед, поражений, ничьих.

Режимы игры: предусмотрено 2 основных режима игры: игрок против игрока, и игрок против ПК.

Для игры против другого игрока – 1ый игрок должен обязательно войти. Если 2 игрок не вошёл – будет выведено соответствующее предложение войти, или играть ему под гостевым аккаунтом. Статистика побед\поражений\ничьих – ведётся автоматически.

Для игры против ПК – используется только аккаунт первого игрока. Если вы выбрали этот режим, но не вошли в аккаунт – ПО предложит вам войти, или продолжить как гость.

Также в этом режиме вам предложат выбрать сложность: «Салага» - ходы ПК случайны, «Ветеран» - ПК ходит с логикой.

В игре есть сохранения для всех режимов, кроме режима против ПК если игрок не вошёл. Доступен только выход из игры, а не меню.

Пункт меню «Помощь» - вкратце описывает правила, а также указывает автора игры.

Пункт меню «Логи игр» - позволяет показать кто с кем играл, и результат игры.

Пункт меню «Выход» - выходит из игры.