**Название проекта:**

**Телеграм-бот «pybot»**

**Выполнил:** Горохов Максим Геннадьевич

**Цель проекта:** сделать телеграм-бота для игр в шахматы и крестики-нолики, а также для узнавания погоды в любом городе.

**Используемые библиотеки:**

SQLAlchemy==2.0.9  
 requests==2.28.2  
 pygame==2.3.0  
 python-telegram-bot==20.2

**Как использовать бота:**

Указано в /help(конечно же, для игр с другим игроком нужен ещё юзер, подключенный к боту).

**Классы и функции:**

**Модуль weather:**

def svg\_to\_png # преобразование svg в png, читаемый Telegram-ом

async def weather # взять погоду из указанного города

**Модуль users:**

class User # ORM модель таблицы users

**Модуль db\_sess:**

def global\_init # Инициалицация БД( а если нет – и создание)

def create\_session #Создание сессии

**Модуль db\_command:**

def createdb\_and\_globinit\_if\_not\_exists\_else\_globinit # Если БД нет – создание БД, и инициализация в любом случае

def add\_user # создать юзера

def user\_is\_exists # проверка существования юзера

def get\_chess\_rating # взять рейтинг в шахматах

def get\_xo\_rating # взять рейтинг в крестиках-ноликах

def inc\_chess\_rating # увеличить рейтинг в шахматах на 1

def dec\_chess\_rating # уменьшить рейтинг в шахматах на 1

def inc\_xo\_rating # увеличить рейтинг в крестиках-ноликах на 1

def dec\_xo\_rating # уменьшить рейтинг в крестиках-ноликах на 1

**Модуль db\_sess:**

def global\_init # Если БД нет – создание БД, и инициализация в любом случае

def create\_session # Создать сессию

**Модуль session:**

class Translator # переводчик координат из “человеческих” в “компьютерные” и наоборот

class Board # доска

def play # функция игры

class Piece # фигура

class Knight # конь

class Rook # Ладья

class Bishop # Слон

class Queen # ферзь

class Pawn # пешка

class King # король

**Модуль cfg** # ключи от API

**Модуль xo:**

class Translator # переводчик координат из “человеческих” в “компьютерные” и наоборот

class XOBoard # доска крестики-нолики

def play # функция игры

**Модуль** **xo\_graphic**:

class TextSprite # спрайт буквы

class ShapeSprite # спрайт фигуры(X или O)

class Board # графическая доска

class PlayAI # вызываемый(с \_\_call\_\_) класс игры с ИИ в крестики-нолики

class PlayXO # вызываемый(с \_\_call\_\_) класс игры с другим игроком в крестики-нолики

**Модуль chess\_graphic:**

class PieceSprite # спрайт фигуры

class TextSprite # спрайт буквы

class TimeSprite # спрайт оставшегося времени

class Board # графическая доска

class Play # вызываемый(с \_\_call\_\_) класс игры в шахматы с другим человеком

**Модуль xoAI\_creator:**

class VariantsTreeNode # узел полного дерева вариантов

class XOAI # класс ИИ в крестики-нолики

@dataclass

class VarTreeNode # узел краткого(сжатого) дерева вариантов

def clear\_tree\_1 # функция очистки дерева, когда ИИ первый

def clear\_tree\_2 # функция очистки дерева, когда ИИ второй

def create\_ai # создать ИИ в файл(с помощью pickle.dumps)

def create\_ai\_if\_not\_exists # создать ИИ в файл(с помощью pickle.dumps) если его не существует

**Модуль xoAI:**

class XOAI # класс ИИ в крестики-нолики по сжатому дереву вариантов

def play1 # игра с ИИ, когда ИИ первый

def play2 # игра с ИИ, когда ИИ второй

**Модуль main:**

class MyCycleCorutine # класс – обёртка для корутины, повторяющаяся циклично вместе с другими(ожидание вывода из других процессов)

class Worker # класс – “работник”. При инициализации создаёт поток, а при вызове принимает объект класса MyCycleCorutine

def \_cleaner\_garbage # один проход очистки мусора(законченных игр)

def garbage\_cleaner\_thread # функция потока очистки законченных игр

def cleaner\_garbage # запуск очистки мусора

class XOAIGame # игра в крестики-нолики с ИИ

class HalfXOGame # половина игры с другими игроками в крестики-нолики. Когда появляются две половинки, они объединяются в XOGame

class XOGame # игра в крестики-нолики с игроком

class HalfChessGame # половина игры с другими игроками в шахматы. Когда появляются две подходящие по рейтингам и таймерам половинки, они объединяются в ChessGame

class XOGame # игра в шахматы с игроком

class Input # ввод в игру с таймером

class Output # вывод из игры в шахматы

class OutputXOAI # вывод из игры в крестики-нолики с ИИ

class OutputXO # вывод из игры в крестики-нолики с другим игроком

async def start # обработчик команды «/start»

async def help\_command # обработчик команды «/help»

async def handle # обработчик других команд и ходов игры.

def main # запуск приложения

Скриншоты проекта





