Final Exam Stage

Python Object-Oriented Programming. Unit Testing with TDD

LEARN. GROW. SUCCEED.

by Viktor Ivanchenko / ivanvikvik@gmail.com / Minsk

Final Exam Stage

Объектно-Ориентированное Программирование на языке Python. Модульное тестирование с использованием методологии TDD

Цель работы 

Проверить фундаментальные практические знания ООП с использованием языка Python и UML на примере экзаменационного задания с использованием модульного тестирования и методологии TDD (разработка через тестирование).

Требования

1) Необходимо спроектировать и реализовать UML-диаграмму взаимодействия классов и объектов разрабатываемой программной системы с отображением всех связей (отношений) между классами и объектами.
2) При проектировании и разработке системы необходимо полностью использовать своё объектно-ориентированное воображение и по максимум использовать возможности, которые предоставляет язык Python для реализации ООП-методологии.
3) Основные классы системы должны быть самодостаточными, т.е. не зависеть, к примеру, от консоли! Любые типы отношений между классами должны применяться обосновано и лишь тогда, когда это имеет смысл.
4) При выполнения задания необходимо по максимуму пытаться разрабатывать универсальный, масштабируемый, легко поддерживаемый и читаемый код.

5) Также рекомендуется придерживаться ***Single Responsibility Principle***, ***SRP*** (принципа единственной ответственности): у каждого пакета, класса или метода должна быть только одна ответственность (цель), т.е. должна быть только одна причина изменить в дальнейшем соответствующий блок кода.

6) Создаваемые классы необходимо грамотно разложить по соответствующим пакетам, которые должны иметь «адекватные» названия. Используйте архитектурный шаблона проектирования ***Model-View-Controller***, ***MVC***.

7) В соответствующих компонентах бизнес-логики необходимо предусмотреть «защиту от дурака», а также обработку ошибок и исключений.

8) Рекомендуется избегать использования глобальных переменных при написании основной логики приложения.

9) Любые ошибки или исключения, которые могут возникнуть в процессе работы основной логики-программы должны грамотно отлавливаться и обрабатываться.

10) Для проверки работоспособности и правильности работы основной логики приложения необходимо по максимум покрыть её модульными тестами. Необходимо проверить все тестовые случаи (варианты) работы основной бизнес-логики программы.

11) На базе спроектированной программной системы реализуйте простейшее интерактивное консольное приложение.

12) Программа должна быть снабжена дружелюбным и интуитивно понятным интерфейсом для взаимодействия с пользователем.

13) Интерфейс программы и комментарии должны быть на английском языке.

14) Для вывода результирующих данных необходимо использовать современные средства форматирования языка Python.

15) При разработке программ придерживайтесь соглашений по написанию кода на языке Python!

Best of LUCK with it, and remember to HAVE FUN while you're learning :)
Victor Ivanchenko

Критерии оценивания

1. Полнота описания архитектуры предметной области программы с использованием **UML-диаграммы классов** и её детализации.
2. Логичность и простота именования пользовательских идентификаторов: пакетов, модулей, классов, полей, методов, функций, локальных и глобальных переменных. Использования соглашений по именованию идентификаторов в языке Python.
3. Организация пакетов, модулей в пакетах и классов в соответствующих модулях (рекомендуется для организации структуры проекта использовать архитектурных шаблон проектирования MVC).
4. Обширность использования связей там, где это действительно нужно, между классами: ассоциации, наследования, композиции, агрегации и делегирования. Рекомендуется избегать множественного наследования классов и минимум реализовать трёхуровневую иерархию наследования классов сущностей предметной области.
5. Реализация инкапсуляции: на уровне сущностей с использованием общих свойств (рекомендуется использовать декораторы и встроенный класс ***property***), свойств только для чтения или только для записи, вычислительного характера свойств; на уровне программных слоёв – с использованием высокоуровневых базовых или контейнерных классов. Чтобы обрабатывать сущности предметной области группой, рекомендуется реализовать специальный контейнерный класс или классы, а не в открытую передавать объекты в виде встроенных в язык Python контейнеров (тем самым нарушая принципы инкапсуляции).
6. Полнота содержимого классов-сущностей: конструктор по умолчанию, конструктор с параметрами (соответствующий магический метод ***__init__()***); деструктор (соответствующий магический метод ***__del__()***); метод(ы) для конвертации содержимого объекта в строковый эквивалент (соответствующие магические методы ***__str__()*** и ***__repl__()***); свойства для инкапсуляции состояния; методы для реализации соответствующего поведения; переопределённые методы стандартного поведения, унаследованные от базового класса ***object***; перегруженные стандартные встроенные операции, которые уместны для соответствующей сущности:

- методы математических операций: ***__add__()***, ***__sub__()***, ***__mul__(), __truediv__()*** и т.д.;
- методы операций отношения (сравнения): ***__lt__()***, ***__le__()***, ***__gt__(), __ge__(), __eq__(), __ne__()***;
- другие методы: ***__len__()***, ***__abs__()***, ***__bool__()*** и т.д.

7. Полнота содержимого контейнерных классов: конструктор по умолчанию, конструктор с параметрами; деструктор; метод(ы) для конвертации содержимого контейнерного объекта в строковый эквивалент; свойства для инкапсуляции состояния; методы для реализации соответствующего поведения; методы класса для организации кода уровня класса, если это необходимо; переопределённые методы стандартного поведения, унаследованные от базового класса ***object***; логически уместные перегруженные стандартные операции (арифметические операции, операции отношения, логические операции и т.д.), которые уместны для соответствующей сущности; специальные методы, для облегчения работы именно с контейнерными элементами:
   - методы для работы с элементами контейнерного объекта через индексатор: ***__getitem__(), __setitem__(), __delitem__()***;
   - методы для адаптации контейнерного объекта для циклической конструкции for для перебора элементов: ***__iter__()*** и ***__next__()***;
   - другие методы: ***__len__()***, ***__abs__()***, ***__bool__()*** и т.д.
8. Полнота содержимого функциональных и утилитных (вспомогательных) классов: статические методы для реализации соответствующего поведения бизнес-логики; методы класса для организации кода уровня класса, если это необходимо.
9. Наличие механизмов обработки ошибок и исключительных ситуаций (конструкция ***try-except-else-finally***), которые могут возникнуть в результате работы программы при неправильных действиях пользователя или некорректных обрабатываемых данных. Наличие собственных исключений и их ручной вызов (с использованием оператора ***raise***).
10. Широта обхват и разнообразие модульных тестов, наличие параметризированных тестов (чтобы избавиться от дублирования кода) и фикстур различного уровня (экземпляра и класса) для подготовки испытательного плацдарма и его очистки.

Экзаменационное задание

Необходимо решить задачу с использованием методологии ООП. Для чего необходимо придумать самостоятельно или подобрать соответствующую проблемную (предметную/доменную) область, которая базируется на объектах и событиях реального мира (примеры соответствующих предметных областей приведены ниже). Спроектировать классы (собственные пользовательские типы данных) в языке Python для программного представления данных объектов и основной логики будущей программной системы.

На базе спроектированной программной системы реализовать программу и продемонстрировать её работоспособность.

Предметная область

Автотранспорт (Automobile Transport). Определить иерархию автотранспорта соответствующей предметной области (– это могут быть только самолёты или поезда, автобусы или машины, а может и всё вперемешку, Ваши фантазии никто не ограничивает). Создать автопарк (стоянку, аэропорт, вокзала и т.д.). Подсчитать какие-нибудь общие характеристики всех объектов иерархии предметной области или специфические характеристики отдельно взятых объектов иерархии: стоимость всего автопарка машин, общую пассажировместимость, общее количество перевозимого груза, ...

Дополнительно можно осуществить поиск соответствующего объекта по существенной характеристике (к примеру, самую дорогую/дешёвую машину, или самый длинный/короткий поезд, самый пассажировместимый самолёт и т.д.). Можно также реализовать сортировку объектов доменной области.

Примеры предметных областей

1. **Цветочница или магазин цветов *(Flower Shop)*.** В магазине цветов можно собрать букет из соответствующих цветов. Необходимо определить вес букета, его стоимость и самый дорогой/недорогой цветок (или цветы, если они одинаковы по стоимости).
2. **Игровая комната *(Game room)*.** Подготовить игровую комнату для детей разных возрастных групп. Игрушек должно быть фиксированное количество в пределах выделенной суммы денег. Должны встречаться игрушки родственных групп, например: маленькие, средние и большие машины, куклы, мячи, кубики. Провести сортировку игрушек в комнате по одному из параметров. Найти игрушки в комнате, соответствующие заданному диапазону параметров, также по экстремальным значениям (самую дорогую/дешёвую игрушку, игрушку с максимальной/минимальной длиной или весом и т.д.).
3. **Шеф-повар *(Chef)*.** Определить иерархию овощей. Сделать несколько салатов. Посчитать калорийность каждого из низ, общий вес и т.д. Провести сортировку овощей для салата на основе одного из параметров. Найти овощи в салате, соответствующие заданному диапазону параметров, а также по экстремальным значениям (самый высококалорийный/низкокалорийный овощ, овощ с максимальным/минимальным содержимым витамина С или любых других витаминов и минералов и т.д.).
4. **Шеф-повар *(Chef)*.** Определить иерархию фруктов. Сделать несколько фруктовых салатов. Посчитать калорийность каждого из низ, общий вес и т.д. Провести сортировку фруктов для салата на основе одного из параметров. Найти фрукты в салате, соответствующие заданному диапазону параметров, а также по экстремальным значениям (самый высококалорийный/низкокалорийный фрукт, фрукт с максимальным/минимальным содержимым витамина С или любых других витаминов и минералов и т.д.).
5. **Налоги *(Taxes)*.** Определить множество и сумму налоговых выплат физического лица за год с учетом доходов с основного и дополнительного мест работы, авторских вознаграждений, продажи имущества, получения в подарок денежных сумм и имущества, переводов из-за границы, льгот на детей и материальную помощь. Провести сортировку налогов по сумме.

6. **Кредиты *(Credits)*.** Спроектировать иерархию банковских кредитов. Сформировать набор предложений клиенту по целевым кредитам различных банков для оптимального выбора. Учитывать возможность досрочного погашения кредита и\или увеличения кредитной линии. Реализовать выбор и поиск кредита, соответствующего критериям клиента. Также необходимо подсчитать общую сумму по всем кредитам клиента и найти максимальную ставку по кредиту и самый дорогой кредит клиента.
7. **Вклады *(Deposites)*.** Спроектировать иерархию банковских вкладов. Сформировать набор предложений клиенту по вкладам различных банков для оптимального выбора. Учитывать возможность досрочного снятия кредита и\или пополнения. Реализовать поиск и сортировку вкладов по соответствующим критериям клиента.
8. **IT-фирма (*IT Company*).** Определить иерархию сотрудников. Создать несколько сотрудников, из которых собрать несколько команд для разработки IT-проекта. Определить стоимость каждой из команд (в человека-часах). Провести сортировку и поиск сотрудников на основе одного или нескольких параметров, которые предъявляются заказчиком. Найти сотрудника, соответствующего заданным критериям заказчика, а также по экстремальным значениям (самый популярный тариф и наоборот, тариф с максимальным/минимальным количеством бонусов и т.д.). Также необходимо найти общий фонд заработной платы компании на месяц, а также определить сотрудников с максимальной (минимальной) заработной платой.
9. **Библиотека (*Library*).** Определить иерархию изданий (книги, журналы, альбомы и т.д.). Подсчитать сумму книг (журналов, альбомов, …) или страниц по жанру. Провести сортировку изданий на основе одного или нескольких параметров. Найти издание соответствующее заданному диапазону параметров, а также по экстремальным значениям (самый популярное или редкое издание и наоборот, издание с максимальным/минимальным количеством страниц или шрифта и т.д.). Также библиотека предоставляет своим читателям книги (и другую литературу), которые можно почитать как в самой библиотеки, так и взять с собой на определённый период. Необходимо дополнительно подсчитать, сколько сейчас книг на руках у читателей, а также определить самую (менее) популярную книгу у читателей.

10. **Мобильная связь *(Mobile Communication)*.** Определить иерархию тарифов мобильной компании. Создать список тарифов компании. Посчитать общую численность клиентов. Провести сортировку тарифов на основе одного из параметров. Найти тариф в компании, соответствующий заданному диапазону параметров, а также по экстремальным значениям (самый популярный тариф и наоборот, тариф с максимальным/минимальным количеством бонусов и т.д.). Также необходимо определить общую сумму дохода компании в месяц, которую она получает от клиентов за использования соответствующих тарифных планов, а также найти клиента, который заплатил больше (меньше) других.
11. **Видео или компьютерный Герой (*Game Hero*).** Определить иерархию артефактов, которые улучшают отдельные характеристики героев. Наделить героев различными супер способностями. Посчитать общее состояние соответствующих характеристик героев, на которых повлияли артефакты и супер способности, а также выявить главную характеристику героя. Провести сортировку артефактов на основе одного из параметров. Найти героев, соответствующие заданному диапазону характеристик или способностей, а также по экстремальным значениям (самый сильный/слабый герой, герой с большим/маленьким интеллектом и т.д.). Дополнительно подсчитать общую стоимость артефактов, а также величину его соответствующих способностей.
12. **Новогодняя ёлка *(Christmas tree)*.** Есть новогодняя ёлка, которую можно украсить соответствующими новогодними игрушками. Необходимо подсчитать вес всей ёлки вместе с игрушками, её общую стоимость, а также найти самую дорогую игрушку.
13. **Жилищно-Коммунальное Хозяйство, ЖКХ *(Housing and Communal Services)*.** В ЖКХ предлагают набор соответствующих услуг по обслуживанию и эксплуатации жилищного хозяйства клиента. Необходимо подсчитать общую стоимость услуг, которые были оказаны клиенту ЖКХ и найти самую дорогую (недорогую) услугу.
14. **Турагентство и Туристические путевки *(Tourist trips)*.** В турфирме можно сформировать набор предложений клиенту по выбору туристической путевки различного типа. Необходимо подсчитать общую сумму, которую клиент должен заплатить за выбранные им путёвки и найти самую выгодную по стоимости путёвки, исходя из расчёта стоимости за один день.

15. **Таксопарк *(Taxi)*.** Определить иерархию легковых автомобилей. Создать таксопарки. Посчитать стоимость всех машин таксопарка. Провести сортировку автомобилей парка по расходу топлива и другим значимым характеристикам. Найти автомобиль в компании, соответствующий заданному диапазону параметров, а также по экстремальным значениям (самый большой/маленький таксопарк, самая дорогая/дешёвая машина таксопарка и т.д.).
16. **Авиакомпания *(Airline)*.** Определить иерархию самолетов. Создать авиакомпании. Посчитать общую вместимость и грузоподъемность. Провести сортировку самолетов компании по дальности полета и другим существенным характеристикам. Найти самолет в компании, соответствующий заданному диапазону параметров, а также по экстремальным значениям (самый пассажировместимый самолёт или наоборот, самолёт с максимальной/минимальной грузоподъёмностью и т.д.).
17. **Камни *(Stones)*.** Определить иерархию драгоценных и полудрагоценных камней. Отобрать камни для ожерелья. Посчитать общий вес (в каратах) и стоимость ожерелья. Провести сортировку камней ожерелья на основе одного из параметров. Найти камни в ожерелье, соответствующие заданному диапазону параметров, а также по экстремальным значениям (самый дорогой/дешёвый камень, камень с максимальным/минимальным весом и т.д.).
18. **Звукозапись *(Sound Recording)*.** Определить иерархию музыкальных композиций. Записать на диск сборку данных композиций. Подсчитать общую продолжительность диска. Провести перестановку (сортировку) композиций диска на основе одного из параметров. Найти композицию, соответствующую заданному диапазону параметров, а также по экстремальным значениям (самую продолжительную композицию и наоборот, самую громкую/тихую и т.д.).
19. **Новогодний подарок *(New Year Gift)*.** Определить иерархию конфет и прочих сладостей. Создать несколько объектов-конфет. Собрать детский подарок с определением его содержимого и веса. Оценить стоимость подарка. Провести сортировку конфет в подарке на основе одного из параметров. Найти сладость в подарке, соответствующую заданному диапазону параметров, а также по экстремальным значениям (самую дорогую/дешёвую сладость, сладость с минимальным/максимальным количеством сахара или калорий и т.д.).

20. **Пассажирский железнодорожный транспорт *(Railway Transport, Passenger Train)*.** Определить иерархию подвижного состава железнодорожного транспорта. Создать пассажирские поезда. Посчитать общую длину каждого поезда, общую численность пассажиров, вес багажа и т.д. Осуществить поиск поездов по заданным экстремальным характеристикам (самый пассажировместимый поезд или наоборот, поезд с максимальным/минимальным багажом и т.д.).
21. **Грузовой железнодорожный транспорт *(Railway Transport, Cargo Train)*.** Определить иерархию подвижного состава железнодорожного транспорта. Создать грузовые поезда. Посчитать общую длину каждого поезда, общую грузоподъёмность и т.д. Осуществить поиск поездов по заданным экстремальным характеристикам (самый длинный/короткий поезд, поезд с максимальной/минимальной грузоподъёмностью и т.д.).
22. **Автосалон *(Car Center)*.** Есть автосалон, который состоит из соответствующих машин. Необходимо подсчитать общую стоимость машин автосалона и найти самую дорогую (недорогую) машину.
23. **Страховое агентство *(Insurance Company)*.** Есть страховая фирма, которая предлагает страховые услуги и обязательства своим клиентам. Необходимо подсчитать общую стоимость страховых услуг, оказываемых конкретному клиенту, а также найти самую дорогую (недорогую) услугу.
24. **Грузоперевозки *(Cargo Transportation)*.** Есть транспортная компания, которая занимается грузовыми перевозками. Необходимо подсчитать максимальное количество грузов, которое за один раз может осуществить компания, исходя из имеющего собственного грузового транспорта, а также найти транспорт, который перевозит максимальное (минимальное) количество груза.
25. **Пассажирские перевозки *(Passenger Operations)*.** Есть транспортная компания, которая занимается пассажирскими перевозками. Необходимо подсчитать максимальное количество пассажиров, которых за один раз может перевезти компания, исходя из имеющего собственного пассажирского транспорта, а также найти транспорт, который перевозит максимальное (минимальное) количество пассажиров.

26. **Фургон кофе *(Coffee Car)*.** Загрузить фургон определенного объема грузом на определенную сумму из различных сортов кофе, находящихся в тому же в разных физических состояниях (зерно, молотый, растворимый в банках и пакетиках). Учитывать объем кофе вместе с упаковкой. Провести различные сортировки товаров на основе соответствующих параметров. Найти товар в фургоне, соответствующий заданному диапазону параметров, а также по экстремальным значениям (самую дорогое/дешёвое кофе, с максимальной/минимальной обжаркой и т.д.).
27. **Рыцарь (Knight).** Определить иерархию амуниции рыцаря. Экипировать рыцаря. Посчитать стоимость. Провести сортировку амуниции на основе одного из параметров. Найти элементы амуниции, соответствующие заданному диапазону параметров, а также по экстремальным значениям (самую дорогую/дешёвую амуницию, амуницию с максимальной/минимальной разрушительной силой или бронёй и т.д.).
28. **Футбольный Менеджер *(Football Manager System)*.** Есть футбольные клубы, у которых есть соответствующие характеристики, влияющие на исход встречи с противником. Необходимо определить шансы футбольной команды на победу против команды-соперника, а также общие шансы на победу в кубке и чемпионате соответствующей страны.
29. **Вклады или кредиты *(Deposits or Credits)*.** Банки предлагают своим клиентам соответствующие вклады или кредиты с различными процентами. Необходимо определить общую сумму дохода соответствующего банка в месяц, которую он получает согласно соответствующей марже. Найти самые дорогие (недорогие) кредиты с учётом всех платежей или самые выгодные (невыгодные) вклады с учётом процентной ставки и доходности по ним.
30. **Спортивная рыбалка *(Sports Fishing)*.** В соревнованиях по спортивной рыбалке собираются команды из нескольких рыбаков. В процессе соревнований каждая из команд за определённое время должна наловить максимальное количество рыбы. Необходимо определить общий улов всех команд, а также найти команду победителя (команда, которая имеет наибольший улов рыбы) и аутсайдера (команда, которая имеет самые скромные результаты по улову).

31. **Зоопарк *(Zoo)*.** Есть зоопарк животных, которые ОЧЕНЬ любят кушать. Необходимо определить общий суточный запас продуктов, необходимых зоопарку, чтобы прокормить всех своих животных. Также необходимо найти самых (менее) прожорливых животных зоопарка.
32. **Домашние электроприборы *(House Equipments)*.** Определить иерархию бытовых электроприборов, создать окружение (дом, квартиру, ...) для них и включить некоторые из них в розетку. Посчитать потребляемую мощность всех приборов созданного окружения, а также только тех, которые включенный в данный момент и работают. Провести сортировку приборов в окружении на основе одного из параметров. Найти прибор в квартире, соответствующий заданному критерию, а также по экстремальным значениям (самый дорог/дешёвый электроприбор, прибор с максимальной/минимальной мощностью и т.д.).
33. **Пчелиная пасека (*Bee Apiary*).** На пасеке есть несколько домиков с ульями пчёл. Каждая пчела собирает за день определённое количество мёда и приносит его в свой улей. Необходимо подсчитать общее количество мёда в день, которая даёт вся пасека, а также определить улей, которые даёт больше (меньше) всего мёда.
34. **Поликлиника (*Clinic/Hospital*).** В городе есть несколько поликлиник (больниц), куда обращаются пациенты, у которых есть проблемы со здоровьем. Необходимо подсчитать общее количество обращений во все поликлиники (больницы) города, а также найти самую (менее) загруженную по посещению поликлинику (больницу).
35. **Тюрьма (*Jail/Prison*).** Есть тюрьма, в которой содержаться заключённые, осужденные по разным статьям и на различный срок. Необходимо подсчитать суммарное количество дней (или месяцев, или лет, ...) всех заключённых, а также найти самую (менее) популярную уголовную статью, за которую отбывают наказание осуждённые, или найти самый большой (короткий) срок заключённого.
36. **Морской порт *(Sea Port)*.** Есть морской порт с пирсами (причалами), на которых каждый день происходит выгрузка и загрузка соответствующих грузов. Необходимо подсчитать общий грузопоток, который проходит через данный порт за сутки (или любой другой период), а также определить максимально (минимально) загруженный пирс (причал).

37. **Суд или судебное делопроизводство (*Court*).** В городе есть суд, который каждый день рассматривает уголовные дела или другие правонарушения, касающиеся нарушения закона. Необходимо подсчитать: общее количество дел, которые были рассмотрены судом за отчётный период; сколько из общих дел были оправдательными, а сколько – с доказанной виной и повлёкшим к заключению под стражу виновных. Можно дополнительно подсчитать общее число лиц, которые были осуждены на соответствующие сроки и сейчас находятся в местах отбывания наказания, а также найти осуждённых, которым судья установил максимальный (минимальный) срок.
38. **Видео/Компьютерная игра (Video/Computer *Game*)**. Есть крутая игра, направленная на зарабатывание игроками в процессе игры определённого количества «качества» – это может быть соответствующие баллы, время, уровни, виртуальные деньги, алмазы, брильянты, золото, другие виртуальные ресурсы и т.д. Необходимо определить максимальное количество «качества», заработанное всеми игроками в данной видео игре, а также выявить победителя (– игрок с максимальным количеством «качества») и аутсайдера (– игрок с минимальным количеством «качества») на данный момент.
39. **Продуктовый магазин или любой другой магазин (*Grocery Store*)**. Есть магазин, в который каждый день приходят покупатели за продуктами. Необходимо подсчитать среднюю выручку магазина за сутки (или за месяц, или год и т.д.), а также найти покупателя, который оставил в магазине само больше (меньше) денег за отчётный период.
40. **Ресторан (*Restaurant*)**. Есть ресторан соответствующей кухни, который своим посетителям предоставляет разнообразное меню блюд. Необходимо подсчитать общую сумму заказа соответствующего столика или общее количества денег, который ресторан получил за отчётный период (день, месяц и т.д.). Также необходимо найти чек (столик) с максимальной (минимальной) суммой заказа.
41. **Автостоянка (*Parking*)**. В городе есть автостоянка (или несколько автостоянок), стоимость машиномест на которой тарифицируется по часам. Необходимо подсчитать общую сумму, которую получает автостоянка за сутки (или любой другой отчётный период), а также автовладельца, который суммарно заплатил больше (меньше) всего денег за парковку.

42. **Футбольный клуб.** Есть разноплановые футбольные игроки с различными характеристиками по выносливости, скорости, силе удара и т.д., а также игровые характеристики – количество забитых голов, голевых передач и т.д. Необходимо из имеющихся множества игроков собрать футбольную команду, рассчитать её специальные показатели. После «сыгранной игры» показать соответствующую статистику по игрокам: сколько кто пробежал, дал голевых пассов, забил голов и т.д.