Факультет компьютерных технологий и прикладной математики Кафедра вычислительных технологий 02.03.02

Паттерны программирования Лабораторная работа № 2. Объекты и классы. YAML. Массив объектов. Инкапсуляция. Ассоциация

Каждое задание должно быть загружено на личный gitрепозиторий отдельным коммитом. Лабораторная работа выполняется в одной папке. Защита работы возможна на любой лабораторной работе от 1 до 16. Каждое из шести заданий проверяется отдельно с учетом вопросов преподавателя. Задание засчитывается отдельно, лабораторная работа зачтена в случае выполнения всех 6 заданий.

Если часть задач выполнена в один коммит, работа не проверяется. Если все коммиты сделаны в один час, работа не проверяется.

Часть заданий выполняется по вариантам.

Все задания дальше будут представлять из себя выполнение одного общего программного продукта. Необходимо отследить прием на должности, увольнение и смену должностей в некоторой абстрактной фирме с несколькими отделами. В каждом отделе есть должность руководителя, перечень выполняемых работ и штатное расписание (набор должностей). Одного из сотрудников выбирают как контактное лицо. В базе хранится информация о соискателях должностей с их трудовыми книжками, соискателей и сотрудников можно принять на работу, уволить с должности. Хранится информация о зарплате. Зарплата может быть только оклад, может быть фиксированная премия в процентах от общей месячной зарплаты, может быть фиксированная надбавка в рублях, может быть выплачиваемая в отдельных случаях премия в процентах от оклада. Может быть что-то одно, может чего-то одного не быть, может быть голый оклад, может быть все вместе, может быть штраф в процентах от оклада. Для отдела должна быть возможность упорядочить по зарплате в текущем месяце. Должна быть получить список высокооплачиваемых возможность самых низкооплачиваемых сотрудников. Работа выполняется в рамках паттерна MVC.

Задание 1. Объекты и классы

Задачи

1. Создать класс Department с полями объекта название, контактный телефон. Написать конструктор, геттеры и сеттеры. Избежать дублирования кода. Рассмотреть

- способы реализации геттеров и сеттеров с помощью символов.
- 2. Создать несколько объектов, вывести информацию о них на экран. Продумать корректный способ вывода информации о текущем состоянии объекта на экран.
- 3. Создать поле выполнение обязанностей. Реализовать методы добавить обязанность, выделить обязанность, удалить обязанность, получить текст выделенной обязанности, заменить текст выделенной обязанности, вывести список обязанностей на экран. Объяснить, почему с точки зрения основных принципов ООП нежелательно получать доступ к массиву целиком. Продумать, каким должен быть В ЭТОМ случае конструктор и внести изменения.
- 4. Создать объект класса Department, добавить, изменить, удалить несколько обязанностей, отследить корректность выполнения методов. Внести коррективы в отображение объекта на экране так, чтобы была возможность отслеживать состояние поля, содержащего обязанности.
- 5. Добавить метод КЛАССА, проверяющий, является ли некоторая строка телефонным номером. Модифицировать класс так, чтобы в произвольный момент времени не мог существовать объект с непозволительной строкой в поле номер телефона. Протестировать полученный класс.

Вопросы.

- а. Что такое класс, что такое объект, как создать объект класса?
- b. В чем заключается принцип инкапсуляции? Как получить доступ к полям объекта из внешнего класса?
- с. Как используются символы для решения задач инкапсуляции и уменьшения количества кода при описании класса?
- d. Опишите структуру классов языка Ruby, как в нее вписывается написанный Вами класс?
- е. Что такое конструктор, зачем он нужен, как описывается конструктор в произвольном классе?
- f. Какие методы объекта обязательно есть у любого написанного Вами класса, опишите, что они делают.
- g. Что такое метод класса, в чем его отличие от метода объекта? Приведите два практических примера, когда введение метода класса вы считаете необходимым согласно концепциям ООП.

Задание 2. Понятие сериализации. Задачи.

1. Продумайте структуру текстового файла, в котором будут храниться объекты класса Department. Напишите такой файл.

- 2. Напишите метод, который читает read_from_txt, который читает несколько отделов из текстового файла и возвращает массив объектов класса Department.
- 3. Написать метод, который выводит массив объектов класса Department на экран.
- 4. Написать метод write_to_txt, который записывает массив объектов класса Department в файл txt.
- 5. Прочитать массив объектов, вывести на экран, добавить еще один отдел, вывести результат на экран, записать измененный массив в тот же файл.
- 6. Написать метод write_to_YAML, который записывает массив объектов класса Department в файл YAML.
- 7. Напишите метод, который читает read_from_YAML, который читает несколько отделов из YAML файла и возвращает массив объектов класса Department.
- 8. Прочитать массив объектов, вывести на экран, добавить еще один отдел, вывести результат на экран, записать измененный массив в тот же файл.

Вопросы.

- 1. Что такое сериализация, для чего она нужна?
- 2. Опишите принципы YAML сериализации в ruby.

Задание 3. Массив объектов.

Задачи.

Прежде, чем перейти к решению задач, рассмотрим простой вопрос. Предположим, что мы обязаны реализовать структуру классов, в которой будет сериализация и десериализация объектов класса Department из YAML и ТХТ файлов. Где необходимо писать методы, в которых сериализация будет реализована?

- 1. Создать класс Department_list, в котором будет поле массив объектов класса Department. Создать коструктор, инициализирующий это поле.
- 2. Написать метод add_note, добавляющий запись, метод choose_note, выделяющий запись, метод change_note, заменяющий выбранную запись метод get_note, возвращающий выбранную запись и delete_note, удаляющий выбранную запись.
 - 3. Реализовать методы сериализации и десериализации.
- 4. Построить конструктор, читающий объект из YAML и коструктор, читающий объект из ТХТ. Как реализовать такие кострукторы, при условии, что переопределение конструкторов в ruby невозможно?
 - 5. Построить метод, сортирующий записи по названию. Вопросы.
- 1. Какой принцип ООП диктует необходимость выделения в отдельный класс массива объектов? Поясните свой ответ?

- 2. В чем суть отсутствия геттера для массива?
- 3. Опишите способы создания нескольких конструкторов, приведите примеры.

Задание 4. Агрегация. Композиция. Задания.

- 1. Создайте класс должность Post с полями отдел, название, оклад, должность вакантна или нет. Реализуйте геттеры, сеттеры и конструктор. Объясните принцип построения. Реализуйте способ вывода должности на экран.
 - 2. По аналогии с классом Department_list напишите класс Post_list. Вопросы.
- 1. Что такое ассоциация, каковы ее виды, в чем разница между ними?

Задание 5. Агрегация и композиция.

Задания.

- 1. В класс Department добавьте поле Post_list. Обновите конструктор.
- 2. Напишите методы добавить должность, выбрать должность, удалить должность, изменить должность.
- 3. Напишите метод, который позволит получить все вакантные должности этого отдела.
- 4. Продумайте метод получения всех должностей данного отдела, обоснуйте выбор.
- 5. Протестируйте написанные методы, покажите все построенные возможности, расскажите, какие возможности теперь доступны при работе с Вашими классами.
- 5. Перепишите методы сериализации/десериализации всех описанных классов так, чтобы метод класса Department_list вызывал соответствующий метод класса Department, который вызывал метод класса Post_list, который вызывает соответствующий метод класса Post.

Задание 6. Агрегация и композиция. Задания.

- 1. Реализуйте возможность сортировки отделов по количеству вакантных должностей, модифицируйте все необходимые классы, обоснуйте свой выбор.
- 2. Напишите методы, которые позволяют просматривать содержимое объекта класса Department в урезанной форме и полную информацию. Определить, что будет отображаться в урезанной форме(урезанной формой будем полагать данные, отображаемые в случае отображения списка отделов).