МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ярославский государственный университет им. П.Г. Демидова

Кафедра вычислительных и программных систем

УТВЕРЖДАЮ
Декан факультета ИВТ
_____ Д.Ю. Чалый
«_24_» ___мая_____ 2022 г.

Рабочая программа дисциплины
«Промышленная разработка веб-приложений»

Направление подготовки
01.03.02 Прикладная математика и информатика

Направленность (профиль)
«Программирование и технологии искусственного интеллекта»

Квалификация выпускника
Бакалавр

Форма обучения
очная

Программа рассмотрена
на заседании кафедры
от 22 марта 2022 г.,
протокол № 7

Программа одобрена НМК
факультета ИВТ
протокол № 6 от
18 апреля 2022 г.

Ярославль

1. Цели освоения дисциплины

Целями дисциплины «Промышленная разработка веб-приложений» являются изучение средств создания приложений различного уровня сложности, предоставляемых современными скриптовыми языками, получение практических навыков в использовании скриптовых языков.

2. Место дисциплины в структуре ОП бакалавриата

Дисциплина «Промышленная разработка веб-приложений» относится к вариативной части ОП бакалавриата. В курсе рассматриваются вопросы разработки приложений на современных скриптовых языках с учётом их специфических особенностей.

Содержание курса тесно связано фактически со всеми дисциплинами, которые изучались студентами. Освоению данной программы предшествуют учебные курсы по программированию и современным информационным технологиям.

Дисциплина «Промышленная разработка веб-приложений» обеспечивает закрепление и углубление теоретических знаний и практических навыков по основным дисциплинам ИТ-цикла. Дисциплина способствует профессиональному росту студентов, повышению их общеметодологического уровня, а также дальнейшему развитию навыков научно-исследовательской деятельности.

3. Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения ОП бакалавриата

Процесс изучения дисциплины направлен на формирование следующих элементов компетенций в соответствии с ФГОС ВО, ОП ВО и приобретения следующих знаний, умений, навыков и (или) опыта деятельности:

Формируемая компетенция (код и формулировка)	**Индикатор достижения компетенции (код и формулировка)**	**Перечень планируемых результатов обучения**
Профессиональные компетенции		
ПК -3 Способен к разработке и проектированию программного обеспечения, к использованию современных технологий программирования	ПК – 3.1 Знает основы разработки алгоритмических и программных решений системного и прикладного программного обеспечения; структуру и виды системного и прикладного программного обеспечения; основные принципы модульного, объектно - ориентированного и событийного программирования	Знать: ● алгоритмы хранения и обработки данных; ● средства стандартных библиотек скриптового языка. Уметь: ● проектировать и реализовывать консольные и графические приложения с использованием скриптового языка; ● использовать инструментарий средств разработки программного обеспечения; ● изменять существующие приложения и их компоненты в соответствии с системными принципами. Владеть: ● приёмами современного объектно-ориентированного

		программирования; ● приёмами современного функционального программирования.

4. Объем, структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 3 зач. ед., 108 акад. час.

№ п/п	Темы (разделы) дисциплины, их содержание	Семестр	Виды учебных занятий, включая самостоятельную работу студентов, и их трудоемкость (в академических часах)						Формы текущего контроля успеваемости Форма промежуточной аттестации (по семестрам)
			Контактная работа						
			лекции	практические	лабораторные	консультации	аттестационные испытания	самостоятельная работа	
1.	Введение в скриптовые языки программирования	4	2		5,4	1		3	Лабораторная работа
2.	Особенности работы с контейнерами	4	2		5,4			3	
3.	Управление зависимостями при создании приложений	4	2		5,4	1		3	Лабораторная работа
4.	Описание собственных классов в языке программирования Ruby	4	2		5,4	1		3	Лабораторная работа
5.	Модель наследования в языке программирования Ruby	4	2		5,4			3	
6.	Использование стандартных типов Ruby	4	2		5,4	1		3	Лабораторная работа
7.	Создание статических веб-приложений	4	2		5,4	1		3	Лабораторная работа
8.	Создание динамического веб-приложения с использованием библиотеки Sinatra	4	2		5,4	1		3	Лабораторная работа

9.	Предоставление данных веб-приложения для внешних приложений	4	1		5, 4	1		3	Лабораторная работа
10.	Автоматическое тестирование консольных и веб-приложений	4	1		5, 4			1,7	
	Всего за 4 семестр		**18**		**54**	**7**		**28,7**	**Зачет**
	Всего		**18**		**54**	**7**		**28,7**	

Содержание разделов дисциплины:

1. Введение в скриптовые языки программирования. Понятие динамической типизации, отличие от статической типизации, отличие интерпретируемых и компилируемых языков программирования, компиляция во время исполнения. Основные элементы синтаксиса языка Ruby: определение методов, строк, особенности определения имён переменных, работа с массивами и ассоциативными массивами, символы и управляющие конструкции.
2. Особенности работы с контейнерами в Ruby, определение блоков и итераторов. Обзор основных методов контейнеров, массивов и ассоциативных массивов, их основные методы и семантика их работы. Определение понятия блока как анонимного метода, ассоциированного с вызываемым методом, контекст, захватываемый при определении блока. Определение собственных итераторов.
3. Управление зависимостями при создании приложений на Ruby. Подключение библиотек, входящих в стандартную поставку интерпретатора Ruby. Использование приложения `gem` для управления списком установленных пакетов, просмотр, установка и удаление пакетов. Использование приложения `bundler` для описания наборов зависимостей для приложения, фиксация конкретных версий используемых пакетов. Рассмотрение стандартной структуры каталогов и директорий в проекте на языке программирования Ruby.
4. Описание собственных классов в языке программирования Ruby. Требования к определению имени классов и файлов, в которых данные классы описаны. Использование стандартного метода инициализации, описание переменных экземпляра, атрибутов класса, методов и переменных классов.
5. Модель наследования в языке программирования Ruby. Классическое наследование классов от классов, переопределение методов, вызов родительских методов. Модули в Ruby, использование модулей для описания пространства имён, для определения наборов методов. Использование модулей в виде примесей для расширения функциональности классов.
6. Использование стандартных типов Ruby: числа, строки, регулярные выражения и последовательности. Использование стандартных итераторов чисел и строк для решения типовых задач. Использование собственных классов в качестве частей последовательностей.
7. Создание статических веб-приложений. Краткий обзор возможностей языка разметки страниц HTML: ссылки, блочные и строчные элементы для отображения содержимого. Использование CSS-фреймворка Zurb Foundation для изменения отображения элементов и добавления динамических элементов в HTML-страницы.
8. Создание динамического веб-приложения с использованием библиотеки Sinatra. Разделение логики приложения согласно шаблону модель-вид-контроллер. Настройка окружения для разработки, динамической перезагрузки кода контроллеров. Описание вида с использованием языка шаблонов ERB.

9. Предоставление данных веб-приложения для внешних приложений. Рассмотрение стандарта структурированного описания ресурсов, рассмотрение схем предоставления ресурсов. Предоставление документов нестандартного типа в качестве ответов контроллеров Sinatra-приложений. Получение документов в качестве тела запроса Sinatra-приложениям.
10. Автоматическое тестирование консольных и веб-приложений. Введение в понятие автоматического тестирования, рассмотрение различных видов тестирования, включая модульное, интеграционное и системное. Написание модульных тестов с использованием библиотеки MiniTest. Написание системных тестов для веб-приложений с использованием библиотеки Capybara.

5. Образовательные технологии, используемые при осуществлении образовательного процесса по дисциплине

В процессе обучения используются следующие образовательные технологии:

Лекция-беседа или «диалог с аудиторией», является наиболее распространенной и сравнительно простой формой активного вовлечения студентов в учебный процесс. Эта лекция предполагает непосредственный контакт преподавателя с аудиторией. Преимущество лекции-беседы состоит в том, что она позволяет привлекать внимание студентов к наиболее важным вопросам темы, определять содержание и темп изложения учебного материала с учетом особенностей студентов.

Мастер-класс – это особая форма учебного занятия, когда преподаватель-мастер передает свой опыт путем прямого и комментированного показа последовательности действий, методов, приемов и форм педагогической деятельности. Целью проведения мастер-класса является профессиональное, интеллектуальное и эстетическое воспитание студентов, и прежде всего, развитие в ходе мастер-класса способности студента самостоятельно и нестандартно мыслить.

Лабораторная работа – организация учебной работы с реальными материальными и информационными объектами, экспериментальная работа с аналоговыми моделями реальных объектов.

Практическое занятие – занятие, посвященное освоению конкретных умений и навыков и закреплению полученных на лекции знаний.

6. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень лицензионного программного обеспечения и информационных справочных систем (при необходимости)

В процессе осуществления образовательного процесса используются: для разработки документов, презентаций, для работы с электронными таблицами

OfficeStd 2013 RUS OLP NL Acdmc 021-10232

LibreOffice (свободное)

издательская система LaTeX;

– Среда разработки NetBeans 8.2 : www.netbeans.org. Доступ свободный

– OS Linux (свободная)

– для поиска учебной литературы библиотеки ЯрГУ – Автоматизированная библиотечная информационная система "БУКИ-NEXT" (АБИС "Буки-Next").

7. Перечень основной и дополнительной учебной литературы, ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

а) основная:

1. Крокфорд, Д., JavaScript : сильные стороны / Д. Крокфорд; [пер. с англ. А. Лузгана], СПб., Питер, 2013, 173с

2. Фримен, Э., Изучаем программирование на JavaScript / Э. Фримен, Э. Робсон; [пер. с англ. Е. Матвеева], СПб., Питер, 2017, 638с

б) дополнительная:

1.Лагутина, Н. С., Разработка программных приложений : практикум для студентов, обучающихся по направлению Фундаментальная информатика и информационные технологии / Н. С. Лагутина, Ю. А. Ларина, А. М. Васильев; Яросл. гос. ун-т., Ярославль, ЯрГУ, 2014, 71 с.

2. Лагутина, Н. С., Разработка программных приложений [Электронный ресурс] : практикум для студентов, обучающихся по направлению Фундаментальная информатика и информационные технологии / Н. С. Лагутина, Ю. А. Ларина, А. М. Васильев; Яросл. гос. ун-т., Ярославль, ЯрГУ, 2014, 71 с. http://www.lib.uniyar.ac.ru/edocs/iuni/20140402.pdf

в) ресурсы сети «Интернет»

- JavaScript для разработчиков https://developer.mozilla.org/ru/docs/Web/JavaScript Доступ свободный

- JavaScript документация http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf Доступ свободный

– Среда разработки NetBeans 8.0.3: www.netbeans.org. Доступ свободный

Электронно-библиотечная система «Юрайт»(https://urait.ru/).

Электронно-библиотечная система «Лань»(https://e.lanbook.com/).

8. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине включает в свой состав специальные помещения:

-учебные аудитории для проведения занятий лекционного типа и практических занятий (семинаров);

-учебные аудитории для проведения лабораторных занятий;

- учебные аудитории для проведения групповых и индивидуальных консультаций,

- учебные аудитории для проведения текущего контроля и промежуточной аттестации;

-помещения для самостоятельной работы;

-помещения для хранения и профилактического обслуживания технических средств обучения.

Специальные помещения укомплектованы средствами обучения, служащими для представления учебной информации большой аудитории.

Для проведения занятий лекционного типа предлагаются наборы демонстрационного оборудования и учебно-наглядных пособий, хранящиеся на электронных носителях и обеспечивающие тематические иллюстрации, соответствующие рабочим программам дисциплин.

Помещения для лабораторных занятий и самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду организации.

Число посадочных мест в лекционной аудитории больше либо равно списочному составу потока, а в аудитории для практических занятий (семинаров) – списочному составу группы обучающихся.

Автор(ы) :

Старший преподаватель кафедры ВПС _____ А.М.Васильев

Приложение №1 к рабочей программе дисциплины
«Промышленная разработка веб-приложений»
Фонд оценочных средств
для проведения текущей и промежуточной аттестации студентов
по дисциплине

1. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

1.1. Контрольные задания и иные материалы, используемые в процессе текущей аттестации

Примеры заданий для лабораторных работ

Примеры лабораторных работ по теме «Введение в скриптовые языки программирования»

Исследуем условный оператор if

Напишите приложение, которое будет просить пользователя ввести вещественное число.

Если число больше нуля, тогда вывести “положительное число”.

Если число меньше нуля, тогда вывести “не положительное, а отрицательное число”.

Если число равну нулю, тогда вывести “непонятное число”.

Приложение должно выводить информацию о работе, приветствие для ввода данных. Реализуйте приложение с помощью условного оператора `if`, реализуйте проверку на правильность ввода.

Игра на рынке

Напишите приложение, которое будет считывать положительное вещественное число - объём изначальных инвестиций. Затем предложите пользователю варианты: подождать на рынке и снять. Если пользователь решит снять деньги, то приложение завершает свою работу и выводит информацию о текущем состоянии счёта (выигрыш на рынке).

Если пользователь решил остаться на рынке, то происходит “игра”. В зависимости от случайного значения, которое вернёт метод `rand(15)` выполните:

Больше 14 - повысьте счёт на 10%

Больше 12 - повысьте счёт на 2%

Больше 9 - оставьте счёт неизменным

Больше 7 - отнимите со счёта 2%

Больше 5 - отнимите со счёта 10%

Остальное - отнимите со счёта 50%

По окончании игры приложение должно показать пользователю текущее состояние счёта и спросить: подождать на рынке или снять деньги со счёта.

Приложение должно выводить информацию о своей работе, приветствие для ввода данных, проверку на правильность ввода. Реализуйте приложение с помощью оператора ветвления `case`.

Заполнение формы

Реализуйте приложение, которое позволит пользователю ввести информацию о себе в форму на приём работы. Приложение должно анализировать данные, введённые в

форму и предлагать на их основании педлагать различные должности, которые может занять кандидат.

Пользователь вводит следующие данные:

Имя и фамилию, непустая строка.

Адрес электроной почты, непустая строка.

Возраст, положительное число от 15 до 100.

Стаж работы в профессии, положительное число.

Приложение должно показать справку об использовании, показывать приглашения для ввода данных и проверять их корректность.

После ввода данных приложение должно проверять данные и показывать список должностей, на которые может претендавать кандидат. Для проверки и вывода свойств используйте модификаторы `if` и `unless`.

Профессия «Руководитель», если имя и фамилия совпадает с «Пётр Петрович».

Профессия «Инженер», если адрес электронной почты включает в себя строку “code”. Проверяйте с помощью метода `String#include?`.

Профессия «Стажёр», если стаж работы в профессии меньше двух лет.

Профессия «Бывалый», если возраст от 45 до 60.

Добавьте к професии слово “Заслуженный”, если стаж работы в профессии больше 15 лет.

Добавьте к професси слово “Известный”, если стаж работы в профессии больше 5 лет.

Примеры лабораторных работ по теме «Управление зависимостями при создании приложений»

Использование низкоуровнего инструмента установки библиотек gem

Цель задачи - научиться искать джемы по названию, использовать инструментарий `gem` для просмотра и управлением установленными джемами.

Подзадачи:

Найдите все библиотеки, в названиях которых есть `json`.

Найдите все библиотеки, в названиях которых есть `xlsx`.

Посмотрите список установленных библиотек.

Установите библиотеку `write_xlsx`.

Какие джемы помимо `write_xlsx` были установлены в систему?

Удалите джем `write_xlsx`. Были ли удалены все джемы, что были поставлены вместе с ним?

Использование высокоуровнего инструмента управления зависимостями.

Цель задачи - научиться использовать Bundler для управления зависимостями приложения.

Шаги по решению задачи.

Создайте каталог для приложения и прейдите в него.

Установите последнюю версию джема Bundler.

Создайте файл конфигурации Gemfile, выполнив `bundle init`.

Добавьте в список зависимостей библиотеку `chunky_png`.

Установите и зафиксируйте последнюю версию библиотеки с помощью Bundler.

Создайте файл `main.rb`, в котором реализуйте приложение, которое считывает последовательность точек из CSV-файла и строит изображение ломанной прямой.

Дополнительно отображайте координатную сетку.

Дополнительно подстраивайте размер изображения под размеры точек.

Для запуска приложения используйте команду `bundler exec ruby main.rb`.

Установите и зафиксируйте последнюю версию джема `rubocop` с помощью Bundler.

Проверьте исходный код приложения `main.rb` с помощью Rubocop и исправьте все возникшие ошибки.

Пример лабораторной работы по теме «Описание собственных классов в языке программирования Ruby»

Начальное состояние состояние YAML-документа, содержащего данные для задания, приведено ниже:

```
students:
  - id: 0
    name: Ivan
    surname: Loskutov
  - id: 2
    name: Maria
    surname: Dovlatova
  - id: 5
    name: Alice
    surname: Dilbert
courses:
  - name: Biology
    grades:
      - student_id: 0
        grade: 5.3
      - student_id: 2
        grade: 4.5
      - student_id: 5
        grade: 5.0
  - name: Math
    grades:
      - student_id: 5
        grade: 4.6
      - student_id: 0
        grade: 5.1
      - student_id: 2
        grade: 5.2
  - name: Computer Science
  - name:
```

Приложение должно считывать информацию из YAML-документа и позволять пользователю:

Просматривать список предметов. Приложение должно выводить список предметов и средний балл по ним.

Просматривать все текущие оценки по выбранному предмету. Приложение должно выводить название предмета, а затем выводить в виде таблицы имя студента и его оценку.

Выставлять оценки известным студентам по предметам, уже добавленным в информационное хранилище.

По окончании работы приложение должно сохранять всю информацию в файл, чтобы работа пользователя была сохранена на жёстком диске.

Дополнительно рекомендуется также реализовать следующие функции:

Добавление нового студента в список студентов.

Редактирование информации о студенте из списка.

Добавление нового предмета в список предметов.

Редактирование названия предмета в списке.

Удаление предмета из списка предметов.

Удаление студента из списка. *Внимание* При удалении студента необходимо также удалять и всего его оценки, выставленные по предметам.

Технические требования к реализации приложения.

Приложение должно показывать краткую справку об использовании при старте.

Приложение должно всегда показывать приглашение ко вводу данных со стороны пользователя.

Приложение должно корректно обрабатывать ситуацию, когда пользователь ввёл некорректное значение в поле ввода.

Приложение должно быть оформлено согласно структуре джема.

Исходный код приложения должен быть проверен с помощью Rubocop и все его сообщения должны быть исправлены.

Примеры лабораторных работ по теме «Использование стандартных типов Ruby»

Вводные задачи на использование чисел и строк

Разработать приложение, которое сможет найти все числа, которые без остатка делятся на 5, 6 и 13. Начало и конец диапазона передаются в качестве аргументов приложения.

Разработать приложение, которое выводит строковое название для чисел от 1 до 100. В качестве параметра приложению передаётся последовательность чисел, разделённых запятыми.

Разработать приложение, которое позволяет подсчитать количество “котиков” в тексте. Имя файла, содержащее текст с “котиком” передаётся в качестве аргумента приложения. Результат обработки текста следует выводить на стандартный поток вывода.

Простейший калькулятор

Создайте простейший калькулятор для работы с рациональными числами. В качестве аргумента приложению передаётся выражение типа `рациональное число операция рациональное число`. Выражение записывается целиком в кавычках. Необходимо реализовать следующие операции: сложение, вычитание, умножение и деление. Рациональное число описывается следующим образом: `целое число / целое число`. Количество пробелов между элементами значения не имеет. Результат вычисления выводить в формате рационального и вещественного чисел.

Работа с диапазонами и собственными типами данных

Создайте класс “Студенческая группа”, которая определяется следующими параметрами:

Короткое название специализации.

Длинное название специализации.

Год обучения (1, 2, 3 и т.д.).

Номер группы.

Тип обучения: магистры или бакалавры.

Добавьте в класс все методы, которые ему необходимы для использования в диапазонах. Номер группы не участвует в Переход из бакалавриата в магистратуру не является автоматическим.

Правила сортировки:

Любые магистры “больше” любых бакалавров.

Сравнение различных специализаций происходит по длинному названию.

Сравнение групп в рамках специализации происходит по году обучения и номеру группы.

С помощью данного класса:

Отобразите группы, в которые в обычной ситуации будет вхоить студенты, поступившие в группы:

ИВТ-11БО

ПИЭ-12БО

ИТ-11МО

В поход собрались студенты из групп ИТ-42БО, ПМИ-21БО, ИТ-12МО, ПИЭ-32БО. Помогите Даше определить группу, которая будет ответствена за данный поход. Имена групп передаются как параметры приложению.

Пример лабораторной работы по теме «Создание статических веб-приложений»

В рамках задания мы создаём сайт для размещения собственных статей на техническую тематику (блог). Основная задача данного сайта состоит в том, чтобы Вы смогли делиться с другими и обсуждать информацию о полезных с Вашей точки зрения технологиях, практиках, инструментах. Дополнительно этот сайт должен быть вашей визитной карточкой, с которой другие люди смогут узнать о Ваших достижениях и компетенциях.

Сайт должен состоять из следующих страниц:

Главная страница. На ней должен отображаться список статей (блог-постов), размещённых на сайте. На вашем сайте должна находится как минимум одна статья, можно больше.

Визитная карточка. Краткая информация о себе. Информация может включать список различных достижений (научных работ (публикаций), приложений, выступлений на конференциях, награды и т.п.). Этот список содержит ваши личные достижения. Можно указывать любые, но в разумных пределах.

Статья, посвящённая разработке первой лабораторной работы. Размер статьи - не менее 400 слов.

Все страницы должны содержать шапку, в которой должны быть ссылки на основные разделы сайта (главная страница, визитная карточка).

На главной странице должен выводиться список аннотаций статей. Аннотация включает в себя название, описание и ссылку на страницу с полным содержанием. При нажатии на ссылку должна открываться страница с полным содержимым статьи. В вашем случае будет всего лишь одна аннотация, ведущая на нужную страницу.

Пример лабораторной работы по теме «Создание динамического веб-приложения с использованием библиотеки Sinatra»

Необходимо создать сайт, позволяющий пользователям решать простые задачи из задачника. Сайт помимо решения этих задач, позволяет пользователю узнать информацию о его создателе.

Сайт должен содержать в себе следующие страницы:

Главная страница, содержащая ссылки на все другие страницы, а также описание решаемой задачи.

Страница для ввода данных для решения задачи №2 из пункта 1.2 задачника.

Страница для показа результата решения задачи №2.

Страница для показа списка результатов решения всех задач.

Страница для показа информации о создателе приложения.

Общий макет страниц должен содержать навигационную панель, с которой можно перейти на главную страницу, страницу ввода информации, страницу со списком результатов, а также страницу для показа информации о создателе сайта.

Главная страница должна содержать описание назначения данной системы.

Страница для ввода данных должна содержать в себе форму для ввода данных. На форме должно быть три поля для ввода числа и кнопка для выполнения запроса на вычисление. Желательно упомянуть на этой странице полное условие задачи. Поля для ввода должны иметь осмысленные метки, а не `a`, `b` и `c`.

Дополнительно, если пользователь ввёл неправильные значения в поля ввода, то приложение должно сообщать пользователю об этом.

Страница для вывода решения должна содержать в себе не только результат вычисления, но также и введённые данные. Таким образом человек должен легко воспроизвести процедуру вычисления.

Страница для вывода списка решений должна предоставлять краткую информацию о введённх данных и результатах, а также предлагать пользователю перейти на страницу для вывода решения для каонкретного результата.

Страница о создателе приложения должна содержать имя, фамилию, номер группы и фотографию создателя.

Пример лабораторной работы по теме «Разработка приложений с поддержкой множества пользователей»

Создайте сайт, который будет решать следующие задачи:

Пользователь может зарегистрироваться на сайте, указав псевдоним и пароль. На форме регистрации вы должны проверить:

Что пользователя с таким псевдонимом не существует на данном сайте.

Что длина пароля больше, чем 7 символов.

Что пароль и его повтор совпадают.

Пользователь может пройти авторизацию на сайте, указав псевдоним и пароль. После успешной авторизации пользователю необходимо показать список пользователей.

Если пользователь указал несуществующий псевдоним, об этом надо оповестить.

Если пользователь указал неверный пароль, об этом надо оповестить.

Авторизованный пользователь может просматривать список зарегистрированных пользователей. Неавторизованный пользователь не должен иметь доступ к данному списку. Список должен быть отсортирован по имени пользователя.

Просмотр списка происходит постранично. Т.е. одновременно на странице не должно быть показано более 5 существующих пользователей.

Внизу страницы должен располагаться блок перехода между страницами, если на сайте сейчас зарегистрировано более 5 пользователей.

Для указания конкретной страницы используйте параметры запроса.

Авторизованный пользователь может выйти из системы. В этом случае доступ ко списку пользователей для него должен быть ограничен.

Пример лабораторной работы по теме «Предоставление данных веб-приложения для внешних приложений»

Реализуйте веб-приложение, которое будет ориентировано на предоставление информации через REST API. Передача данных должна осуществляться в формате JSON-документов. Для тестирования работы веб-приложения также напишите второе приложение, которое будет обращаться к вашему веб-приложению и выполнять на нём необходимые запросы.

Веб-приложение должно оперировать списком задач. **Дополнительно** можете сделать так, чтобы списки задач были уникальны для каждого пользователя. Задача описывается следующими полями:

Название задачи. *Обязательное поле*. Формат: обычный текст длиной не более 280 символов.

Срок выполнения задачи. *Обязательное поле*. Формат: `ГГГГ-ММ-ДД`, например `2018-06-23`.

Описание задачи. *Необязательное поле*. Формат: обычный текст.

Исполнитель задачи. *Необязательное поле*. Формат: `Имя Фамилия <почтовый адрес>`, например `Пётр Петрович <sidor@mail.com>`.

Статус задачи. *Обязательное поле*. Формат: одна из констант `new`, `in progress`, `completed`.

Веб-приложение должно обеспечивать выполнение всех действий над ресурсом задач, включая:

Отображение всех незавершённых на настоящее время задач, `get /tasks`.

Отображение всех задач, `get /tasks?status=any`.

Добавление новой задачи в список `post /tasks`.

Получение одной задачи по номеру в списке `get /tasks/10`.

Обновление одной задачи по номеру в списке `put /tasks/10`.

Удаление одной задачи по номеру в списке `delete /tasks/10`.

Для проверки работоспособности данного веб-приложения создайте консольное приложение, которое будет выполнять следующие сценарии.

Отобразить текущий список задач, которые запланированы у пользователя.

Отобразить все задачи, которые запланированы у пользователя.

Добавить новую задачу.

Обновить существующую задачу.

При реализации сценариев клиент не должен выполнять сложной предварительной обработки данных. Достаточно их получить, упаковать в запрос и отправить на сторону веб-приложения. Если в запросе содержаться ошибки, то они должны быть обработаны на стороне сервера и возвращены клиентскому приложению. Последнее может показать их конечному пользователю.

Требования к лабораторным работам

Требования к лабораторной работе по консольным приложениям

Требования к знаниям учащихся

- Уровень знакомства с синтаксисом языка Ruby
  - Запуск приложения
  - Переменные, методы
  - Работа контейнерами, итераторами, нумераторами
  - Описание собственных классов
  - Наследование классов
  - Использование модулей и примесей
  - Знание деталей работы чисел
  - Знание деталей вызова методов и указания аргументов
- Уровень знакомства с экосистемой Ruby
  - Установка, удаление зависимостей с помощью `gem`
  - Управление зависимостями с помощью Bundler
  - Проверка качества исходного кода с помощью Rubocop

Требования к выполнению задания

- Приложение должно реализовать все требования, указанные в задании. Однако, к защите допускаются приложения, реализующие только часть задач.
- Исходный код приложения должен быть проверен с помощью приложения Rubocop с настройками по-умолчанию или с конфигурацией, предложенной на соответствующем занятии. Для каждого нарушения, которое находит Rubocop,

должно быть объяснение почему оно не было исправлено. Допускаются аргументы с точки зрения архитектуры приложения, другого рода аргументы не принимаются.
- Все зависимости приложения должны управляться с помощью Bundler, установка зависимостей с помощью `gem` запрещена. Исключением является джем `bundler`.
- Для разрабатываемого приложения должны быть создан отдельный каталог.
  - В корне каталога должны располагаться конфигурационные файлы: `Gemfile`, `Gemfile.lock`.
  - В корне каталога приложения должны находится каталоги `bin`, `lib`, в которых должен находиться исходный код приложения. В каталоге `bin` находится исполняемый файл, в каталоге `lib` находится описание всех собственных классов и модулей.
- Весь код приложения должен быть разбит на модули и классы, запрещается использовать файлы с простым набором методов, которые не пренадлежат какому-либо модулю.
- Необходимо выделить отдельные классы, описывающие предметную область.
- В одном модуле нельзя совмещать логику обработки предметной области и операции ввода-вывода.
- Приложение должно корректно обрабатывать неправильный ввод от пользователя.

Требования к лабораторным работам по веб-приложениям

- Приложение должно реализовать все требования, указанные в задании. Однако, к защите допускаются приложения, реализующие только часть задач.
- Требования должны быть реализованы как функции веб-приложения, даже если в самом задании сказано, что необходимо разработать настольное приложение.
- Исходный код приложения должен быть проверен с помощью приложения Rubocop с настройками по-умолчанию. Для каждого нарушения, которое находит Rubocop, должно быть объяснение почему оно не было исправлено. Допускаются аргументы с точки зрения архитектуры приложения, другого рода аргументы не принимаются.
- Все зависимости приложения должны управляться с помощью Bundler, установка зависимостей с помощью `gem` запрещена.
- Для разрабатываемого приложения должны быть создан отдельный каталог.
  - В корне каталога должны располагаться конфигурационные файлы: `Gemfile`, `Gemfile.lock`, `Rakefile`.
  - Для веб-приложений необходимо создать файл `config.ru` и обеспечить запуск с помощью приложения `rackup`.
- Весь код приложения должен быть разбит на модули и классы, запрещается использовать файлы с простым набором методов, которые не пренадлежат какому-либо модулю. Это требование не накладывается на файл-приложения Sinatra.
- Необходимо выделить отдельные классы, описывающие предметную область.
- В одном модуле нельзя совмещать логику обработки предметной области и операции ввода-вывода.
- Приложение должно корректно обрабатывать неправильный ввод от пользователя.
- Для приложения необходимо написать системные тесты, которые будут проверять работу всех сценариев работы приложения. Системные тесты не должны влиять на степень покрытия тестами.

Требования к знаниям учащихся

- Уровень знакомства с синтаксисом языка Ruby:
  - Запуск приложения из командной строки.
  - Переменные, методы, блоки.
  - Работа контейнерами, итераторами, нумераторами.
  - Описание собственных классов.
  - Наследование классов.
  - Использование модулей и примесей.
  - Использование стандартных классов чисел и диапазонов.
  - Использование стандартных классов строк, регулярных выражений.
  - Знание деталей вызова методов и указания его аргументов.
- Уровень знакомства с экосистемой Ruby:
  - Установка, удаление зависимостей с помощью `gem`.
  - Управление зависимостями проекта с помощью Bundler.
  - Проверка качества исходного кода с помощью Rubocop.
- Уровень знакомства с разработкой веб-приложений с помощью библиотеки Sinatra:
  - Обработка запросов GET для получения информации.
  - Использование шаблонов Foundation / Bootstrap для оформления сайта.
  - Обработка запросов POST на редактирование информации.
  - Сохранение данных пользователя в настройках приложения.
  - Передача аргументов в контроллер через адресную строку.
  - Написание системных тестов для проверки функциональности приложений.

Критерии оценивания лабораторных работ

Оценка	**Критерии**
Отлично Уровень формирования компетенций: высокий	ОПК-2: Выполнены все описанные выше требования к заданию. Знает принципы построения приложений. Выбирает оптимальные средства и инструменты разработки. Умеет работать с документацией. Анализирует поставленную задачу, использует при этом несколько источников информации: учебники, статьи, лекционные материалы.
Хорошо Уровень формирования компетенций: продвинутый	ОПК-2: Выполнены почти все описанные выше требования к заданию, нарушено не более одного – трех пунктов из каждого набора требований. Знает принципы построения приложений. Выбирает оптимальные средства и инструменты разработки для большинства используемых алгоритмов. Умеет работать с документацией. Использует при решении задачи в основном лекционный материал, но может пользоваться другими источниками информации, возможно не полностью понимая качество получаемого решения.
Удовлетворительно Уровень формирования компетенций: пороговый	ОПК-2: Выполнены описанные выше требования к заданию, нарушено не более половины пунктов из каждого набора требований. Знает принципы построения приложений. Выбирает подходящие средства и инструменты разработки, возможно не достаточно эффективные. Работает с документацией с затруднениями. Использует при решении задачи в только лекционный материал. Другие источники информации воспринимает с трудом или не пытается анализировать.
Неудовлетворительно	ОПК-2: Описанные выше требования к заданию не выполнены по большинству пунктов из каждого набора требований. Не знает

	принципы построения приложений. Не может выбрать средства и инструменты разработки. Не умеет работать с документацией. Не умеет использовать лекционный материал, а также другие источники информации для решения поставленной задачи.

Список вопросов к зачету:

1. Понятие динамической типизации, отличие от статической типизации, отличие интерпретируемых и компилируемых языков программирования, компиляция во время исполнения.
2. Основные элементы синтаксиса языка Ruby: определение методов, строк, особенности определения имён переменных, работа с массивами и ассоциативными массивами, символы и управляющие конструкции.
3. Особенности работы с контейнерами в Ruby, определение блоков и итераторов.
4. Обзор основных методов контейнеров, массивов и ассоциативных массивов, их основные методы и семантика их работы.
5. Определение понятия блока как анонимного метода, ассоциированного с вызываемым методом, контекст, захватываемый при определении блока.
6. Определение собственных итераторов.
7. Подключение библиотек, входящих в стандартную поставку интерпретатора Ruby.
8. Рассмотрение стандартной структуры каталогов и директорий в проекте на языке программирования Ruby.
9. Требования к определению имени классов и файлов, в которых данные классы описаны.
10. Использование стандартного метода инициализации, описание переменных экземпляра, атрибутов класса, методов и переменных классов.
11. Классическое наследование классов от классов, переопределение методов, вызов родительских методов.
12. Модули в Ruby, использование модулей для описания пространства имён, для определения наборов методов.
13. Использование модулей в виде примесей для расширения функциональности классов.
14. Использование стандартных типов Ruby: числа, строки, регулярные выражения и последовательности.
15. Использование стандартных итераторов чисел и строк для решения типовых задач.
16. Использование собственных классов в качестве частей последовательностей.
17. Создание статических веб-приложений.
18. Создание динамического веб-приложения с использованием библиотеки Sinatra.
19. Разделение логики приложения согласно шаблону модель-вид-контроллер.
20. Описание вида с использованием языка шаблонов ERB.
21. Рассмотрение стандарта структурированного описания ресурсов, рассмотрение схем предоставления ресурсов.
22. Предоставление документов нестандартного типа в качестве ответов контроллеров Sinatra-приложений. Получение документов в качестве тела запроса Sinatra-приложениям.
23. Определение автоматического тестирования, рассмотрение различных видов тестирования, включая модульное, интеграционное и системное.
24. Написание модульных тестов с использованием библиотеки MiniTest.
25. Написание системных тестов для веб-приложений с использованием библиотеки Capybara.

Методические указания по выставлению зачета

Зачет выставляется по результатам выполнения лабораторных работ на оценку не ниже удовлетворительно. Лабораторные работы должны быть сданы в течение семестра последовательно в процессе освоения материала или на зачете. В случае необходимости преподаватель в ходе сдачи лабораторных работ может провести беседу по вопросам к зачету.

2. Перечень компетенций, этапы их формирования, описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкалы оценивания

2.1. Шкала оценивания сформированности компетенций и ее описание

Оценивание уровня сформированности компетенций в процессе освоения дисциплины осуществляется по следующей трехуровневой шкале:

Пороговый уровень - предполагает отражение тех ожидаемых результатов, которые определяют минимальный набор знаний и (или) умений и (или) навыков, полученных студентом в результате освоения дисциплины. Пороговый уровень является обязательным уровнем для студента к моменту завершения им освоения данной дисциплины.

Продвинутый уровень - предполагает способность студента использовать знания, умения, навыки и (или) опыт деятельности, полученные при освоении дисциплины, для решения профессиональных задач. Продвинутый уровень превосходит пороговый уровень по нескольким существенным признакам.

Высокий уровень - предполагает способность студента использовать потенциал интегрированных знаний, умений, навыков и (или) опыта деятельности, полученных при освоении дисциплины, для творческого решения профессиональных задач и самостоятельного поиска новых подходов в их решении путем комбинирования и использования известных способов решения применительно к конкретным условиям. Высокий уровень превосходит пороговый уровень по всем существенным признакам.

2.2. Перечень компетенций, этапы их формирования, описание показателей и критериев оценивания компетенций на различных этапах их формирования

Код компе-тенции	Форма контроля	Этапы форми-рования (№ темы (раздела)	Показатели оценивания	Шкала и критерии оценивания компетенций на различных этапах их формирования		
				Пороговый уровень	Продвинутый уровень	Высокий уровень
Общепрофессиональные компетенции						
ОПК-2	Лабораторная работа. Зачет.	1-7	Знать: ● синтаксис и семантику скриптового языка, предложенного к изучению; ● средства стандартных библиотек языка. Уметь: ● проектировать и реализовывать консольные и графические приложения с использованием скриптового языка; ● использовать инструментарий средств разработки программного обеспечения; ● изменять существующие приложения и их компоненты в	Студент должен знать синтаксис и семантику одного скриптового языка; средства стандартных библиотек. Студент должен уметь проектировать и реализовывать консольные и графические приложения с использованием скриптового языка, использовать инструментарий средств разработки программного обеспечения; Студент должен владеть приёмами объектно-ориентированного и функционального программирования.	Студент должен знать синтаксис и семантику нескольких скриптовых языков, предложенных к изучению; средства стандартных библиотек изучаемых языков, выбирать наиболее подходящие средства для решения поставленной задачи. Студент должен уметь проектировать и реализовывать консольные и графические приложения с использованием скриптового языка, использовать инструментарий средств разработки программного обеспечения; изменять существующие приложения и их компоненты. Студент должен владеть	Студент должен знать синтаксис и семантику нескольких скриптовых языков, предложенных к изучению; средства стандартных библиотек изучаемых языков, оценивать эффективность их применения, выбирать наиболее подходящие средства для решения поставленной задачи. Студент должен уметь проектировать и реализовывать консольные и графические приложения с использованием скриптового языка, удовлетворяющие стандартам и стиля программирования; эффективно и грамотно использовать инструментарий средств разработки программного обеспечения; изменять существующие приложения и их компоненты в соответствии с системными принципами.

			соответствии с системными принципами. Владеть: ● приёмами современного объектно-ориентированного программирования; приёмами современного функционального программирования.		приёмами современного объектно-ориентированного и функционального программирования.	Студент должен владеть приёмами современного объектно-ориентированного программирования; приёмами современного функционального программирования.

3. Методические рекомендации преподавателю по процедуре оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Целью процедуры оценивания является определение степени овладения студентом ожидаемыми результатами обучения (знаниями, умениями, навыками и (или) опытом деятельности).

Процедура оценивания степени овладения студентом ожидаемыми результатами обучения осуществляется с помощью методических материалов, представленных в разделе «Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций»

3.1 Критерии оценивания степени овладения знаниями¸ умениями, навыками и (или) опытом деятельности, определяющие уровни сформированности компетенций

Пороговый уровень (общие характеристики):

- владение основным объемом знаний по программе дисциплины;
- знание основной терминологии данной области знаний, стилистически грамотное, логически правильное изложение ответа на вопросы без существенных ошибок;
- владение инструментарием дисциплины, умение его использовать в решении стандартных (типовых) задач;
- способность самостоятельно применять типовые решения в рамках рабочей программы дисциплины;
- усвоение основной литературы, рекомендованной рабочей программой дисциплины;
- знание базовых теорий, концепций и направлений по изучаемой дисциплине;
- самостоятельная работа на практических и лабораторных занятиях, периодическое участие в групповых обсуждениях, достаточный уровень культуры исполнения заданий.

Продвинутый уровень (общие характеристики):

- достаточно полные и систематизированные знания в объёме программы дисциплины;
- использование основной терминологии данной области знаний, стилистически грамотное, логически правильное изложение ответа на вопросы, умение делать выводы;
- владение инструментарием дисциплины, умение его использовать в решении учебных и профессиональных задач;
- способность самостоятельно решать сложные задачи (проблемы) в рамках рабочей программы дисциплины;
- усвоение основной и дополнительной литературы, рекомендованной рабочей программой дисциплины;
- умение ориентироваться в базовых теориях, концепциях и направлениях по изучаемой дисциплине и давать им сравнительную оценку;
- самостоятельная работа на практических и лабораторных занятиях, участие в групповых обсуждениях, высокий уровень культуры исполнения заданий.

Высокий уровень (общие характеристики):

- систематизированные, глубокие и полные знания по всем разделам дисциплины;
- точное использование терминологии данной области знаний, стилистически грамотное, логически правильное изложение ответа на вопросы, умение делать обоснованные выводы;

- безупречное владение инструментарием дисциплины, умение его использовать в постановке и решении научных и профессиональных задач;
- способность самостоятельно и творчески решать сложные задачи (проблемы) в рамках рабочей программы дисциплины;
- полное и глубокое усвоение основной и дополнительной литературы, рекомендованной рабочей программой дисциплины;
- умение ориентироваться в основных теориях, концепциях и направлениях по изучаемой дисциплине и давать им критическую оценку;
- активная самостоятельная работа на практических и лабораторных занятиях, творческое участие в групповых обсуждениях, высокий уровень культуры исполнения заданий.

3.2 Описание процедуры выставления оценки

В зависимости от уровня сформированности каждой компетенции по окончании освоения дисциплины студенту выставляется оценка «зачтено», «незачтено».

Показатели и критерии, используемые при выставлении оценки подробно описаны в разделе «Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций».

Высокий уровень формирования компетенций соответствует оценке «отлично» за лабораторные работы.

Продвинутый уровень формирования компетенций соответствует оценке «хорошо» за лабораторные работы.

Пороговый уровень формирования компетенций соответствует оценке «удовлетворительно» за лабораторные работы.

Оценка «отлично» выставляется студенту, у которого каждая компетенция (полностью или частично формируемая данной дисциплиной) сформирована на высоком уровне.

Оценка «хорошо» выставляется студенту, у которого каждая компетенция (полностью или частично формируемая данной дисциплиной) сформирована не ниже, чем на продвинутом уровне.

Оценка «удовлетворительно» выставляется студенту, у которого каждая компетенция (полностью или частично формируемая данной дисциплиной) сформирована не ниже, чем на пороговом уровне.

Оценка «неудовлетворительно» выставляется студенту, у которого хотя бы одна компетенция (полностью или частично формируемая данной дисциплиной) сформирована ниже, чем на пороговом уровне.

Оценка «зачет» выставляется студенту, у которого каждая компетенция (полностью или частично формируемая данной дисциплиной) сформирована не ниже, чем на пороговом уровне.

Оценка «незачтено» выставляется студенту, у которого хотя бы одна компетенция (полностью или частично формируемая данной дисциплиной) сформирована ниже, чем на пороговом уровне.

Приложение №2 к рабочей программе дисциплины
«Промышленная разработка веб-приложений»

Методические указания для студентов по освоению дисциплины

Занятия проводятся в интерактивной форме с использованием мультимедиа-технологий. Занятия предполагают наличие дискуссий по поводу тех или иных вопросов разработки программных приложений осуществляемых в результате соответствующего предложения преподавателя.

Практическое применение полученных знаний отрабатывается и во время лабораторных занятий, ориентированных помимо закрепления лекционного материала на разбор различных модельных ситуаций, характерных для разработки современных мобильных сервисов. Для успешного освоения дисциплины очень важно решение достаточно большого количества задач, как в аудитории, так и самостоятельно в качестве домашних заданий. В основном такими задачами являются лабораторные работы различного объема, а так же небольшие задачи связанные с исправлением ошибок и доработкой программ. Примеры решения задач разбираются на занятиях. Для решения всех задач необходимо знать и понимать лекционный материал. Поэтому в процессе изучения дисциплины рекомендуется регулярное повторение пройденного лекционного материала. Материал, законспектированный на лекциях, необходимо дома еще раз прорабатывать и при необходимости дополнять информацией, полученной на консультациях, лабораторных занятиях или из учебной литературы.

Текущий контроль успеваемость осуществляется в форме выполнения лабораторных работ по основным понятиям и концепциям курса, осуществляемый в ходе лабораторных занятий. Для самостоятельной работы студентам предлагается разрабатывать собственные идеи приложений с применением разобранных во время лекций и лабораторных занятий подходов и методик. Окончательная аттестация осуществляется в форме зачета, основную часть которого составляют результаты лабораторных работ, а также собеседование по тематике курса.

Учебно-методическое обеспечение самостоятельной работы студентов по дисциплине

Для самостоятельной работы особенно рекомендуется использовать учебную литературу, указанную в разделе № 7 данной рабочей программы.

Также для подбора учебной литературы рекомендуется использовать широкий спектр интернет-ресурсов:

1. Электронно-библиотечная система «Университетская библиотека online» (www.biblioclub.ru) - электронная библиотека, обеспечивающая доступ к наиболее востребованным материалам-первоисточникам, учебной, научной и художественной литературе ведущих издательств (*регистрация в электронной библиотеке – только в сети университета. После регистрации работа с системой возможна с любой точки доступа в Internet.).

2. Для самостоятельного подбора литературы в библиотеке ЯрГУ рекомендуется использовать:

1. Личный кабинет (http://lib.uniyar.ac.ru/opac/bk_login.php) дает возможность получения on-line доступа к списку выданной в автоматизированном режиме литературы, просмотра и копирования электронных версий изданий сотрудников университета (учеб. и метод. пособия, тексты лекций и т.д.) Для работы в «Личном кабинете» необходимо зайти на сайт Научной библиотеки ЯрГУ с любой точки, имеющей доступ в Internet, в пункт меню «Электронный каталог»; пройти процедуру авторизации, выбрав вкладку «Авторизация», и заполнить представленные поля информации.

2. Электронная библиотека учебных материалов ЯрГУ

(http://www.lib.uniyar.ac.ru/opac/bk_cat_find.php) содержит более 2500 полных текстов учебных и учебно-методических материалов по основным изучаемым

дисциплинам, изданных в университете. Доступ в сети университета, либо по логину/паролю.

3. Электронная картотека «Книгообеспеченность»

(http://www.lib.uniyar.ac.ru/opac/bk_bookreq_find.php) раскрывает учебный фонд научной библиотеки ЯрГУ, предоставляет оперативную информацию о состоянии книгообеспеченности дисциплин основной и дополнительной литературой, а также цикла дисциплин и специальностей. Электронная картотека «Книгообеспеченность» доступна в сети университета и через Личный кабинет.