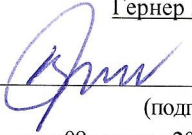


Автономная некоммерческая образовательная организация дополнительного профессионального
образования «Учебный Центр Информационные Бизнес Системы»
(АНО ДПО «УЦ ИБС»)

Утверждаю
Директор АНО ДПО «УЦ ИБС»
Гернер В.А.


(подпись)
«09» января 2024 г.



ОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА

«Java-разработчик. Middle Developer»

ДЛЯ ОБУЧЕНИЯ В РАМКАХ ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
(ПРОФЕССИОНАЛЬНАЯ ПЕРЕПОДГОТОВКА)

Программист
(Код 06.001)

1. ОБЩИЕ ПОЛОЖЕНИЯ

Дополнительная профессиональная программа по профессиональной переподготовке «Java-разработчик. Middle Developer» разработана на основе следующих нормативных документов:

- Федеральный закон от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации»;
- Приказ Министерства образования и науки Российской Федерации от 1 июля 2013 г. № 499 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам»;
- Профессиональный стандарт «Программист» №4, утвержденный приказом Министерства труда и социальной защиты Российской Федерации от 20.07.2022 № 424н.

2. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

2.1 Целью преподавания дисциплины является

Освоение профессии «Программист» (Java-разработчик).

2.2. Задачи изучения дисциплины:

- Изучить ключевые концепции ООП и функционального программирования, а также эффективного использования языка для поддержки сложного кода;
- Овладеть инструментами экосистемы Java для создания бизнес-приложений и понимания архитектуры современных Java-программ;
- Освоить методы многопоточности и технологий, позволяющие приложениям работать под высокими нагрузками.

2.3 Связь дисциплины с другими учебными дисциплинами

Дисциплина "Java-разработчик. Middle Developer" тесно связана с курсами по основам программирования и алгоритмам, что позволяет применять базовые концепции при разработке сложных программных решений. Знания о системах управления базами данных и веб-технологиях необходимы для создания полноценных бизнес-приложений на Java, что углубляет понимание экосистемы разработки.

3. СОДЕРЖАНИЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Образовательная программа «Java-разработчик. Middle Developer» представляет собой комплексный курс для современного Java-разработчика. Программа охватывает все современные аспекты промышленной разработки на Java, начиная с ООП и функционального программирования, погружает в Spring Framework и работу с базами данных, и включает необходимые профессиональному разработчику темы многопоточности, ввода-вывода и архитектуры REST-сервисов.

Категория слушателей программы:

- Java-разработчики;
- практикующие разработчики на других языках программирования (C/C++/PHP/C# и т.д.), желающих освоить язык Java;
- тестировщики с уверенным знанием Java.

К освоению учебной программы допускаются:

- лица, имеющие среднее профессиональное и (или) высшее образование;
- лица, получающие среднее профессиональное и (или) высшее образование.

Базовые компетенции, которыми должен владеть слушатель программы:

- Основы программирования: знание базовых концепций программирования, таких как переменные, операторы, условия и циклы;
- Знание языка Java: базовые навыки работы с языком Java, включая синтаксис, структуры данных и основные библиотеки;
- Алгоритмическое мышление: умение разрабатывать простые алгоритмы и решать базовые задачи с использованием логического мышления;
- Работа с системами управления версиями: знание основ работы с системами контроля версий, такими как Git, для управления кодом и совместной работы;

- Базовые навыки работы с базами данных: понимание основ реляционных баз данных и базовых операций с ними, таких как создание, чтение, обновление и удаление данных (CRUD).

4. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Программа «Java-разработчик. Middle Developer» позволит углубить свои знания языка Java, получить опыт решения сложных задач и подготовиться к сдаче сертификации в IBS или к сдаче Oracle сертификации OCPJP.

После завершения обучения слушатели смогут:

1. Формировать JavaDoc-документацию;
2. Читать базовые типы UML-диаграмм;
3. Разрабатывать и запускать Java-приложения;
4. Использовать в приложениях примитивные и объектные типы;
5. Использовать в приложениях ключевые операторы языка;
6. Использовать в приложениях абстрактные классы и интерфейсы;
7. Применять при проектировании приложений ключевые принципы проектирования и шаблоны проектирования (design patterns);
8. Использовать в приложениях assertions;
9. Использовать в приложениях вложенные классы;
10. Использовать в приложениях механизм исключений.

5. МОДУЛЬНЫЙ ПЛАН ПРОГРАММЫ

Программа состоит из 8 курсов, содержащих теорию, видео-разборы, задачи и кодовую базу. По итогам обучения предусмотрено выполнение дипломной работы.

№	Наименование модулей	Объем в академ. часах		
		Всего часов	Теория	Практика
1	2	3	4	5
1.	JVA-I-001 Применение ООП и функциональной парадигмы	38	12	26
2.	JVA-I-002 Разработка бизнес-приложений на фреймворке Spring	30	8	22
3.	JVA-I-003 Работа с базами данных в Java	34	12	22
4.	JVA-I-004 Архитектура REST сервисов	34	12	22
5.	JVA-I-005 Вспомогательные инструменты Java - разработчика	34	12	22
6.	JVA-I-006 Избранные классы стандартной библиотеки	38	12	26
7.	JVA-I-007 Многопоточность в Java	42	16	26
8.	JVA-I-008 Эффективное программирование на Java	24	10	14
9.	Итоговая дипломная работа	10	-	10
ИТОГО		284	94	190

6. УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН МОДУЛЕЙ ПРОГРАММЫ

1. JVA-I-001 Применение ООП и функциональной парадигмы

№	Тема	Количество часов	Форма контроля
1	Углубленный дизайн классов	6	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем
2	Дженерики и коллекции	6	
3	Лямбда-выражения и встроенные функциональные интерфейсы. Stream API	8	
4	Нововведения (до 22-й версии)	6	
5	Избранные шаблоны проектирования	8	
6	Проект и живая практика с преподавателем	4	

	Итого	38	
--	--------------	----	--

2. JVA-I-002 Разработка бизнес-приложений на фреймворке Spring

№	Тема	Количество часов	Форма контроля
1	Основы Spring	10	- тестовые вопросы - задачи для самостоятельной практики - живая практика (лабораторные) с экспертом-преподавателем
2	Разработка Spring Boot приложения	16	
3	Проект и живая практика с преподавателем	4	
	Итого:	30	

3. JVA-I-003 Работа с базами данных в Java

№	Тема	Количество часов	Форма контроля
1.	Основы работы с базами данных	8	- тестовые вопросы - задачи для самостоятельной практики - живая практика (лабораторные) с экспертом-преподавателем
2.	Основы JPA	10	
3.	Spring Data	12	
4.	Проект и живая практика с преподавателем	4	
	Итого	34	

4. JVA-I-004 Архитектура REST сервисов

№	Тема	Количество часов	Форма контроля
1	HTTP, REST, принципы проектирования REST API	4	- тестовые вопросы - задачи для самостоятельной практики - живая практика (лабораторные) с экспертом-преподавателем
2	Основы Spring REST	6	
3	Richardson Maturity Model	8	
4	Swagger/OpenAPI	12	
5	Проект и живая практика с преподавателем	4	
	Итого	34	

5. JVA-I-005 Вспомогательные инструменты Java - разработчика

№	Тема	Количество часов	Форма контроля
1	Сборщики проектов - Maven / Gradle	6	- тестовые вопросы - задачи для самостоятельной практики - живая практика (лабораторные) с экспертом-преподавателем
2	Docker	12	
3	CI/CD	8	
4	Lombok	4	
5	Проект и живая практика с преподавателем	4	
	Итого	34	

6. JVA-I-006 Избранные классы стандартной библиотеки

№	Тема	Количество часов	Форма контроля
---	------	------------------	----------------

1	Продвинутая обработка исключений	8	- тестовые вопросы - задачи для самостоятельной практики - живая практика (лабораторные) с экспертом-преподавателем
2	Проверка инвариантов	6	
3	Основы ввода-вывода	4	
4	Ввод-вывод на базе NIO.2	6	
5	DateTime API	2	
6	Локализация	4	
7	Аннотации и рефлексия	4	
8	Проект и живая практика с преподавателем	4	
	Итого	38	

7. JVA-I-007 Многопоточность в Java

№	Тема	Количество часов	Форма контроля
1	Основы работы с подпроцессами	8	- тестовые вопросы - задачи для самостоятельной практики - живая практика (лабораторные) с экспертом-преподавателем
2	Многопоточные решения в стандартной библиотеке	12	
3	Упрощение синхронизации: Locking Framework	10	
4	Рекурсивная многопоточность: Fork / Join Framework	8	
5	Проект и живая практика с преподавателем	4	
	Итого	42	

8. JVA-I-008 Эффективное программирование на Java

№	Тема	Количество часов	Форма контроля
1	Создание и уничтожение объектов	2	- тестовые вопросы - задачи для самостоятельной практики - живая практика (лабораторные) с экспертом-преподавателем
2	Методы, применяемые ко всем объектам	2	
3	Классы и интерфейсы	2	
4	Обобщенные типы	2	
5	Enums и аннотации	2	
6	Методы	2	
7	Общее программирование	2	
8	Исключения	2	
9	Параллелизм	2	
10	Сериализация	2	
11	Проект и живая практика с преподавателем	4	
	Итого	24	

9. Итоговая дипломная работа

№	Тема	Количество часов	Форма контроля
1	Разработка приложения для анализа нарушений парковки	10	Онлайн защита
	Итого	10	

7. ФОРМА ОБУЧЕНИЯ

Обучение проходит в системе дистанционного обучения в смешанном формате:

- Материалы курсов изучаются в системе дистанционного обучения (СДО). По окончании теоретической части каждого курса программы предусмотрен тест для проверки уровня знаний.
- Практические групповые сессии с преподавателем проходят в режиме вебинара.

8. ФОРМА КОНТРОЛЯ

Контроль усвоения учебной программы проводится в различных формах:

1. Решение практических задач;
2. Промежуточный контроль (тестовый контроль);
3. Тестовый итоговый контроль;
4. Защита дипломной работы.

Курс считается успешно пройденным при достижении следующих результатов:

1. В системе дистанционного обучения прогресс по изучению материалов не менее 80%;
2. Тестовый итоговый контроль сдан не менее чем на 80%;
3. Защита итоговой дипломной работы на оценку не ниже удовлетворительно.

При успешном прохождении итоговой аттестации или получении на итоговой аттестации удовлетворительных результатов, выдается Диплом о профессиональной переподготовке установленного образца.

Лицам, не прошедшим итоговой аттестации или получившим на итоговой аттестации неудовлетворительные результаты выдаётся Справка об обучении по программам дополнительного профессионального образования в соответствии с Положением о порядке выдаче и форме данной справки.

Примеры тестовых вопросов:

1. Какой результат даст такой код? :

```
public class Question_6_1 {
    public static void main(String[] args) {
        Question_6_1 q = new Question_6_1();
        List<Integer> l = new ArrayList<>();
        l.add(20);
        l.add(30);
        q.m1(l);
    }

    private void m1(List<?> l) {
        m2(l);           // 1
    }

    private <T> void m2(List<T> l) {
        l.set(1, l.get(0)); // 2
        System.out.println(l);
    }
}
```

- a. [20, 20]
 - b. Ошибка компиляции на строке 1
 - c. Ошибка компиляции на строке 2
 - d. Выброс исключения
2. Как можно использовать @Value для внедрения значений из файла свойств?
- a. @Value("#{propertyFile['key']}")
 - b. @Value("\${property.name}")
 - c. @Value("#{property.name}")
 - d. @Value("@property.name")
3. Выберите все верные утверждения. Подход Инфраструктура как код (IaC):
- a. Позволяет отслеживать изменения с помощью системы контроля версий git;
 - b. Позволяет развертывать среды с приложением автоматически за счет выполнения инфраструктурного кода;
 - c. Наиболее просто реализуется при использовании режима изоляции приложения от хоста среды выполнения (виртуализация или контейнеризация)
 - d. Обеспечивает только резервное копирование конфигураций со стендов в систему контроля версий
 - e. Упрощает процесс ручного развертывания приложений на серверах
4. Для чего служат реестры Docker (docker registry)? Выберите все верные ответы:
- a. Для хранения образов Docker.
 - b. Для хранения и поиска образов в общедоступном реестре (например, Docker Hub).
 - c. Для загрузки образов с реестра с помощью docker cli.
 - d. Для хранения файловых систем контейнеров
 - e. Для автоматического выполнения тестов на каждом загружаемом образе
5. Что следует сделать, чтобы повысить эффективность и надежность этого кодового сниппета?

```

public enum Operation {
    PLUS, MINUS, TIMES, DIVIDE;
    public double apply(double x, double y) {
        switch(this) {
            case PLUS: return x + y;
            case MINUS: return x - y;
            case TIMES: return x * y;
            case DIVIDE: return x / y;
        }
        throw new AssertionError("Unknown op: " + this);
    }
}

```

- a. Убрать выброс исключения, т.к. оно избыточно, поскольку оператор switch работает со всеми полями инама Operation
- b. Вообще убрать switch, сделать метод apply() абстрактным и в каждом поле переопределить поведение под конкретную операцию
- c. Объявить инам Operation статическим и финальным
- d. В методе apply() надо категорически и обязательно предусмотреть проверку знаменателя на ноль

9. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ПРОГРАММЕ

1. Джошуа Блох. Effective Java. 2018
2. Уоллс Крейг. Spring в действии. 2013
3. Эрих Гамма, Ричард Хелм, Ральф Джонсон, Джон Влиссидес. Паттерны проектирования.201
4. Мартин Фаулер. Рефакторинг: улучшение существующего кода
5. Джошуа Кериевски. Рефакторинг с использованием шаблонов
6. Веб-сайт Documentation Library for Oracle 11gR2 <http://www.oracle.com/pls/db102/homepage>
7. Веб-сайт Oracle Metalink (при наличии учетной записи) <http://metalink.oracle.com>
8. Веб-сайт OWASP www.owasp.org
9. Веб-сайт Spring Blog blog.springsource.com
10. Scott W. Ambler Refactoring Databases : Evolutionary Database Design, 2006
11. Wake, William C. Refactoring Workbook, 2003
12. Feathers, Michael C. Working Effectively with Legacy Code, 2004

Учебная программа разработана: Отделом разработки и методологии АНО ДПО «УЦ ИБС»