

Министерство науки и высшего образования Российской Федерации

ЧИТИНСКИЙ ИНСТИТУТ (ФИЛИАЛ)
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО
УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«БАЙКАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

УТВЕРЖДЕН

на заседании кафедры информационных техно-
логий и высшей математики

24 февраля 2025 г. протокол № 6

Заведующий кафедрой

Л.И. Трухина



**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ
(ФОНД ОЦЕНОЧНЫХ СРЕДСТВ)
ПО УЧЕБНОЙ ДИСЦИПЛИНЕ**

Б1.Э.1 Технологии программирования для мобильных систем

Направление подготовки: 38.03.05 Бизнес-информатика

Направленность (профиль): Цифровая экономика

Квалификация выпускника: бакалавр

Чита, 2025 г.

**Структура
фонда оценочных средств
по дисциплине «Технологии программирования для мобильных систем»**

№ п/п	Этапы формирования компетенций	Перечень формируемых компетенций	ЗУНы (З.1, У1, Н1...)	Контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы	Описание показателей и критериев оценивания компетенций на различных этапах формирования, описания шкал оценивания
1	Введение в программирование под мобильные устройства	ПК-4	З.Знать современные тенденции развития и применения инновационных цифровых компонент ИТ-инфраструктуры предприятия У.Уметь проектировать, разрабатывать и внедрять инновационные цифровые компоненты ИТ-инфраструктуры предприятия Н.Владеть навыками проектирования, разработки и внедрения инновационных цифровых компонент ИТ-инфраструктуры предприятия	Тест по основам проектирования и разработки МП	(процент правильных ответов * 20 баллов), округленных вверх до ближайшего целого числа, если процент правильных ответов меньше 50 – то 0 баллов (20)

2	Программная платформа Xamarin. Основы XAML. Элементы управления. Компоновка элементов и навигация между страницами. Понятие привязки. Жизненный цикл приложения. Манифест.	ПК-4	З.Знать современные тенденции развития и применения инновационных цифровых компонент ИТ-инфраструктуры предприятия У.Уметь проектировать, разрабатывать и внедрять инновационные цифровые компоненты ИТ-инфраструктуры предприятия Н.Владеть навыками проектирования, разработки и внедрения инновационных цифровых компонент ИТ-инфраструктуры предприятия	Лабораторная работа №1. Основы работы с Xamarin. Язык разметки XAML. Элементы компоновки	Полностью выполненная лабораторная работа -20 баллов, частично - доля правильно выполненных заданий*20 баллов, если доля меньше 0,5 - 0 баллов (20)
3	Паттерн MVVM (Model - View - ViewModel). Команды и взаимодействие с пользователем в MVVM	ПК-4	З.Знать современные тенденции развития и применения инновационных цифровых компонент ИТ-инфраструктуры предприятия У.Уметь проектировать, разрабатывать и внедрять инновационные цифровые компоненты ИТ-инфраструктуры предприятия Н.Владеть навыками проектирования, разра-	Лабораторная работа №2. Паттерн Model-View-ViewModel. Привязка (Binding)	Полностью выполненная лабораторная работа - 20 баллов, частично - доля правильно выполненных заданий*20 баллов, если доля меньше 0,5 - 0 баллов (20)

			ботки и внедрения инновационных цифровых компонент ИТ-инфраструктуры предприятия		
4	Элементы управления данными. Linq. Работа с данными	ПК-4	3.Знать современные тенденции развития и применения инновационных цифровых компонент ИТ-инфраструктуры предприятия У.Уметь проектировать, разрабатывать и внедрять инновационные цифровые компоненты ИТ-инфраструктуры предприятия Н.Владеть навыками проектирования, разработки и внедрения инновационных цифровых компонент ИТ-инфраструктуры предприятия	Лабораторная работа №3. Работа с данными. Файлы и база данных. Linq	Полностью выполненная лабораторная работа - 20 баллов, частично - доля правильно выполненных заданий*20 баллов, если доля меньше 0,5 - 0 баллов (20)
5	Итого по текущей аттестации	ПК-4	3.Знать современные тенденции развития и применения инновационных цифровых компонент ИТ-инфраструктуры предприятия У.Уметь проектировать, разрабатывать и внедрять инновацион-		100

			ные цифровые компоненты ИТ-инфраструктуры предприятия Н. Владеть навыками проектирования, разработки и внедрения инновационных цифровых компонент ИТ-инфраструктуры предприятия		
6	Промежуточная аттестация	ПК-4	З. Знать современные тенденции развития и применения инновационных цифровых компонент ИТ-инфраструктуры предприятия У. Уметь проектировать, разрабатывать и внедрять инновационные цифровые компоненты ИТ-инфраструктуры предприятия Н. Владеть навыками проектирования, разработки и внедрения инновационных цифровых компонент ИТ-инфраструктуры предприятия		100

Министерство науки и высшего образования Российской Федерации
ЧИТИНСКИЙ ИНСТИТУТ (ФИЛИАЛ)
ФГБОУ ВО «БАЙКАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Кафедра информационных технологий и высшей математики

Тест по основам проектирования и разработки МП:

1. **Что такое мобильное приложение?**
 - a) Программа, работающая на настольном компьютере
 - b) Программа, разработанная для работы на мобильных устройствах
 - c) Веб-сайт, оптимизированный для мобильных устройств
 - d) Игра для консоли
2. **Какой язык программирования используется для разработки приложений под Android?**
 - a) Swift
 - b) Java
 - c) C#
 - d) HTML
3. **Что такое UI (User Interface)?**
 - a) Логика приложения
 - b) Визуальные элементы, с которыми взаимодействует пользователь
 - c) База данных приложения
 - d) Серверная часть приложения
4. **Какой из следующих паттернов архитектуры чаще всего используется в мобильной разработке?**
 - a) MVC (Model-View-Controller)
 - b) MVP (Model-View-Presenter)
 - c) MVVM (Model-View-ViewModel)
 - d) Все вышеперечисленные
5. **Что такое API (Application Programming Interface)?**
 - a) Интерфейс пользователя
 - b) Набор правил и протоколов для взаимодействия между программами
 - c) База данных приложения
 - d) Язык программирования
6. **Какой из следующих фреймворков используется для кроссплатформенной разработки мобильных приложений?**
 - a) React Native
 - b) Django
 - c) Ruby on Rails
 - d) Angular

7. **Что такое "привязка данных" в контексте разработки приложений?**
- a) Процесс подключения к базе данных
 - b) Связывание пользовательского интерфейса с данными модели
 - c) Сохранение данных на устройстве пользователя
 - d) Передача данных между сервером и клиентом
8. **Какой из следующих элементов является частью пользовательского интерфейса в мобильном приложении?**
- a) Кнопка (Button)
 - b) Модель данных (Data Model)
 - c) Серверная логика (Server Logic)
 - d) База данных (Database)
9. **Что такое "отзывчивый дизайн" (Responsive Design)?**
- a) Дизайн, который адаптируется к различным размерам экранов и устройствам
 - b) Дизайн, который требует постоянного подключения к интернету
 - c) Дизайн, который не изменяется в зависимости от устройства
 - d) Дизайн, который использует только текстовые элементы
10. **Какой из следующих инструментов используется для тестирования мобильных приложений?**
- a) Selenium
 - b) JUnit
 - c) Appium
 - d) Git
11. **Что такое "пользовательский опыт" (User Experience, UX)?**
- a) Процесс создания графического интерфейса пользователя
 - b) Общее впечатление пользователя от взаимодействия с приложением
 - c) Код, написанный для приложения
 - d) Набор функций приложения
12. **Какой из следующих языков используется для разработки под iOS?**
- a) Kotlin
 - b) Swift
 - c) JavaScript
 - d) C++
13. **Что такое "гибкая верстка" (Flexible Layout)?**
- a) Верстка, которая фиксирована по размеру экрана устройства
 - b) Верстка, которая автоматически адаптируется к размеру экрана
 - c) Верстка, которая требует ручного изменения при каждом обнов-

лении

d) Верстка без использования CSS

14. Какой из следующих методов используется для хранения данных на устройстве?

a) REST API

b) SQLite

c) JSON

d) XML

15. Что такое "кэширование" в контексте мобильных приложений?

a) Процесс удаления ненужных файлов

b) Хранение временных данных для ускорения доступа к ним

c) Процесс шифрования данных

d) Хранение всех данных на сервере

16. Какой из следующих подходов позволяет улучшить производительность мобильного приложения?

a) Увеличение размера изображений

b) Оптимизация запросов к базе данных

c) Использование большого количества анимаций

d) Игнорирование отзывов пользователей

17. Что такое "интерфейс программирования приложений" (API)?

a) Набор инструментов для создания графического интерфейса

b) Набор функций и протоколов для взаимодействия между различными программами

c) Способ хранения данных на устройстве

d) Метод тестирования производительности приложения

18. Какой из следующих принципов является основным при проектировании пользовательского интерфейса?

a) Сложность интерфейса делает его более привлекательным

b) Пользователь должен легко понимать и использовать интерфейс

c) Интерфейс должен быть как можно более ярким и насыщенным цветами

d) Интерфейс не должен содержать текстовых элементов

19. Что такое "прототипирование" в разработке приложений?

a) Процесс создания окончательной версии приложения

b) Создание предварительной модели или макета приложения для тестирования идей

c) Процесс написания документации по проекту

d) Метод тестирования производительности приложения

20. Какое значение имеет отзывчивый дизайн в разработке мобиль-

ных приложений?

- a) Он не имеет значения; все пользователи используют одинаковые устройства
- b) Он позволяет улучшить взаимодействие пользователей с приложением на различных устройствах и экранах
- c) Он усложняет процесс разработки приложения
- d) Он требует больше времени на реализацию

Ответы:

- 1. b
- 2. b
- 3. b
- 4. d
- 5. b
- 6. a
- 7. b
- 8. a
- 9. a
- 10. c
- 11. b
- 12. b
- 13. b
- 14. b
- 15. b
- 16. b
- 17. b
- 18. b
- 19. b
- 20. b

Лабораторная работа №1. Основы работы с Xamarin и XAML

Цель работы:

Научиться создавать простое мобильное приложение с использованием Xamarin.Forms и языка разметки XAML. Ознакомиться с основными элементами управления, их свойствами и событиями.

Задание:

1. Создание проекта:

- 1. Создайте новый проект в Visual Studio с использованием шаблона "Mobile App (Xamarin.Forms)".
- 2. Выберите шаблон "Blank" для создания пустого приложения.

2. Структура приложения:

- 1. В проекте создайте три страницы:

1. MainPage.xaml — главная страница приложения.
2. SecondPage.xaml — вторая страница, на которую будет переходить пользователь.
3. ThirdPage.xaml — третья страница, которая будет отображать информацию о приложении.

3. Разметка главной страницы (MainPage.xaml):

1. Используйте следующие элементы управления:
 1. Label для отображения заголовка приложения.
 2. Button для перехода на SecondPage.
 3. Button для перехода на ThirdPage.
2. Пример разметки:

```
xml
Копировать код

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms" xmlns:x="http://schema
```

4. Код за главной страницей (MainPage.xaml.cs):

1. Реализуйте обработчики событий для кнопок, чтобы осуществлять навигацию между страницами.

```
csharp
Копировать код

private async void OnSecondPageButtonClicked(object sender, EventArgs e)
{
    await Navigation.PushAsync(new SecondPage());
}

private async void OnThirdPageButtonClicked(object sender, EventArgs e)
{
    await Navigation.PushAsync(new ThirdPage());
}
```

5. Разметка второй страницы (SecondPage.xaml):

1. Добавьте элемент Label, который будет отображать текст "Это вторая страница".
2. Добавьте кнопку "Назад", которая будет возвращать пользователя на главную страницу.

```
xml
Копировать код

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    x:Class="YourNamespace.SecondPage">
    <StackLayout Padding="20">
        <Label Text="Это вторая страница"
            FontSize="24"
            HorizontalOptions="Center" />
        <Button Text="Назад"
            Clicked="OnBackButtonClicked" />
    </StackLayout>
</ContentPage>
```

Код за второй страницей (SecondPage.xaml.cs):

```
csharp

private async void OnBackButtonClicked(object sender, EventArgs e)
{
    await Navigation.PopAsync();
}
```

6. Разметка третьей страницы (ThirdPage.xaml):

1. Добавьте элемент Label, который будет отображать информацию о приложении (например, название и версию).
2. Добавьте кнопку "Назад", которая будет возвращать пользователя на главную страницу.

7. Код за третьей страницей (ThirdPage.xaml.cs):

```
csharp

private async void OnBackButtonClicked(object sender, EventArgs e)
{
    await Navigation.PopAsync();
}
```

Дополнительные задания:

1. Измените стиль элементов управления (цвета, шрифты) с помощью свойств XAML.
2. Добавьте обработку событий для изменения текста метки при нажатии кнопки.
3. Реализуйте возможность возврата на предыдущую страницу с помощью жестов (например, свайп).

Оценивание (20 баллов):

1. Полнота выполнения задания: 50%
2. Качество кода и использование XAML: 30%
3. Дополнительные функции и креативность: 20%

Заключение

После выполнения лабораторной работы студенты должны быть знакомы с основами разработки приложений на Xamarin.Forms, а также уметь использовать язык разметки XAML для создания пользовательского интерфейса.

Лабораторная работа №2. Паттерн Model-View-ViewModel (MVVM) и Привязка данных

Цель работы:

Научиться применять паттерн MVVM в разработке мобильных приложений с использованием Xamarin.Forms. Ознакомиться с концепцией привязки данных и реализацией интерфейса пользователя через ViewModel.

Задание:

1. Создание проекта:

1. Создайте новый проект в Visual Studio с использованием шаблона "Mobile App (Xamarin.Forms)".
2. Выберите шаблон "Blank" для создания пустого приложения.

2. Структура проекта:

1. Создайте следующие папки в проекте:
 1. Models — для хранения моделей данных.
 2. ViewModels — для хранения классов ViewModel.
 3. Views — для хранения страниц приложения.

3. Создание модели (Model):

1. В папке Models создайте класс Person, который будет представлять человека с двумя свойствами: Name и Age.

csharpКопировать код

```
public class Person
{
    public string Name { get; set; }
    public int Age { get; set; }
}
```

4. Создание ViewModel:

1. В папке ViewModels создайте класс PersonViewModel, который будет реализовывать интерфейс INotifyPropertyChanged.
2. Добавьте свойства для модели Person и реализуйте логику изменения этих свойств.

csharpКопировать код

```
using System.ComponentModel;

public class PersonViewModel : INotifyPropertyChanged
{
    private Person _person;

    public PersonViewModel()
    {

```

```
    _person = new Person { Name = "Иван", Age = 25 };  
}
```

```
public string Name  
{  
    get => _person.Name;  
    set  
    {  
        if (_person.Name != value)  
        {  
            _person.Name = value;  
            OnPropertyChanged(nameof(Name));  
        }  
    }  
}
```

```
public int Age  
{  
    get => _person.Age;  
    set  
    {  
        if (_person.Age != value)  
        {  
            _person.Age = value;  
            OnPropertyChanged(nameof(Age));  
        }  
    }  
}
```

```
public event PropertyChangedEventHandler PropertyChanged;
```

```
protected virtual void OnPropertyChanged(string propertyName)  
{  
    PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(prop-  
ertyName));  
}
```

5. Создание представления (View):

1. В папке Views создайте страницу MainPage.xaml.
2. Используйте привязку данных для отображения свойств из ViewModel.

xmlКопировать код

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
```

```

        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
        x:Class="YourNamespace.Views.MainPage">
<StackLayout Padding="20">
    <Label Text="Имя:" />
    <Entry Text="{Binding Name}" />

    <Label Text="Возраст:" />
    <Entry Text="{Binding Age}" Keyboard="Numeric" />

    <Label Text="Вы ввели:" FontSize="20" />
    <Label Text="{Binding Name}" FontSize="24" />
    <Label Text="{Binding Age}" FontSize="24" />
</StackLayout>
</ContentPage>

```

6. Привязка ViewModel к представлению:

1. В коде за страницей (MainPage.xaml.cs) создайте экземпляр PersonViewModel и установите его как контекст привязки.

csharpКопировать код

```

using Xamarin.Forms;

namespace YourNamespace.Views
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
            BindingContext = new PersonViewModel();
        }
    }
}

```

7. Тестирование приложения:

1. Запустите приложение на эмуляторе или реальном устройстве.
2. Изменяйте значения в полях ввода, чтобы убедиться, что они корректно обновляют соответствующие метки.

Дополнительные задания:

1. Добавьте кнопку "Сбросить", которая будет сбрасывать значения имени и возраста на значения по умолчанию.
2. Реализуйте валидацию ввода, чтобы возраст не мог быть отрицательным числом.

3. Создайте дополнительную страницу, которая будет отображать список людей, используя коллекцию в ViewModel.

Оценивание (20 баллов):

1. Полнота выполнения задания: 50%
2. Качество кода и использование MVVM: 30%
3. Дополнительные функции и креативность: 20%

Заключение

После выполнения лабораторной работы студенты должны быть знакомы с основами паттерна MVVM, а также уметь использовать привязку данных для связывания пользовательского интерфейса с логикой приложения.

Лабораторная работа №3. Работа с данными. Файлы и база данных. LINQ

Цели работы:

1. Научиться работать с файлами (чтение и запись).
2. Ознакомиться с основами работы с базами данных.
3. Использовать LINQ для выполнения запросов к данным.

Задание:

1. Создание проекта:

1. Создайте новый проект мобильного приложения (например, на платформе Xamarin или .NET MAUI).

2. Работа с файлами:

1. Создайте класс Person с полями: Id, Name, Age.
2. Реализуйте методы для записи и чтения списка объектов Person в/из текстового файла (например, в формате CSV).
 1. Метод записи должен принимать список объектов Person и сохранять их в файл.
 2. Метод чтения должен загружать данные из файла и возвращать список объектов Person.

3. Работа с базой данных:

1. Настройте SQLite в вашем проекте.
2. Создайте таблицу для хранения объектов Person.
3. Реализуйте методы для добавления, удаления и получения списка объектов Person из базы данных.

4. Использование LINQ:

1. Напишите запросы LINQ для выполнения следующих операций:
 1. Получение всех людей старше 18 лет.
 2. Получение человека с максимальным возрастом.
 3. Подсчет количества людей в базе данных.

5. Интерфейс пользователя:

1. Создайте простой интерфейс пользователя, который позволит:

1. Добавлять новых людей (через ввод имени и возраста).
2. Отображать список всех людей из базы данных.
3. Сохранять данные в файл и загружать их обратно.

6. Тестирование:

1. Протестируйте все функции вашего приложения, убедитесь, что данные корректно сохраняются и загружаются как из файла, так и из базы данных.

7. Отчет:

1. Подготовьте отчет о выполненной работе, включающий:
 1. Описание структуры проекта.
 2. Код реализованных методов.
 3. Скриншоты интерфейса приложения.
 4. Результаты тестирования.

Дополнительные рекомендации:

1. Используйте обработку исключений для управления ошибками при работе с файлами и базой данных.
2. Обратите внимание на форматирование кода и комментарии для улучшения читаемости.

Оценивание (20 баллов):

1. Полнота выполнения задания (30%)
2. Качество кода (30%)
3. Работоспособность приложения (20%)
4. Отчет о выполненной работе (20%)

Это задание поможет студентам получить практический опыт работы с данными в мобильных приложениях, а также освоить основные концепции работы с файлами, базами данных и LINQ. Если вам нужно внести изменения или добавить дополнительные элементы, дайте знать!

Лабораторная работа №4. Создание кросс-платформенного приложения

Цели работы:

1. Ознакомиться с основами разработки кросс-платформенных приложений.
2. Научиться использовать фреймворки для создания мобильных приложений (например, Xamarin, .NET MAUI или Flutter).
3. Реализовать простое приложение с использованием пользовательского интерфейса и взаимодействия с данными.

Задание:

1. **Создание проекта:**

1. Создайте новый проект кросс-платформенного приложения с использованием выбранного вами фреймворка (Xamarin, .NET MAUI или Flutter).
2. **Определение функциональности:**
 1. Определите функциональность вашего приложения. Например, это может быть приложение для заметок, список задач или простая игра.
 2. Напишите краткое описание приложения и его основных функций.
3. **Разработка пользовательского интерфейса:**
 1. Создайте основной экран приложения с элементами управления (кнопки, текстовые поля, списки и т.д.).
 2. Используйте адаптивный дизайн для обеспечения корректного отображения на разных устройствах (телефонах и планшетах).
4. **Реализация логики приложения:**
 1. Реализуйте основные функции вашего приложения:
 1. Добавление новых элементов (например, заметок или задач).
 2. Отображение списка элементов.
 3. Удаление элементов из списка.
 2. Используйте подход MVVM (Model-View-ViewModel) для организации кода (если применимо).
5. **Работа с данными:**
 1. Реализуйте сохранение данных локально (например, используя SQLite или Shared Preferences).
 2. Обеспечьте возможность загрузки и сохранения данных при запуске и закрытии приложения.
6. **Тестирование:**
 1. Протестируйте приложение на различных устройствах и эмуляторах.
 2. Убедитесь, что все функции работают корректно и интерфейс адаптируется под разные размеры экранов.
7. **Документация:**
 1. Подготовьте документацию по вашему приложению, включающую:
 1. Описание функциональности.
 2. Инструкции по установке и запуску.
 3. Скриншоты интерфейса.
8. **Презентация:**
 1. Подготовьте короткую презентацию о вашем приложении, в которой расскажете о его функциональности, процессе разработки и возможных улучшениях.

Оценивание (20 баллов):

1. Полнота выполнения задания (30%)
2. Качество пользовательского интерфейса (30%)
3. Работоспособность приложения (20%)
4. Документация и презентация (20%)

Дополнительные рекомендации:

1. Используйте сторонние библиотеки для улучшения функциональности вашего приложения (например, для работы с сетью или анимацией).
2. Обратите внимание на производительность и отзывчивость интерфейса.

Это задание поможет студентам получить практический опыт в разработке кросс-платформенных приложений и освоить основные концепции проектирования пользовательского интерфейса и работы с данными. Если вам нужно внести изменения или добавить дополнительные элементы, дайте знать!

Список вопросов к зачету

1. SQLite. Применение в UWP. Возможности. Entity Framework
2. Адаптивный дизайн и код. Представления XAML
3. Архитектура операционной системы Android
4. Архитектура приложений UWP
5. Архитектура приложений Xamarin
6. Виды приложений Android и их структура
7. Двухмерная графика. Базовые графические примитивы.
8. Диаграммы UML.
9. Установка разработанного программного обеспечения (манифест, магазины)
10. Основные понятия и структура проекта информационной системы (ИС)
11. Основные принципы дизайна интерфейса мобильных/универсальных приложений
12. Основные типы разработки мобильных приложений.
13. Основные элементы XAML и их атрибуты
14. Основные языки и парадигмы программирования для разработки мобильных приложений
15. Понятие привязки, способы привязки данных
16. Работа с аудио и видео. Фоновые задачи.
17. Работа с графикой и мультимедиа в UWP. Основные приемы и используемые классы
18. Работа с файловой системой на примере UWP или Android. Основные концепции, классы и объекты, принципы
19. Разграничение прав пользователей ИС
20. Стили и шаблоны UWP. Способы создания, примеры использования
21. Тестирование и отладка ИС
22. Элементы управления данными. ObservableCollection
23. Элементы управления данными. Их поведение и предназначение, примеры использования
24. Этапы жизненного цикла программного обеспечения. Модели жизненного цикла

Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Система критериев оценки определяет оценку успеваемости по каждому заданию (вопросу) экзаменационного билета или заданию для зачета с использованием интервальной шкалы баллов, применяемой в привязке к рейтинговой 100-балльной системе.

ОЦЕНКА ОТВЕТА НА ТЕОРЕТИЧЕСКИЙ ВОПРОС В УСТНОЙ ИЛИ ПИСЬМЕННОЙ ФОРМЕ:

Оценка «отлично» / «зачтено» (91-100 баллов) выставляется при соблюдении следующих условий: Ответ отличается глубиной и полнотой, свободным владением понятийно-категориальным (терминологическим) аппаратом изученной дисциплины. Отражает знание не только основной, но и дополнительной литературы. Приведены примеры, отражающие умение связать теорию с практикой. Ответ изложен логически последовательно, грамотно и корректно.

Оценка «хорошо» / «зачтено» (76-90 баллов) выставляется при соблюдении следующих условий: Ответ отличается полнотой, владением понятийно-категориальным (терминологическим) аппаратом изученной дисциплины, но в ответе могут присутствовать неточности. Отражает знание основной литературы. Приведены примеры, отражающие умение связать теорию с практикой. Ответ изложен логически последовательно, грамотно и корректно, но недостаточно аргументирован.

Оценка «удовлетворительно» / «зачтено» (61-75 баллов) выставляется при соблюдении следующих условий: В ответе отражено знание понятийно-категориального (терминологического) аппарата изучаемой дисциплины, но присутствуют отдельные ошибки и неточности. Ответ характеризуется недостаточным знанием рекомендованной литературы. Примеры, отражающие умение связать теорию с практикой, тривиальны, либо отсутствуют. Ответ неполный, носит фрагментарный, непоследовательный характер.

Оценка «неудовлетворительно» / «не зачтено» (0-60 баллов) выставляется при соблюдении следующих условий: Ответ характеризуется незнанием, либо фрагментарным представлением о понятийно-категориальном аппарате дисциплины, содержит множество ошибок. Примеры и иллюстрации отсутствуют. Ответ логически непоследователен.

ОЦЕНКА ВЫПОЛНЕНИЯ ЗАДАНИЯ В ФОРМЕ CASE-STUDY (СИТУАЦИИ)

Оценка «отлично» / «зачтено» (91-100 баллов) выставляется при соблюдении следующих условий: Четкая формулировка проблемы. Полное и соответствующее ситуации решение, основанное на знании правовых норм и технологий (опыте), применяемых в реальных организациях (известных компаниях). Предполагаемые действия описаны логично и последовательно. Даны дополнительные авторские комментарии и предложения к решению ситуации.

Оценка «хорошо» / «зачтено» (76-90 баллов) выставляется при соблюдении следующих условий: Понимание сути проблемы, но ее формулирование затруднено. Решение соответствует ситуации, отражает знание правовых норм и опыт работы других организаций при решении подобных ситуаций. Логика и последовательность действий не нарушены.

Оценка «удовлетворительно» / «зачтено» (61-75 баллов) выставляется при соблюдении следующих условий: Проблема не сформулирована. Приведен набор действий, потенциально способствующих улучшению ситуации и решению проблемы.

Оценка «неудовлетворительно» / «не зачтено» (0-60 баллов) выставляется при соблюдении следующих условий: Предложенный перечень мероприятий не соответствует

ситуации.

ОЦЕНКА РЕШЕНИЯ ЗАДАЧИ

Оценка «отлично» / «зачтено» (91-100 баллов) выставляется при соблюдении следующих условий: Полное верное решение - оценивается в n баллов (n – максимальное количество баллов за решение задачи в структуре экзаменационного билета/задания).

Оценка «хорошо» / «зачтено» (76-90 баллов) выставляется при соблюдении следующих условий: Верное решение; имеются небольшие недочеты, в целом не влияющие на решение – оценивается в диапазоне от $0,76 \cdot n$ баллов до $0,9 \cdot n$ баллов (n – максимальное количество баллов за решение задачи в структуре экзаменационного билета/задания).

Оценка «удовлетворительно» / «зачтено» (61-75 баллов) выставляется при соблюдении следующих условий: Решение в целом верное; однако оно содержит ряд ошибок, либо не учитывает отдельных случаев, но может стать правильным после некоторых исправлений или дополнений – оценивается в диапазоне от $0,61 \cdot n$ баллов до $0,75 \cdot n$ баллов (n – максимальное количество баллов за решение задачи в структуре экзаменационного билета/задания).

Оценка «неудовлетворительно» / «не зачтено» (0-60 баллов) выставляется при соблюдении следующих условий: Решение неверное; изначально выбран неверный ход решения, или решение отсутствует – оценивается в 0 баллов.

ОЦЕНКА ВЫПОЛНЕНИЯ ТЕСТОВОГО ЗАДАНИЯ

Подсчитывается доля набранных баллов в максимальной сумме баллов за все задания теста:

– Каждый правильный ответ на тестовый вопрос (тип выборочный, одинарный, множественный, открытый) оценивается в m баллов (число m определяется путем деления максимального количества баллов за выполнение теста в структуре экзаменационного билета/задания на количество тестовых заданий);

– Каждый частично правильный ответ на тестовый вопрос (тип выборочный, множественный, открытый) оценивается в $m/2$ баллов независимо от соотношения правильно/неправильно выбранных вариантов (число m определяется путем деления максимального количества баллов за выполнение теста в структуре экзаменационного билета/задания на количество тестовых заданий);

– Каждый неправильный ответ на тестовый вопрос (тип выборочный, одинарный) оценивается в 0 баллов.

Оценка «отлично»/ «зачтено» (91-100 баллов) выставляется, если доля набранных баллов составляет 91-100%.

Оценка «хорошо»/ «зачтено» (76-90 баллов), если доля набранных баллов составляет 76-90%.

Оценка «удовлетворительно»/ «зачтено» (61-75 баллов), если доля набранных баллов составляет 61-75%.

Оценка «неудовлетворительно»/ «не зачтено» (0-60 баллов), если доля набранных баллов составляет не более 60%.