

Скоков
Роман
Юрьевич
Ректор "
ВИЭПП"

2026.02.27
15:46:55
+03'00'

Приложение 2 к рабочей программе дисциплины
«Визуальное программирование на языке Python»

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОЛЖСКИЙ ИНСТИТУТ ЭКОНОМИКИ, ПЕДАГОГИКИ И ПРАВА»

Факультет информационной безопасности и права

Фонд оценочных средств
по дисциплине
«Визуальное программирование на языке Python»

Направление подготовки
09.03.03 Прикладная информатика

Направленность (профиль) образовательной программы:
Общий правовой профиль

Уровень высшего образования:
бакалавриат

Квалификация выпускника:
«бакалавр»

Волжский 2026 г.

Содержание

1	Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы, описание показателей, критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания.....	3
2	Методические материалы и типовые контрольные задания, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы.....	4
2.1	Типовые контрольные задания, используемые для проведения входного контроля	4
2.2	Типовые контрольные задания, используемые для промежуточной аттестации по дисциплине.....	5
2.3	Методические материалы и типовые контрольные задания, используемые для текущего контроля по дисциплине	6
2.3.1	Методические материалы, используемые для текущего контроля знаний по дисциплине.....	6
2.3.2	Вопросы, выносимые на самостоятельное изучение	7
2.3.3	Задания для самостоятельной работы	8
2.3.4	Тесты по дисциплине	8
2.3.5	Типовые задания.....	9
2.3.6	Задания для контрольной работы	10
2.3.7	Методика проведения лекции-дискуссии	10
2.3.8	Методика проведения интерактивного решения задач	11
3	Методические материалы, определяющие процедуры оценивания знаний, умений, навыков по дисциплине	12
3.1	Балльно-рейтинговая система оценки успеваемости по дисциплине	12

1 Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы, описание показателей, критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания

Перечень компетенций	Показатели оценивания компетенций	Критерии оценивания компетенций	Этапы формирования компетенций	Шкала оценивания
ПК-2 Способен осуществлять руководство проверкой работоспособности компьютерного программного обеспечения	– знает архитектуру событийно-ориентированных приложений и принципы проектирования GUI; – умеет разрабатывать приложения с использованием библиотек Tkinter, PyQt6, Kivy; – владеет методами тестирования и отладки визуальных интерфейсов.	– имеет базовые знания архитектуры событийно-ориентированных приложений; – умеет на основе типовых примеров создавать простые GUI-приложения; – владеет навыками базовой отладки интерфейсов с использованием стандартных инструментов.	Начальный	удовлетворительно (60–74 балла)
	– знает архитектуру событийно-ориентированных приложений и принципы проектирования GUI; – умеет разрабатывать приложения с использованием библиотек Tkinter, PyQt6, Kivy; – владеет методами тестирования и отладки визуальных интерфейсов.	– знает особенности архитектуры различных библиотек визуального программирования; – умеет самостоятельно проектировать и реализовывать многооконные приложения с использованием паттернов проектирования; – владеет навыками модульного тестирования интерфейсов и профилирования производительности.	Основной	хорошо (75–89 баллов)
	– знает архитектуру событийно-ориентированных приложений и принципы проектирования GUI; – умеет разрабатывать приложения с использованием библиотек Tkinter, PyQt6, Kivy; – владеет методами тестирования и отладки визуальных интерфейсов.	– знает современные подходы к проектированию пользовательских интерфейсов (UX/UI); – умеет руководить процессом разработки сложных кроссплатформенных приложений; – владеет опытом организации комплексного тестирования работоспособности ПО, включая нагрузочное и регрессионное тестирование.	Завершающий	отлично (90–100 баллов)

2 Методические материалы и типовые контрольные задания, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

2.1 Типовые контрольные задания, используемые для проведения входного контроля

1. Что такое событийно-ориентированное программирование? а) Программирование, основанное на последовательном выполнении инструкций б) Программирование, при котором выполнение кода инициируется внешними событиями в) Программирование с использованием рекурсивных функций г) Программирование с параллельным выполнением потоков
2. Какая из перечисленных библиотек НЕ является библиотекой для создания графического интерфейса на Python? а) Tkinter б) PyQt в) NumPy г) Kivy
3. Что такое паттерн проектирования MVC? а) Метод оптимизации алгоритмов б) Архитектурный шаблон, разделяющий приложение на модель, представление и контроллер в) Способ организации базы данных г) Технология шифрования данных
4. Какой метод используется для запуска главного цикла обработки событий в Tkinter? а) main() б) start() в) mainloop() г) run()
5. Что такое сигналы и слоты в контексте PyQt? а) Механизм обмена данными между потоками б) Способ организации сетевого взаимодействия в) Механизм связи между объектами для обработки событий г) Способ хранения конфигурационных данных
6. Какой менеджер компоновки в Tkinter позволяет размещать виджеты в таблицу? а) pack() б) place() в) grid() г) table()
7. Что такое кроссплатформенная разработка? а) Разработка приложений для одной операционной системы б) Разработка приложений, работающих на разных операционных системах без модификации в) Разработка веб-приложений г) Разработка мобильных приложений только для Android
8. Какой инструмент используется для упаковки Python-приложения в исполняемый файл? а) pip б) PyInstaller в) virtualenv г) pytest
9. Какой принцип проектирования интерфейсов предполагает минимизацию количества действий для достижения цели? а) Последовательность б) Эффективность в) Узнаваемость г) Эстетика
10. Что такое юнит-тестирование в контексте GUI-приложений? а) Тестирование производительности приложения б) Тестирование отдельных компонентов интерфейса на корректность работы в) Тестирование безопасности приложения г) Тестирование совместимости с разными ОС
11. Какой виджет в Tkinter используется для многострочного ввода текста? а) Entry б) Label в) Text г) Button
12. Что такое сигнал в PyQt? а) Специальный тип данных б) Механизм уведомления о наступлении события в) Способ передачи данных между потоками г) Тип исключения
13. Какой подход используется для предотвращения блокировки интерфейса при выполнении длительных операций? а) Рекурсия б) Многопоточность или асинхронное программирование в) Увеличение размера стека г) Оптимизация алгоритмов
14. Что такое стилизация интерфейса через QSS? а) Создание 3D-эффектов б) Применение каскадных таблиц стилей для оформления виджетов Qt в) Анимация переходов между окнами г) Настройка цветовой палитры операционной системы
15. Какой элемент в Kivy используется для описания интерфейса в декларативном стиле? а) Python-код б) KV-язык в) XML г) JSON
16. Что такое «жизненный цикл» GUI-приложения? а) Время работы приложения от запуска до закрытия б) Последовательность этапов: инициализация,

обработка событий, завершение в) Период поддержки приложения разработчиком г) Время выполнения главного цикла событий

17. Какой метод в PyQt используется для подключения сигнала к слоту? а) connect() б) bind() в) link() г) attach()

18. Что такое «привязка событий» в контексте Tkinter? а) Связывание виджета с переменной б) Назначение обработчика на определенное событие (клик, нажатие клавиши) в) Создание связи между двумя виджетами г) Привязка приложения к операционной системе

19. Какой принцип проектирования интерфейсов требует, чтобы система всегда информировала пользователя о происходящем? а) Свобода действий б) Последовательность в) Обратная связь г) Минимализм

20. Что такое «кастомный виджет»? а) Стандартный виджет библиотеки б) Виджет, созданный путем наследования и расширения функционала базового виджета в) Виджет с измененным цветом г) Временный виджет для отладки

2.2. Типовые контрольные задания, используемые для промежуточной аттестации по дисциплине

2.2.1. Примерный перечень вопросов к зачету

Теоретические вопросы

1. Концепция графического пользовательского интерфейса (GUI). Эволюция подходов к созданию интерфейсов.
2. Архитектура событийно-ориентированных приложений. Цикл обработки событий (event loop).
3. Принципы проектирования пользовательских интерфейсов (UX/UI). Законы Нильсена.
4. Библиотека Tkinter: архитектура, основные компоненты, преимущества и ограничения.
5. Менеджеры компоновки в Tkinter (pack, grid, place). Сравнительный анализ.
6. Обработка событий в Tkinter: привязка событий, обработчики, передача параметров.
7. Многопоточность в GUI-приложениях. Проблема блокировки интерфейса. Паттерн «Очередь событий».
8. Архитектура Qt и PyQt6. Signals & Slots как основа взаимодействия компонентов.
9. Инструмент Qt Designer. Создание интерфейсов в визуальном редакторе и интеграция с кодом Python.
10. Сложные виджеты PyQt6: QTableView, QTreeView, QTabWidget. Модель-представление.
11. Стилизация интерфейсов через Qt Style Sheets (QSS). Каскадные таблицы стилей для виджетов.
12. Создание кастомных виджетов в PyQt6 путем наследования от базовых классов.
13. Особенности кроссплатформенной разработки. Библиотека Kivy: архитектура и преимущества.
14. KV-язык в Kivy. Декларативное описание интерфейсов.
15. Мультиязычные интерфейсы и анимации в Kivy.
16. Деплоймент Python-приложений: упаковка в исполняемые файлы (PyInstaller), создание установщиков.
17. Тестирование GUI-приложений: подходы, инструменты (unittest, pytest-qt), стратегии тестирования.

18. Профилирование и оптимизация производительности визуальных приложений.
19. Интеграция GUI-приложений с базами данных (SQLite, PostgreSQL).
20. Паттерны проектирования в GUI-разработке: MVC, Observer, Singleton. Применение на практике.

Практические вопросы

1. Разработайте архитектуру приложения «Юридический калькулятор» с использованием паттерна MVC.
2. Создайте многооконное приложение на Tkinter с передачей данных между окнами.
3. Реализуйте обработку длительной операции (например, обработка большого файла) без блокировки интерфейса с использованием очереди событий.
4. Спроектируйте и реализуйте интерфейс управления базой данных клиентов с использованием PyQt6 и модели представления.
5. Создайте кастомный виджет для визуализации статистических данных (график, диаграмма).
6. Реализуйте систему авторизации пользователя с хранением учетных данных в зашифрованном виде.
7. Разработайте приложение с поддержкой тем оформления (светлая/темная) с возможностью переключения в реальном времени.
8. Создайте приложение для мобильной платформы с использованием Kivy с поддержкой мультитач-жестов.
9. Реализуйте модульное тестирование для виджета ввода данных с валидацией.
10. Спроектируйте систему логирования действий пользователя в GUI-приложении с возможностью экспорта отчетов.
11. Разработайте приложение с поддержкой локализации (нескольких языков интерфейса).
12. Создайте приложение с интеграцией веб-API (например, получение данных о курсах валют).
13. Реализуйте механизм отмены/повтора действий (undo/redo) в текстовом редакторе.
14. Спроектируйте систему горячих клавиш для управления функционалом приложения.
15. Разработайте приложение с поддержкой работы в офлайн-режиме с синхронизацией данных при подключении к сети.
16. Создайте систему автоматического сохранения данных при закрытии приложения.
17. Реализуйте валидацию ввода данных в реальном времени с визуальной обратной связью.
18. Разработайте приложение с поддержкой нескольких документов (MDI – Multiple Document Interface).
19. Создайте систему плагинов для расширения функционала приложения без изменения основного кода.
20. Спроектируйте и реализуйте комплексную систему тестирования работоспособности GUI-приложения.

2.2.2 Примерное задание на зачет

Задание:

Разработайте полнофункциональное GUI-приложение «Менеджер юридических документов» с использованием библиотеки PyQt6, включающее:

1. Модуль авторизации пользователей с хранением учетных данных в зашифрованном виде;
2. Основное окно с древовидной структурой документов и таблицей просмотра метаданных;
3. Возможность добавления, редактирования, удаления и поиска документов;
4. Интеграцию с локальной базой данных SQLite для хранения метаданных;
5. Систему валидации ввода данных с визуальной обратной связью;
6. Поддержку тем оформления (светлая/темная);
7. Модуль экспорта данных в форматы PDF и DOCX;
8. Систему логирования действий пользователя;
9. Модульное тестирование ключевых компонентов приложения.

Предоставьте: – Исходный код приложения с комментариями; – Архитектурную схему приложения с обоснованием выбора паттернов проектирования; – Отчет о тестировании с результатами прохождения тестов; – Инструкцию по установке и использованию приложения.

2.3 Методические материалы и типовые контрольные задания, используемые для текущего контроля по дисциплине

2.3.1 Методические материалы, используемые для текущего контроля знаний по дисциплине

Наименование оценочного средства	Краткая характеристика оценочного средства	Представление оценочного средства в фонде	Наименование оценочного средства
Контрольный опрос	Метод оценки уровня освоения компетенций на основе непосредственного взаимодействия преподавателя и студента	Примерный перечень вопросов к зачету, вопросы для самостоятельного изучения	Контрольный опрос
Собеседование	Относительно свободный диалог между преподавателем и студентом на заданную тему	Вопросы к практическим занятиям, задания для самостоятельной работы	Собеседование
Тестовые задания	Эмпирико-аналитическая процедура оценки уровня освоения компетенций	Тесты по дисциплине (раздел 2.3.4)	Тестовые задания
Самостоятельное решение задач	Метод приобретения навыков творческого мышления и самостоятельного решения проблем	Типовые задания (раздел 2.3.5), задания для контрольной работы	Самостоятельное решение задач
Лекция-дискуссия	Оценочное средство для включения обучающихся в	Методика проведения лекции-дискуссии (раздел	Лекция-дискуссия

	обсуждение спорных вопросов	2.3.7)	
Интерактивное решение задач	Метод модерации с участием всех обучающихся под руководством преподавателя	Методика проведения интерактивного решения задач (раздел 2.3.8)	Интерактивное решение задач

2.3.2 Вопросы, выносимые на самостоятельное изучение

Теоретический блок:

1. Сравнительный анализ архитектурных подходов к построению GUI в различных библиотеках (Tkinter, PyQt, Kivy).
2. Эволюция парадигм проектирования пользовательских интерфейсов от desktop к мобильным и веб-платформам.
3. Принципы доступности (accessibility) в проектировании интерфейсов для пользователей с ограниченными возможностями.
4. Безопасность GUI-приложений: уязвимости, связанные с обработкой пользовательского ввода.
5. Современные тренды в дизайне интерфейсов: материальный дизайн, неоморфизм, темная тема.

Практический блок:

1. Разработка стратегии тестирования для сложного многооконного приложения.
2. Оптимизация производительности при работе с большими объемами данных в таблицах.
3. Реализация кастомного делегата для отображения сложных данных в QTableView.
4. Создание анимированных переходов между экранами в Kivy-приложении.
5. Интеграция системы контроля версий (Git) в процесс разработки GUI-приложения.

2.3.3 Задания для самостоятельной работы

1. Создайте приложение-калькулятор с графическим интерфейсом на Tkinter, поддерживающее базовые арифметические операции и историю вычислений.
2. Разработайте приложение для управления списком задач (to-do list) с возможностью добавления, удаления, отметки выполнения и сохранения в файл.
3. Реализуйте приложение для просмотра и редактирования изображений с базовыми операциями (поворот, изменение размера, применение фильтров).
4. Создайте клиент для работы с открытым API (например, погодным сервисом) с графическим интерфейсом для отображения результатов.
5. Разработайте приложение для визуализации структуры файловой системы в виде дерева с возможностью навигации.

2.3.4 Тесты по дисциплине

Тест 1. Основы событийно-ориентированного программирования

1. Главный цикл обработки событий в GUI-приложении называется: а) Цикл обновления б) Event loop в) Цикл рендеринга г) Цикл опроса
2. Какой из перечисленных элементов НЕ является частью архитектуры событийно-ориентированного приложения? а) Генератор событий б) Очередь событий в) Диспетчер событий г) Стек вызовов

3. В каком порядке обрабатываются события в стандартной очереди событий?
 а) По приоритету б) В порядке поступления (FIFO) в) В случайном порядке г) По времени обработки

Тест 2. Библиотека Tkinter

4. Какой виджет используется для отображения однострочного текста в Tkinter? а) Text б) Entry в) Label г) Message

5. Какой метод используется для размещения виджета с абсолютными координатами? а) pack() б) grid() в) place() г) layout()

6. Как правильно привязать обработчик к событию нажатия кнопки мыши? а) widget.bind('<Button-1>', handler) б) widget.on_click(handler) в) widget.connect('click', handler) г) widget.onclick = handler

Тест 3. Библиотека PyQt6

7. Какой сигнал генерируется при нажатии кнопки QPushButton? а) clicked б) pressed в) toggled г) activated

8. Как правильно подключить сигнал к слоту в PyQt6? а) button.signal.connect(slot) б) button.connect(signal, slot) в) QObject.connect(button, signal, slot) г) button.on(signal, slot)

9. Какой класс используется для создания таблицы в PyQt6? а) QTable б) QTableView в) QGrid г) QDataTable

Тест 4. Тестирование и отладка

10. Какой фреймворк используется для модульного тестирования в Python? а) pytest б) unittest в) Оба варианта верны г) Нет правильного ответа

11. Какой инструмент позволяет анализировать производительность Python-кода? а) pdb б) cProfile в) logging г) trace

12. Что такое мок-объект (mock object) в контексте тестирования? а) Фиктивный объект, имитирующий поведение реального компонента б) Объект для хранения тестовых данных в) Инструмент для генерации тестовых данных г) Объект для визуализации результатов тестирования

Тест 5. Деплоймент и оптимизация

13. Какой инструмент используется для упаковки Python-приложения в один исполняемый файл? а) pip б) PyInstaller в) setuptools г) wheel

14. Что такое «замораживание» (freezing) приложения? а) Приостановка выполнения приложения б) Упаковка интерпретатора Python и зависимостей в дистрибутив приложения в) Оптимизация кода компилятором г) Создание резервной копии приложения

15. Какой подход рекомендуется для предотвращения утечек памяти в GUI-приложениях? а) Ручное управление памятью б) Использование слабых ссылок (weak references) для обратных связей в) Отключение сборщика мусора г) Периодическая перезагрузка приложения

2.3.5 Типовые задания

Задача 1: Разработка многооконного приложения с передачей данных

Описание

задачи:

Разработайте приложение «Управление клиентской базой юридической фирмы», состоящее из трех окон:

1. Главное окно со списком клиентов
2. Окно добавления/редактирования клиента
3. Окно просмотра детальной информации о клиенте

Требования:

- Использовать архитектуру с разделением на модель, представление и контроллер
- Реализовать передачу данных между окнами без глобальных переменных

- Обеспечить валидацию ввода данных (ФИО, дата рождения, контакты)
- Сохранять данные в локальной базе SQLite
- Поддерживать поиск по клиентам в реальном времени

Критерии оценки:

- Корректность архитектурного решения (3 балла)
- Надежность передачи данных между окнами (2 балла)
- Качество валидации и обработки ошибок (2 балла)
- Производительность при работе со списком из 1000+ записей (2 балла)
- Качество кода и документирование (1 балл)

Задача 2: Создание кастомного виджета визуализации данных

Описание

задачи:

Разработайте кастомный виджет для визуализации статистики по делам юридической фирмы в виде круговой диаграммы с анимацией построения.

Требования:

- Реализовать виджет путем наследования от QWidget (PyQt6) или Canvas (Tkinter)
- Поддерживать динамическое обновление данных с плавной анимацией
- Обеспечить кастомизацию цветовой схемы и стилей
- Добавить всплывающие подсказки при наведении на сегменты диаграммы
- Интегрировать виджет в полноценное приложение с загрузкой данных из файла

Критерии оценки:

- Корректность реализации кастомной отрисовки (3 балла)
- Качество анимации и отзывчивости интерфейса (2 балла)
- Гибкость настройки внешнего вида (2 балла)
- Обработка крайних случаев (пустые данные, большое количество сегментов) (2 балла)
- Интеграция с основным приложением (1 балл)

2.3.6 Задания для контрольной работы

2.3.6.1 Задание к теоретической части контрольной работы

Вариант 1. Архитектурный анализ GUI-фреймворков

Задание:

Проведите сравнительный анализ архитектур трех библиотек визуального программирования на Python (Tkinter, PyQt6, Kivy) по следующим критериям:

1. Модель событий и обработки пользовательского ввода
2. Система управления памятью и жизненным циклом виджетов
3. Поддержка кроссплатформенности и нативный вид на разных ОС
4. Возможности кастомизации внешнего вида
5. Производительность при работе со сложными интерфейсами
6. Экосистема и доступность документации

Предоставьте таблицу сравнения и обоснование выбора библиотеки для разработки корпоративного приложения юридической фирмы с требованиями: кроссплатформенность, поддержка тем оформления, интеграция с базой данных, возможность расширения функционала через плагины.

2.3.6.2 Задание к практической части контрольной работы

Вариант 1. Разработка модуля авторизации

Задание:

Реализуйте модуль авторизации пользователя для GUI-приложения с использованием PyQt6, включающий:

1. Окно входа с полями логин/пароль и кнопкой «Войти»
2. Окно регистрации нового пользователя с валидацией полей
3. Хранение учетных данных в зашифрованном виде (использовать bcrypt или аналог)
4. Систему блокировки после 3 неудачных попыток входа
5. Возможность восстановления пароля через контрольный вопрос
6. Модульные тесты для всех компонентов модуля

Требования к реализации:

- Использовать паттерн «Наблюдатель» для уведомления основного приложения об изменении состояния авторизации
- Обеспечить защиту от SQL-инъекций при работе с базой данных
- Реализовать визуальную обратную связь при ошибках ввода
- Код должен соответствовать стандарту PEP 8

2.3.7 Методика проведения лекции-дискуссии

Цель: Формирование критического мышления при выборе архитектурных решений для GUI-приложений; развитие навыков аргументации профессиональной позиции.

Задачи:

- Анализ преимуществ и недостатков различных подходов к проектированию интерфейсов
- Поиск оптимальных решений для конкретных задач
- Развитие умения отстаивать свою точку зрения с опорой на технические аргументы

Методика проведения:

1. Назначение секретаря лекции-дискуссии и инструктаж по фиксации ключевых аргументов
2. Представление спорной темы (например: «Встроенные средства стилизации vs CSS-подобные подходы в GUI»)
3. Предварительное голосование участников по позициям
4. Свободное обсуждение с обязательным аргументированием каждой позиции
5. Промежуточное резюмирование преподавателем ключевых аргументов «за» и «против»
6. Повторное голосование и анализ изменений позиций участников
7. Формулирование выводов и практических рекомендаций

Критерии оценки:

- Секретарь лекции-дискуссии – 0,5–1,5 балла
- Активное участие с аргументированными выступлениями – 0,5–1 балл за выступление

2.3.8 Методика проведения интерактивного решения задач

Цель: Развитие навыков совместного решения профессиональных задач в условиях, приближенных к реальной разработке ПО.

Задачи:

- Формирование навыков командной работы при разработке интерфейсов
- Развитие умения находить компромиссные решения при проектировании
- Отработка методик декомпозиции сложных задач на подзадачи

Методика проведения:

1. *Подготовительный этап:* Преподаватель разрабатывает проект решения задачи с начальной архитектурой и выделенными проблемными зонами.
2. *Организационный этап:*
 - Объявление темы задачи (например: «Оптимизация производительности таблицы с 10 000 записей»)

- Назначение секретаря и тайм-кепера
 - Распределение ролей в группе (архитектор, разработчик, тестировщик)
 - 3. *Интеллектуальный этап:*
 - Анализ исходного кода и выявление узких мест
 - Мозговой штурм возможных решений
 - Выбор оптимального подхода и распределение задач
 - Совместная реализация решения с поочередным написанием кода
 - 4. *Заключительный этап:*
 - Демонстрация результата и сравнение производительности до/после оптимизации
 - Обсуждение принятых решений и альтернативных подходов
 - Оценка работы группы по критериям: эффективность решения, качество кода, командная работа
- Критерии оценки:*
- Секретарь – 1 балл
 - Активное участие в решении задачи – 1–3 балла
 - Качество предложенного решения – до 2 баллов

3 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков по дисциплине

3.1 Балльно-рейтинговая система оценки успеваемости по дисциплине

В целях оперативного контроля уровня усвоения материала учебной дисциплины и стимулирования активной учебной деятельности студентов (очной формы обучения) используется балльно-рейтинговая система оценки успеваемости. В соответствии с этой системой оценки студенту в ходе изучения дисциплины предоставляется возможность набрать не менее 60 баллов за текущую работу в семестре и до 40 баллов на зачете (итоговый контроль). Результирующая оценка по дисциплине складывается из суммы баллов текущего и итогового контроля.

К зачету допускаются студенты, посетившие все лекции и практические занятия, прошедшие промежуточную аттестацию, с рейтингом не менее 30 баллов.

Балльно-рейтинговая система по дисциплине «Визуальное программирование на языке Python»

Виды контроля	Максимальная сумма баллов на выполнение одного вида задания	Всего возможных за семестр работ	Итого максимальная сумма баллов
Присутствие на лекционных и практических занятиях	0,12	60	7
Работа на практических занятиях (включая интерактивные)	1,0	36	36
СРС, в т.ч.:			17
– написание	0,1	10	1

конспектов лекций			
– выполнение домашних заданий по разработке приложений	0,5	12	6
– подготовка рефератов по современным технологиям GUI	0,3	5	2
– участие в код-ревью проектов одногруппников	0,2	10	2
– разработка документации к проекту	0,3	5	2
– защита итогового проекта (предварительная)	1,0	4	4
Контроль СРС (тестирование, решение кейсов)	0,8	10	8
Итого за семестр	–	–	60
Зачет (защита итогового проекта)	40,0	1	40
Итого по дисциплине	–	–	100

Примечание: Для заочной формы обучения максимальная сумма баллов за СРС увеличена до 55 баллов с соответствующей корректировкой других видов контроля.

Нормы выставления дополнительных баллов

Виды работ	Максимальная сумма баллов
Публикация исходного кода проекта в открытых репозиториях (GitHub/GitLab) с документацией	3
Участие в хакатонах/конкурсах по разработке приложений	4
Доклад на студенческой конференции по тематике дисциплины	3
Итого	10

Шкала итоговой оценки

Количество баллов	Оценка
90–100	отлично
75–89	хорошо
60–74	удовлетворительно
0–59	не зачтено