Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КАЛИНИНГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Л. Г. Высоцкий

ВЫСОКОУРОВНЕВЫЕ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ (ВТП)

учебно-методическое пособие по изучению дисциплины
для студентов, обучающихся в бакалавриате по направлению:
09.03.01 Информатика и вычислительная техника

Калининград
Изд-во ФГБОУ ВО «КГТУ»
2022

УДК 004.9(075)

Рецензент:
кандидат педагогических наук, доцент кафедры прикладной информатики ФГБОУ ВО «Калининградский государственный технический университет» Е. Ю. Заболотнова

Высоцкий, Л. Г.
Высокоуровневые технологии программирования: учебно-методическое пособие по изучению дисциплины для студентов, обучающихся в бакалавриате по направлению подготовки 09.03.01 Информатика и вычислительная техника / **Л. Г. Высоцкий.** – Калининград: Изд-во ФГБОУ ВО «КГТУ», 2022. – 21 с.

Учебно-методическое пособие включает тематический план с расчасовкой и перечень лабораторных работ, которые должны быть выполнены студентами при прохождении курса «Высокоуровневые технологии программирования». Все работы выполняются по индивидуальным заданиям. В пособие также включен список рекомендуемой литературы по курсу.

Учебно-методическое пособие рассмотрено и одобрено в качестве локального электронного методического материала кафедрой прикладной информатики института цифровых технологий ФГБОУ ВО «Калининградский государственный технический университет» 19 сентября 2022 г., протокол № 3

Учебно-методическое пособие рекомендовано к использованию в качестве локального электронного методического материала в учебном процессе методической комиссией ИЦТ 20 сентября 2022 г., протокол № 6

УДК 004.9(075)

ОГЛАВЛЕНИЕ

1. ВВЕДЕНИЕ

Данное учебно-методическое пособие предназначено для студентов направления подготовки 09.03.01 Информатика и вычислительная техника, изучающих дисциплину «Высокоуровневые технологии программирования».

Цель освоения дисциплины:

В результате освоения дисциплины ожидается, что студенты получат комплекс знаний и навыков в области использования на практике объектно-ориентированного подхода в среде событийного и визуального программирования.

Для успешного освоения дисциплины в соответствии с учебным планом ей предшествуют дисциплины «Программирование», «Информатика», «Дискретная математика», «Вычислительная техника», «Операционные системы», «Математическая логика и теория алгоритмов».

Далее в пособии представлен тематический план, содержащий перечень изучаемых тем, обязательных лабораторных работ, мероприятий текущей аттестации и отводимое на них аудиторное время (занятия в соответствии с расписанием) и самостоятельную работу.

В разделе «Содержание дисциплины» приведены подробные сведения об изучаемых вопросах, по которым студент может ориентироваться в случае пропуска каких-то занятий, а также методические рекомендации преподавателя для самостоятельной подготовки, каждая тема имеет ссылки на литературу (или иные информационные ресурсы), а также контрольные вопросы для самопроверки.

Раздел «Текущая аттестация» содержит описание обязательных мероприятий контроля самостоятельной работы и усвоения разделов или отдельных тем дисциплины. Далее изложены требования к завершающей аттестации – экзамену.

В разделе «Балльно-рейтинговая система» приведен порядок применения балльно-рейтинговой системы контроля успеваемости.

Помимо данного пособия, студентам следует использовать материалы, размещенные в соответствующем данной дисциплине разделу ЭИОС, в которые более оперативно вносятся изменения для адаптации дисциплины под конкретную группу.

Техническое обеспечение дисциплины составляют:

1. Операционная система Windows 7 (получаемая по программе Microsoft "Open Value Subscription");
2. Офисное приложение MS Office Standard 2010 (получаемое по программе Microsoft "Open Value Subscription");
3. Kaspersky Endpoint Security;
4. Google Chrome;
5. Python

2. ТЕМАТИЧЕСКИЙ ПЛАН

	Раздел (модуль) дисциплины	Тема	Объем аудиторной работы, ч	Объем самостоятельной работы, ч
		Лекции		
1.1	Основы формирования GUI на базе модуля tkinter	Современные тенденции в программировании	1	1
1.2		Базовые классы виджетов в tkinter-приложениях	2	1
1.3		Классы-переменные tkinter	1	1
1.4		Формирование меню в tkinter-приложениях	2	1
1.5		Основы событийного управления	2	1
1.6		Менеджеры геометрии окон	2	1
2.1	Работа с графикой в tkinter-приложениях	Импорт графики	1	1
2.2		Управление временем в python-приложениях	1	1
2.3		Формирование статических и динамических графических изображений	2	1
			14	**9**
		Лабораторные занятия		
1.1	Основы формирования GUI на базе модуля Tkinter	ЛР 1. Основные элементы пользовательского графического интерфейса	2	3,3
1.2		ЛР 2. Разработка меню	2	3
1.3		ЛР 3. Разработка усложненного Python-проекта	2	2
1.4		ЛР 4. Управление временем в Python-проектах	2	2
2.1	Работа с графикой в tkinter-приложениях	ЛР 5. Графика на основе канвы	2	3
2.2		ЛР 6. Построение графиков	2	4
2.3		ЛР 7. Обработка матриц	2	2
2.4		ЛР 8. Обработка событий	2	2
			16	**21,3**

		Курсовая работа		
1.1	Формирование структуры графического пользовательского интерфейса	Контрольная точка 1. Раздел проекта 1	2	8
1.2	Формирование функционала программного приложения и его привязка к интерфейсу	Контрольная точка 2. Раздел проекта 2	2	12
		Оформление работы. Защита	3	4
			7	**24**

		Рубежный (текущий) и итоговый контроль		
1.1	Рубежный контроль	Лекция № 1	0	1,5
1.2	Рубежный контроль	Лекция № 2	0	1,5
1.3	Рубежный контроль	Лекция № 3	0	1,5
1.4	Рубежный контроль	Лекция № 4	0	1,5
1.5	Рубежный контроль	Лекция № 5	0	1,5
1.6	Рубежный контроль	Лекция № 6	0	1,5
1.7	Рубежный контроль	Лекция № 7	0	1,5
	Итоговый контроль	Экзамен	3	39,2
			3	**49,7**

Всего	**40**	**104**

3. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ И УКАЗАНИЯ К ИЗУЧЕНИЮ

3.1. Раздел 1. Основы формирования GUI на базе модуля Tkinter

Тема 1.1 Современные тенденции в программировании

Перечень изучаемых вопросов

Кратко рассматривается история смены алгоритмических языков как история появления новых методологий программирования. Объясняются различия в подходах к формированию базовых элементов в программах на основе структурной и объектной методологий. Указываются достоинства и недостатки объектной методологии, объясняется сущность визуального программирования и реализуемой на его основе событийной модели управления ходом выполнения программы.

Рассматриваемый в данной теме материал используется на протяжении всего лабораторного практикума, поскольку каждая лабораторная работа предполагает создание некоторого визуального интерфейса, реализующего событийное управление при решении практической задачи, соответствующей рассматриваемой теоретической теме.

Это же в полной мере относится и к курсовой работе, поскольку она представляет также разработку более сложного графического интерфейса с привязанным к нему функционалом.

Методические указания к изучению:

Необходимо очень четко понять разницу в двух основных методологиях программирования: структурной и объектной, - поскольку именно они определяют общий подход к формированию любого программного приложения. С другой стороны, в каждом современном программном продукте имеет место сочетание этих двух методологий.

Событийное управление программой, возникшее благодаря появлению визуального программирования на базе графических дисплеев, используется на протяжении всего лабораторного практикума курса, поэтому сразу надо определить набор базовых событий и механизм их использования для формирования управляющей структуры программы.

Литература

7. Обзор методов и средств визуального программирования и автоматизации программирования, с. 8 – 19.
8. 1.1. Объектно-ориентированное программирование (ООП). С. 8, 9.

Контрольные вопросы

1. Какая методология программирования является исторически первой?
2. Что такое событие?
3. Кто или что может быть источником события?
4. Благодаря каким научным достижениям стало возможным визуальное программирование?
5. Какая модель управления в программе предшествовала событийному управлению?
6. Что такое визуальное программирование?
7. На что декомпозируется проблема (задача) при структурном (императивном) программировании?
8. Что первично при структурном (императивном) программировании: действие или объект действия?
9. На что декомпозируется проблема (задача) при объектном программировании?
10. Что первично при объектном программировании: действие или объект действия?
11. Что первично при объектном программировании: действие или объект действия?

Тема 1.2 Базовые классы виджетов в tkinter-приложениях

Перечень изучаемых вопросов:

Базовые виджеты: кнопки, метки, однострочный и многострочный редакторы, радиокнопки, списки, линейка прокрутки. Их свойства, задание, модификация. Краткий перечень событий, генерируемых виджетами. Привязка функционала приложения к виджетам через события.

Данная теоретическая тема практически закрепляется лабораторными работами «Основные элементы пользовательского интерфейса» и «Разработка усложненного Python-проекта».

В курсовой работе материал этой темы используется активно для формирования пользовательского графического интерфейса как основы всего программного приложения.

Методические указания к изучению

Базовые виджеты являются объектами уже сформированных в среде языка Python классов в парадигме ООП. Поэтому формирование GUI на их основе заключается в параметризации свойств используемых виджетов и их модификации в ходе конструирования программного приложения и/или его выполнения. Подавляющее количество свойств являются общими для всех виджетов как визуальных компонентов интерфейса (положение на форме, размер, цвет фона, текста и т. д.). Поэтому на основе одного какого-то виджета (например, кнопки) целесообразно рассмотреть подробно назначение каждого из свойств, возможные значения, методы изменения, и в дальнейшем, при рассмотрении других виджетов, останавливаться только на их специфических свойствах.

Для обеспечения нормального хода лабораторного практикума необходимо хотя бы в минимальном объеме изучить перечень событий, генерируемых каждым из виджетов и кратко освоить суть обоих механизмов связывания обработчиков событий с виджетами-генераторами этих событий. Важно понять, какие условия должны соблюдаться при использовании каждого из механизмов.

Литература

1. 3.2 Пользовательские подпрограммы и моделирование. Модуль Tkinter. с. 113 – 132
2. 23.1. Основы Tk. с. 78 - 84

Контрольные вопросы

1. Как расшифровывается и переводится GUI?
2. Что такое **виджет**?
3. Какой атрибут у кнопки определяет цвет фона?
4. Сколько есть вариантов задания цвета в языке Python?
5. В каких единицах задаются размеры кнопки?
6. Какая команда (метод) делает кнопку невидимой?
7. Что задает атрибут bg у кнопки?
8. В какое состояние переходит кнопки, если она нажимается мышью?
9. В каком состоянии находится кнопка исходно?
10. Какой метод делает виджет временно невидимым?
11. Какого размера по умолчания генерируется исходное окно модулем tkinter?
12. Сколько есть вариантов импорта модуля tkinter в программу?
13. В каком месте программы должен стоять метод mainloop?
14. Какое свойство позволяет устанавливать размер главного окна в tkinter-программе?
15. В каких единицах устанавливается размер окна в tkinter-программе?
16. Какие параметры шрифта можно задавать у надписей на виджетах?
17. Как прижать окно tkinter-программы к правому нижнему краю экрана?

18. Как вывести текст на заголовок окна tkinter-программы?
19. Какое имя стандартно дается окну tkinter-программы?
20. Может ли пользователь произвольно задавать программное имя окну tkinter-программы?
21. На какие два вида (класса) делятся виджеты?
22. При задании свойств виджетов их надо указывать в строгом или можно в произвольном порядке?
23. Какие параметры можно задавать для текстовых надписей на кнопке?
24. Как называются методы, с помощью которых создаются объекты конкретных классов?
25. Какой метод является деструктором виджетов tkinter?
26. Какой оператор запускает цикл обработки событий в tkinter-приложениях?
27. Как должны указываться названия классов в Python-приложениях?
28. Как задать изменение размера окна tkinter-программы только в определенных пределах?
29. В каких единицах устанавливается положение окна tkinter-программы на экране?
30. Как заблокировать изменение размера окна tkinter-программы во время работы программы?

1.3 Классы-переменные tkinter

Перечень изучаемых вопросов

Назначение переменных модуля tkinter. Виды переменных, их возможные значения. Виджет, использующие эти переменные. Взаимозаменяемость переменных.

Методические указания к изучению

Переменные модуля tkinter играют важную роль в функционировании виджетов, принимающих несколько разных значений в ходе работы приложения. Поэтому важно понять, как задается перечень этих значений, переменные каких типов могут использоваться в каждой конкретной ситуации. Особенно важную роль играют эти переменные в организации нескольких однотипных виджетов в одну радиогруппу, поэтому рекомендуется обязательно рассмотреть пример организации такой радиогруппы.

Материал этой темы используется в каждой лабораторной работе, если ее графический интерфейс содержит виджеты, функционирование которых включает наличие таких переменных.

Это же в полной мере относится и к курсовой работе.

Литература

3. 3.2 Пользовательские подпрограммы и моделирование. Модуль Tkinter. с. 113 – 132.
6. Объекты-переменные. с. 15 – 17.

Контрольные вопросы

1. Можно ли переменную типа BoolenVar использовать для связывания в одну группу радиокнопок?
2. Какие методы обеспечивают установку и считывание переменных tkinter?
3. Сколько есть типов переменных tkinter?
4. Сколько возможных значений может быть у переменной типа IntVar()?
5. Сколько возможных значений может быть у переменной типа BooleanVar()?
6. Переменным каких типов можно одновременно присваивать числовые и логические значения?

Тема 1.4 Формирование меню в tkinter-приложениях

Перечень изучаемых вопросов:

Главное меню. Контекстное меню. Ниспадающее меню. Привязка функционала к опциям меню. Формирование радионаборов из опций меню. Горячие клавиши. Привязка пиктограмм к опциям меню.

Методические указания к изучению

Рекомендуется перед созданием в программном приложении разветвленного меню нарисовать предварительное его дерево. Особое внимание следует обратить на механизм переключения главных меню в одном окне. Для избегания неоднозначностей в ходе работы программы рекомендуется «горячие клавиши» всегда представлять в виде сочетания, как минимум, двух или трех функциональных и алфавитно-цифровых кнопок клавиатуры.

Теоретический материал этой темы поддерживается лабораторной работой «Разработка меню».

В курсовой работе для управления ходом выполнения программного приложения рекомендуется активно использовать разного вида меню, что требует хорошего знания материала этой темы.

Литература

12. Создание меню

Контрольные вопросы

1. Какой атрибут опции меню используется для задания взаимного расположения надписи на опции и пиктограммы?
2. Какой виджет используется для привязки пиктограммы к опции меню?
3. К опциям контекстного меню можно привязывать пиктограммы?
4. Какие графические форматы можно использовать для пиктограмм опций меню?
5. Как вызывается контекстное меню?
6. Одно контекстное меню может быть привязано к разным виджетам?
7. К одному виджету можно привязать несколько контекстных меню?
8. При изменении местоположения виджета на форме сохранится ли привязка к нему контекстного меню?
9. Можно ли сформировать иерархическое контекстное меню, т.е. из нескольких уровней?
10. Какой метод привязывает контекстное меню к виджету?
11. В какой системе координат отображается при вызове контекстное меню на форме?
12. Как связываются опции контекстного меню в один набор?
13. В одной Python-программе может использоваться несколько контекстных меню?
14. Как Python-программа отделяет одно контекстное меню от другого?
15. Как Python-программа отделяет одно контекстное меню от другого?
16. К какой системе координат принадлежит координата event.x?
17. К какой системе координат принадлежит координата event.x_root?
18. Сколько вариантов меню предлагает модуль tkinter?
19. Какие варианты меню всегда присутствуют на форме Python-программы?
20. Как привязывается главное меню к Python-программы?
21. Может ли быть несколько главных меню у Python-программы?
22. Могут ли быть несколько главных меню активными в Python-программе?
23. Если в Python-программе несколько главных меню, то какое из них является активным?
24. Какой метод надо использовать, чтобы создавать «листья» (опции нижнего уровня) в главном меню?

25. Какой метод надо использовать, чтобы создавать «заголовок» выпадающего меню в главном меню?
26. К каким опциям в меню можно привязывать запуск программ?
27. Через какое свойство заносится текст на опцию меню?
28. Может ли «горячая клавиша» быть одиночной?
29. Можно ли в качестве использовать комбинацию из двух клавиш?
30. Где указываются «горячие клавиши» в опции?
31. В чем смысл «горячих клавиш» в меню?
32. Как в меню можно фиксировать последнюю выбранную опцию?
33. Опции выпадающего меню должны формироваться до указания его заголовка, после или это не имеет значения?
34. Для чего при формировании меню используют метод add_checkbutton?
35. Для чего при формировании меню используют метод add_radiobutton()?
36. К опции, сформированной методом add_checkbutton, можно привязывать запуск функций?
37. С какими координатами «работает» метод post()?
38. Что стоит слева от метода post() при его активизации

Тема 1.5 Основы событийного управления

Перечень изучаемых вопросов

Понятие события. Источники событий. События, генерируемые мышью. События, генерируемые клавиатурой. События, генерируемые виджетами интерфейса. Атрибуты событий. Разновидности метода bind.

Методические указания к изучению

Событие – это основа управления в программных приложениях с графическим интерфейсом. Кроме простого перечисления всех событий, генерируемых мышью, виджетами и клавиатурой, необходимо акцентировать внимание на параметрах событий, которые упрощают анализ вводимой информации, позволяют отслеживать координаты мыши на интерфейсе в любой момент времени, реализовывать управление ходом выполнения программы и т.п. Необходимо обратить внимание на различие в синтаксисе описания событий, генерируемых мышью и клавиатурой.

Все работы лабораторного практикума реализуются на основе событийного управления. Более полно данный материал закрепляется практически в ходе выполнения лабораторной работы «Обработка событий».

Всё управление в программном приложении, реализуемом в виде курсовой работы, является событийным.

Литература:

2. 23.3 События. с. 80 – 82
3. События. с. 262 – 265
4. 5.1. Событийно-ориентированное программирование. С. 180 - 186

Контрольные вопросы

1. “<KeyRelease>” – это описание какого события?
2. Какое событие является обратным для “<KeyRelease>”?
3. Синоним события "<Key>"?
4. С какими событиями ассоциируется атрибут num?
5. Что можно выяснить через атрибут num события?
6. Чем отличаются координаты x, y и x_root, y_root мыши?

7. С помощью какого атрибута события можно организовать безэховый ввод?
8. Что такое «безэховый ввод»?
9. С какими событиями ассоциируется атрибут **char**?

Тема 1.6 Менеджеры геометрии окон

Перечень изучаемых вопросов:

Назначение менеджеров геометрии. Менеджер pack(), достоинства и недостатки. Менеджер grid(), достоинства и недостатки. Менеджер place(), достоинства и недостатки.

Методические указания к изучению

Менеджеры геометрии (разметки) играют важную роль в создании эргономичных пользовательских интерфейсов, поэтому важно понять, в каких конкретно ситуациях использовать тот или иной менеджер. Менеджер pack() прост в реализации, но временное сокрытие по каким-либо причинам некоторых виджетов приводит к движению остальных виджетов по экрану, а изменение некоторых параметров виджетов вызывает сопутствующее изменение их формы. Подобные побочные эффекты могут привести к критическим ситуациям при взаимодействии через такие интерфейсы с жизненно важными системами.

Менеджер grid() эффективен при создании интерфейсов с регулярным (матричным) расположением виджетов, но возникают проблемы при модификации таких интерфейсов (прежде всего, добавлении новых виджетов).

Менеджер place() является наиболее гибким в использовании и позволяет производить «тонкую настройку» интерфейсов и, соответственно, является наиболее трудоемким.

Все лабораторные и курсовая работы используют тот или иной вариант менеджера геометрии. При этом студенту рекомендуется тщательно проанализировать и выбрать наиболее подходящий для решаемой задачи менеджер.

Литература

2. 23. Основы Tk, с. 78 – 80.
3. Менеджеры расположения, с. 271 – 273.
4. 5.3.1. Создание и конфигурирование виджета. Менеджер размещения, с. 187 – 190.
6. Менеджеры компоновки, с. 18 – 28.

Контрольные вопросы

1. Какой менеджер геометрии позволяет задавать размер виджета в пикселях?
2. Какое значение установлено по умолчанию у параметра fill менеджера pack()?
3. Перечислите возможные значения параметра expand менеджера pack()?
4. Какое значение установлено по умолчанию у параметра expand менеджера pack()?
5. У каких менеджеров геометрии есть параметр anchor?
6. На какие стороны света указывает это значение параметра anchor = NE?
7. Какие методы позволяют удалять виджеты при использовании менеджера геометрии pack()?
8. У какого менеджера геометрии есть параметр sticky?
9. Какое значение задано по умолчанию атрибуту side менеджера pack()?
10. Какие виджеты-контейнеры используются для получения требуемого расположения виджетов с помощью менеджера pack()?
11. Что задает параметр ipady при использовании менеджера pack()?
12. Перечислите возможные значения параметра fill менеджера pack()?
13. Что произойдет, если задан такой вариант параметра виджета sticky=W+E в менеджере геометрии grid()?

3.2. Раздел 2. Работа с графикой в tkinter-приложениях

Тема 2.1 Импорт графики

Перечень изучаемых вопросов

Средства модуля tkinter, используемые для загрузки готовых графических изображений. Последовательность этапов загрузки. Форматы графических файлов, допускающих загрузку. Преобразование изображений в процессе загрузки: масштабирование, вращение, отображение. Коррекция загружаемых изображений.

Методические указания к изучению:

Импорт графики требует подключение к программному приложению библиотеки PIL. Именно ее методы позволяют реализовать этот процесс сначала загрузкой изображения в контейнер PhotoImage, а затем на пространство канвы canvas. Важно уяснить, что расположение изображения регулируется двумя параметрами: координатами привязки и точкой привязки.

Библиотека PIL содержит целый набор средств для манипулирования изображением при его загрузке. Это подгонка размера канвы под размеры изображения или обратные преобразования, отображения изображения относительно горизонтали и вертикали, вращение изображения по часовой стрелке с дискретом 90^0. При увеличении изображения происходит потеря его качества. С целью минимизации этих потерь библиотека PIL содержит ряд фильтров, ориентированных на конкретные виды графики. Поэтому рекомендуется при работе с конкретным изображением испробовать все варианты фильтрации и выбрать из них лучший.

В лабораторных работах «Основные элементы пользовательского интерфейса», «Разработка усложненного Python-проекта» и «Разработка меню» часть заданий требует загрузки на канву или другие виджеты готовых графических изображений. Такая же ситуация может возникнуть и в процессе выполнения курсовой работы, если это требуется выбранным вариантом задания.

Литература

3. Изображения в Tkinter, с. 273 – 279
13. Получение информации об изображении через Pillow. Обрезка изображений через Pillow (crop). Поворачивание изображения — метод rotate() Pillow.

Контрольные вопросы

1. Какие виды фильтров включает библиотека PIL?
2. Что реализует вариант преобразования изображения FLIP_TOP_BOTTOM?
3. В векторной графике размер графического файла зависит от сложности изображения?
4. В растровой графике размер графического файла зависит от сложности изображения?
5. Какой метод библиотеки PIL масштабирует размер канвы по размеру графического изображения?
6. Почему качество компьютерного графического изображения ухудшается при его увеличении?
7. Можно ли повернуть графическое изображение средствами библиотеки PIL на произвольный угол?
8. Как библиотека PIL подключается к tkinter-приложению?
9. Какие варианты отображения графического изображения доступны в библиотеке PIL?

10. Изображения каких графических форматов можно загрузить средствами библиотеки PIL?

Тема 2.2 Управление временем в python-приложениях

Перечень изучаемых вопросов

Методы формирования временных интервалов в Python-программах: after() и sleep(). Специфика использования метода after(). Специфика использования метода sleep(). Регенерация изображения при создании динамических эффектов. Реализация временных интервалов на основе системного времени компьютера. Управление временем в циклических и параллельных процессах.

Методические указания к изучению

Создание динамических изображений в графических интерфейсах возможно только на основе механизмов управления временем. Для этого в состав модуля tkinter введено несколько соответствующих средств. Необходимо четко разбираться в их возможностях. Метод after() не позволяет создавать временные интервалы меньше 1 млсек. Метод sleep() реализует временные интервалы любой длительности, но требует подключение к модулю класса time. В обоих случаях необходимо использовать механизм регенерации изображения после каждого его изменения. Системное время не требует циклической регенерации изображения, на его базе также можно организовать параллельные процессы программе.

Изучению механизмов использования времени отдельно посвящена лабораторная работа «Управление временем в Python-проектах». Также эти вопросы изучаются и в лабораторной работе «Графика на основе канвы». Если в курсовой работе реализуется динамический графический процесс, то он обязательно использует рассматриваемые здесь средства.

Литература

4. 2.7. Работа с датой и временем, с. 97 – 103.
5. 5. Модули. Время и дата, с. 253, 254.
14. Работа с датой и временем в Python

Контрольные вопросы

1. Какой класс позволяет использовать в Python-программе системные часы?
2. Какой класс позволяет использовать в Python-программе системные часы?
3. Какой метод класса datetime используется для работы с системными часами?
4. Как подключаются системные часы к Python-программе?
5. Можно ли при работе со временем в tkinter-приложениях на основе метода after() установить временной интервал меньше одной миллисекунды?
6. Можно ли при работе со временем в tkinter-приложениях на основе метода sleep() установить временной интервал меньше одной миллисекунды?
7. К какому модулю относится метод after()?
8. Сколько одновременно значений может выдавать метод time класса datetime?
9. В каких единицах фиксирует временные отрезки метод after()?
10. В каких единицах фиксирует временные отрезки метод sleep()?

Тема 2.3 Формирование статических и динамических графических изображений

Перечень изучаемых вопросов

Виджет Canvas – полотно для создания графики средствами языка Python. Понятие векторной графики. Основные методы создания графических примитивов: полилинии, по-

лигона, прямоугольника, овала, дуги, сектора, текста. Методы перемещения фигур. Понятие тега.

Методические указания к изучению

Необходимо четко осознать различие в растровой и векторной графиках, поскольку эта разница существенно влияет на методы создания простейших графических примитивов и средства манипулирования ими в ходе работы программы. В векторной графике возможно менять только параметры всей фигуры в целом, а не отдельных ее частей. Также перемещение фигур не требует стирания фигур на старом месте и их прорисовки на новом, а реализуется просто перемещением фигуры. При этом надо помнить, что любые преобразования возможны только тех фигур. У которых есть идентификаторы или теги. С другой стороны, созданные в векторной графике изображения требуют дополнительных средств для их сохранения в виде графических файлов.

Теоретический материал этой темы практически закрепляется в лабораторных работах «Графика на основе канвы», «Построение графиков» и «Обработка матриц». Курсовая работа, предполагающая создание статических или динамических графических изображений средствами языка Python, также опирается на данный материал.

Литература

1. 3.2 Пользовательские подпрограммы и моделирование. Модуль Tkinter, с. 113 – 132
3. Изображения в Tkinter, с. 274 – 276.
4. 53.9. Canvas (изображение), с. 203 – 207.

Контрольные вопросы

1. Какой тип графики реализуется виджетом Canvas?
2. Какая фигура создается, если у метода create_arc отсутствует параметр style?
3. В каком виде компьютерной графики можно оперировать отдельным пикселем?
4. Какой параметр задает цвет линии на Canvas?
5. В растровой графике размер графического файла зависит от сложности изображения?
6. В векторной графике размер графического файла зависит от сложности изображения?
7. Какие значения могут быть у параметра arrow линии?
8. Сколько есть стандартных методов перемещения графических фигур по канве?
9. Можно ли перемещать графическую фигуру по канве, если у нее нет идентификатора (имени)?
10. Чем полигон отличается от обычной ломаной линии при рисовании на Canvas?
11. Количество параметров у метода move() постоянно или может меняться?
12. Количество параметров у метода itemconfig() постоянно или может меняться?
13. Что задает параметр dash при рисовании линии?
14. Количество параметров у метода coords() постоянно или может меняться, если метод используется для задания новых параметров?
15. Какие параметры определяют положение текста на канве Canvas?
16. Какой метод позволяет одновременно перемещать графическую фигуру и менять ее цветовые параметры?
17. Как нарисовать пунктирную линию?
18. Может ли у графической фигуры быть одновременно и имя, и тег?
19. В какую фигуру вписывается создаваемый овал?
20. Какая операция циклически выполняется при рисовании мышью по канве?
21. Сколько вариантов значений есть у параметра ARROW метода create_line?

22. Какие графические примитивы, созданные на Canvas, нельзя уничтожить отдельно в ходе работы программы?
23. Какие есть варианты идентификации графических примитивов на канве Canvas?
24. Какой графический примитив канвы Canvas также называется fill area?
25. Чем отличаются графические примитивы polyline и polygon?
26. Какой метод создает на канве сектор?
27. Сколько параметров минимально надо задать для создания на канве сектора?

4. ТРЕБОВАНИЯ К АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

4.1. Текущая аттестация

В ходе изучения дисциплины студентам предстоит пройти следующие этапы текущей аттестации:

1. Опрос после каждой лекции, представляющий ответ на пять вопросов. Правильный ответ на каждый вопрос добавляет балл к итоговой оценке.
2. Оценка вовремя выполненной курсовой работы по пятибалльной системе.
3. Наличие к началу сессии всех выполненных и защищенных лабораторных работ.

4.2. Условия получения положительной оценки

Завершающим этапом изучения дисциплины является промежуточная аттестация, представляющая собой:

Условие получения положительной оценки по БРС является наличие сданной курсовой работы и выполненный полностью к началу сессии лабораторный практикум. К оценкам, полученным в ходе опроса после каждой лекции, добавляется оценка за курсовую работу, после чего находится средняя оценка, которая округляется до ближайшего целого числа. Если студент не соглашается с полученной оценкой, он сдает экзамен по традиционной схеме.

Студенты, не выполнившие план учебной работы в течение семестра, а также при наличии ликвидации всех задолженностей, сдают экзамен по традиционной схеме.

4.3. Примерные вопросы к экзамену по дисциплине

1. Структура tkinter-программы.
2. Окно tkinter-программы.
3. Виджет Button: основные свойства.
4. Основные события, используемые в tkinter-приложениях.
5. Методы привязки функций к виджетам через события.
6. Виджет Label: основные свойства.
7. Виджет Entry: основные свойства и методы.
8. Виджет Text: основные свойства и методы.
9. Виджет Listbox: основные свойства и методы.
10. Виджет Radiobutton: основные свойства и методы.
11. Переменные модуля tkinter.
12. Формирование меню в tkinter-приложениях.
13. Виджет Scale: основные свойства и методы.
14. Виджет Checkbutton: основные свойства и методы.
15. Менеджеры разметки
16. Виджет Combobox: основные свойства и методы
17. Виджет Spinbox: основные свойства и методы
18. Виджет Progressbar: основные свойства и методы
19. Кнопка Speedbutton: варианты реализации.
20. Виджет Notebook: основные свойства и методы.
21. Средства работы со временем в tkinter-приложениях.
22. Импорт графики в tkinter-приложения.
23. Виджет Canvas: средства формирования статических и динамических изображений
24. События, генерируемые мышью.
25. События, генерируемые клавиатурой.
26. Свойства событий.
27. Работа с файлами в tkinter-приложениях.

5. ЗАКЛЮЧЕНИЕ

УМП по изучению дисциплины **«Высокоуровневые технологии программирования»** включает описание всех компонентов, требуемых для эффективного освоения данного предмета студентами направления подготовки 09.03.01 Информатика и вычислительная техника.

6. ЛИТЕРАТУРА

Основная литература:

1. Хахаев, И. А. Практикум по алгоритмизации и программированию на Python: курс : учеб. пособие : [16+] / И. А. Хахаев. – 2-е изд., исправ. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 179 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=429256 (дата обращения: 30.03.2022). – Библиогр. в кн. – Текст : электронный.
2. Буйначев, С. К. Основы программирования на языке Python: учебное пособие / С. К. Буйначев, Н. Ю. Боклаг ; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург : Изд-во Уральского университета, 2014. – 92 с. : табл., ил. – Режим доступа: по подписке. – URL https://biblioclub.ru/index.php?page=book&id=275962 (дата обращения: 30.03.2022). – Библиогр. в кн. – ISBN 978-5-7996-1198-9. – Текст : электронный.
3. Сузи, Р. А. Язык программирования Python: учебное пособие : [16+] / Р.А.Сузи. – 2-е изд., испр. – Москва : Интернет-Университет Информационных Технологий (ИНТУИТ) : Бином. Лаборатория знаний, 2007. – 327 с. – (Основы информационных технологий). – Режим доступа: по подписке. – URL:https://biblioclub.ru/index.php?page=book&id=233288 (дата обращения: 30.03.2022). – ISBN 978-5-9556-0109-0. – Текст : электронный.
4. Жуков Р.А. Язык программирования Python: практикум: учеб, пособие / Р.А. Жуков. — Москва: ИНФРА-М, 2019. — 216с. + Доп. материалы [Электронный ресурс]; Режим доступа: https://codernet.ru/books/python/yazyk_programmirovaniya_python_praktikum/ (дата обращения: 27.03.2022).
5. Саммерфилд, М. Программирование на Python 3. Подробное руководство. – Пер. с англ. – Санкт-Петербург: Символ-Плюс, 2009. – 608 с., ил. - Электрон. дан. – Режим доступа: http://uchcom7.botik.ru/L/prog/python/python_08.pdf. (дата обращения: 27.03.2022).
6. Лекция 14 Разработка приложений с графическим интерфейсом пользователя Библиотека Tkinter – Текст: [Электронный ресурс]; Режим доступа: http://python.inr.ru/s1914.pdf . (дата обращения: 24.04.2022).

Дополнительная литература:

7. Методы и технологии визуального программирования: Учебное пособие / Коварцев А.Н., Жидченко В.В., Попова-Коварцева Д.А. – Самара: ООО «Офорт», 2017. - 197 с.: 83 ил материалы [Электронный ресурс; Режим доступа:
28. http://repo.ssau.ru/bitstream/Uchebnye-posobiya/Metody-i-tehnologii-vizualnogo-programmirovaniya-66439/3/Учебник%20визуальное%20программирование%202018_итог.pdf (дата обращения: 23.04.2022).
8. Карташов, Д. В. К 27 Объектно-ориентированное программирование: учебно-методическое пособие / Д. В. Карташов, Л. А. Непомнящая. – Томск : Изд-во ТГПУ, 2019. – 52 с. материалы [Электронный ресурс]; Режим доступа: http://fulltext.tspu.edu.ru/OA/m2019-29.pdf (дата обращения: 23.04.2022).
9. Хахаев И. А. Практикум по алгоритмизации и программированию на Python: / И. А. Хахаев — Москва : Альт Линукс, 2010. — 126 с. : ил. — (Библиотека ALT Lin17 с. ux). - Электрон. дан. – Режим доступа: https://all-journals.com/books/program-

ming/13066-praktikum-po-algoritmizacii-i-programmirovaniyu-na.html (дата обра-щения: 29.03.2022).
10. Учись tkinter. – Бесплатная электронная книга, 38 с. - Электрон. дан. – Режим доступа: https://riptutorial.com/Download/tkinter-ru.pdf (дата обращения: 27.03.2022).
11. Python. Лекция 12. Создание приложений с GUI, 17 с. - Электрон. дан. – Режим доступа: https://ideafix.name/wp-content/uploads/2012/08/Python-12.pdf (дата обращения: 27.03.2022)
12. 3 вида меню в Tkinter: создание, наполнение верхнего и выпадающего меню [Электронный ресурс]; Режим доступа: https://pythonru.com/uroki/sozdanie-menju-tkinter-10 (дата обращения: 25.04.2022).
13. Pillow обработка изображений в Python на примерах [Электронный ресурс]; Режим доступа: https://python-scripts.com/pillow (дата обращения: 28.04.2022).
14. Работа с датой и временем в Python [Электронный ресурс]; Режим доступа: https://python-scripts.com/datetime-time-python (дата обращения: 28.04.2022).

Локальный электронный методический материал

Леонид Григорьевич Высоцкий

ВЫСОКОУРОВНЕВЫЕ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Редактор Г. А. Смирнова

Уч.-изд. л. 1,25. Печ. л. 1,25.

Издательство федерального государственного бюджетного образовательного учреждения высшего образования
«Калининградский государственный технический университет».
236022, Калининград, Советский проспект, 1