МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«ИВАНОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Шуйский филиал ИвГУ

Кафедра математики, информатики и методики обучения

ОДОБРЕНО:

Руководитель ОП

_____ (подпись) С.В. Белов

«21» февраля 2023 г.

Рабочая программа дисциплины

ПРОГРАММИРОВАНИЕ

Уровень высшего образования:	бакалавриат
Квалификация выпускника:	бакалавр
Направление подготовки:	44.03.01 Педагогическое образование
Направленность (профиль) образовательной программы:	Информационные технологии и робототехника

Шуя

1. Цели освоения дисциплины

содействие профессиональному становлению педагога и развитию его компетенции в области проектной деятельности по информационным технологиям и робототехники через развитие навыков алгоритмизации, формирование у студентов умения разрабатывать программы на языке программирования высокого уровня Python, в среде программирования Python IDLE.

2. Место дисциплины в структуре ОП

Дисциплина «Программирование» является обязательной для изучения; относится к обязательной части блока дисциплин Б1, входит в предметно-методический модуль.

Успешное освоение данной дисциплины будет способствовать готовности обучающихся к освоению дисциплин: «Информационные технологии в образовании»; «Образовательная робототехника»; «Соревновательная робототехника»; «Теория и методика обучения информационным технологиям»; «Базы данных»; «Системное программирование»; «Прикладное программирование»; «Основы искусственного интеллекта»; «Логическое программирование»; «Теория и методика обучения робототехнике»; успешному прохождению следующих практик: «Учебная практика, технологическая (проектно-технологическая) практика, по программированию (Предметно-методический модуль)», «Учебная практика, технологическая (проектно-технологическая) практика, по информационным технологиям и робототехнике (Предметно-методический модуль)», подготовке к сдаче и сдаче государственного экзамена, выполнению и защите выпускной квалификационной работы.

Студент, приступающий к изучению данной дисциплины, должен:

Знать:

– основы современных технологий сбора, обработки и представления информации;

Уметь:

– получать, обрабатывать и анализировать информацию из различных источников;

– создавать и редактировать простейшие графические изображения;

– использовать ресурсы сети интернет для самообразования в области технического творчества;

Владеть:

– навыками сопоставления разных источников информации с целью выявления их противоречий и поиска достоверных суждений;

– технологиями анализа и синтеза информации на основе системного подхода.

3. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

3.1. Компетенции, формированию которых способствует данная дисциплина

При реализации дисциплины формируются следующие компетенции в соответствии с ФГОС ВО по данному направлению подготовки:

а) универсальные компетенции (УК) и индикаторы их достижения:

Наименование категории (группы) универсальных компетенций	Код и наименование универсальной компетенции выпускника	Код и наименование индикатора достижения универсальной компетенции
Системное и критическое	УК-1. Способен осуществлять	УК-1.1. Демонстрирует знание

мышление	поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	особенностей системного и критического мышления и готовность к нему. УК-1.2. Применяет логические формы и процедуры, способен к рефлексии по поводу собственной и чужой мыслительной деятельности. УК-1.3. Анализирует источник информации с точки зрения временных и пространственных условий его возникновения. УК-1.4. Анализирует ранее сложившиеся в науке оценки информации. УК-1.5. Сопоставляет разные источники информации с целью выявления их противоречий и поиска достоверных суждений. УК-1.6. Аргументировано формирует собственное суждение и оценку информации, принимает обоснованное решение. УК-1.7. Определяет практические последствия предложенного решения задачи.
Разработка и реализация проектов	УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1. Определяет совокупность взаимосвязанных задач, обеспечивающих достижение поставленной цели, исходя из действующих правовых норм. УК-2.2. Определяет ресурсное обеспечение для достижения поставленной цели. УК-2.3. Оценивает вероятные риски и ограничения в решении поставленных задач. УК-2.4. Определяет ожидаемые результаты решения поставленных задач.

б) общепрофессиональные компетенции (ОПК) и индикаторы их достижения:

Наименование категории (группы) общепрофессиональных компетенций	Код и наименование общепрофессиональной компетенции выпускника	Код и наименование индикатора достижения общепрофессиональной компетенции
Научные основы педагогической деятельности	ОПК-8. Способен осуществлять педагогическую деятельность на основе специальных научных знаний	ОПК-8.1. Применяет методы анализа педагогической ситуации, профессиональной рефлексии на основе специальных научных знаний. ОПК-8.2. Проектирует и осуществляет учебно-воспитательный процесс с опорой на знания основных закономерностей возрастного развития когнитивной и личностной сфер обучающихся, научно-обоснованных закономерностей организации образовательного процесса, научных основ преподаваемых предметов, их истории и места в мировой культуре и науке.

		ОПК-8.3. Осуществляет мониторинг и отбор программ профессионального и личностного развития.

в) профессиональные компетенции (ПК) и индикаторы их достижения:

Код и наименование профессиональной компетенции выпускника	Код и наименование индикатора достижения профессиональной компетенции	Основание (ПС, анализ опыта)
Тип задач профессиональной деятельности: проектный		
ПК-9. Способен участвовать в проектировании, конструировании, программировании, настройке, экспериментальной проверке и презентации созданных робототехнических конструкций.	ПК-9.1. Умеет использовать ресурсы сети интернет для самообразования в области технического творчества. ПК-9.2. Осуществляет проектирование и конструирование робототехнических устройств. ПК-9.3. Разрабатывает и реализовывает алгоритмы по управлению манипуляторами. ПК-9.4. Самостоятельно создает программы и осуществляет обучение школьников программированию робототехнических устройств. ПК-9.5. Организовывает экспериментальную проверку и презентацию созданных робототехнических конструкций.	Профессиональный стандарт 01.003 «Педагог дополнительного образования детей и взрослых»

3.2. Планируемые результаты обучения по дисциплине, соотнесенные с индикаторами достижения формируемых компетенций

В результате освоения дисциплины обучающийся должен:

Знать:

– логические формы и процедуры (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– особенности проектного мышления (УК-2 (индикаторы – 1, 2, 3, 4);

– основные этапы проектирования, их последовательность и взаимосвязь (УК-2 (индикаторы – 1, 2, 3, 4);

– научные основы преподаваемого предмета «Информационные технологии», его историю и место в мировой культуре и науке (ОПК-8 (индикаторы – 1, 2, 3);

– этапы развития предметной области алгоритмизации и программирования (ОПК-8 (индикаторы – 1, 2, 3);

– содержательные, методологические и мировоззренческие связи предметной области со смежными научными областями (ОПК-8 (индикаторы – 1, 2, 3);

– основные программы для организации обучения школьников программированию робототехнических устройств (ПК-9 (индикаторы – 1, 2, 3, 4, 5);

Уметь:

– аргументированно формировать собственное суждение и оценку информации (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– осуществлять корректный подбор методов анализа, проводить обработку данных исследования и правильную интерпретацию результатов (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– формулировать задачи на основе этапов получения промежуточных результатов (УК-2 (индикаторы – 1, 2, 3, 4);

– оценивать уровень и качество каждого ресурса, обеспечивающего выполнение определенной задачи (УК-2 (индикаторы – 1, 2, 3, 4);

– руководствоваться в профессиональной деятельности научными основами преподаваемого предмета «Информационные технологии» (ОПК-8 (индикаторы – 1, 2, 3);

– соотносить основные этапы развития предметной области с ее актуальными задачами, методами и концептуальными подходами, тенденциями и перспективами ее современного развития (ОПК-8 (индикаторы – 1, 2, 3);

– самостоятельно создавать программы и осуществлять обучение школьников программированию робототехнических устройств (ПК-9 (индикаторы – 1, 2, 3, 4, 5);

Владеть:

– способами решения конкретных задач проекта на уровне заявленного качества и за установленное время (УК-2 (индикаторы – 1, 2, 3, 4);

– навыками публичного представления результатов решения конкретной задачи проекта (УК-2 (индикаторы – 1, 2, 3, 4);

– научными основами преподаваемого предмета «Информационные технологии» (ОПК-8 (индикаторы – 1, 2, 3);

– теоретическими и практическими знаниями для постановки и решения исследовательских задач в предметной области и в области образования (ОПК-8 (индикаторы – 1, 2, 3);

– навыками организации экспериментальной проверки и разработки презентаций созданных робототехнических конструкций (ПК-9 (индикаторы – 1, 2, 3, 4, 5).

4. Объём и содержание дисциплины

Очная форма обучения

Объём дисциплины составляет 3 зачётных единиц (81 час.)
Контактная работа 25,75 час.
Самостоятельная работа 55,25 час.

Виды учебных занятий и формы промежуточной аттестации	4 сем.	
	Контактная работа с преподавателем	Самостоятельная работа обучающегося
Лекции	12	55,25
Практические занятия	13,5	
Зачёт с оценкой	0,25	
Всего	25,75	55,25

4.1. Содержание дисциплины по разделам (темам), соотнесенное с видами и объемом учебных занятий

Очная форма обучения

№ п/п	Раздел (тема) дисциплины	Семестр	Виды учебных занятий, включая самостоятельную работу обучающихся и объем (в часах)				Формы текущего контроля успеваемости Формы промежуточной аттестации
			Занятия лекционного типа	Занятия семинарского типа		Самостоятельная работа обучающегося	
				Лаб.	Пр.		
1.	Подпрограммы. Процедуры и функции	4	1,5	-	1,5	7,5	практические работы
2.	Организация ввода-вывода данных при помощи файлов	4	1,5	-	1,5	8,75	практические работы
3.	Основные принципы объектно-ориентированного программирования (ООП)	4	1,5	-	1,5	9	практические работы
4.	Этапы разработки приложения	4	1,5	-		7,5	практические работы
5.	Визуальное событийно-управляемое программирование	4	1,5	-	1,5	7,5	
6.	Разработка оконного приложения	4	1,5	-	4,5	15	
ИТОГО			12	-	13,5	55,25	Зачёт с оценкой

4.2. Развернутое описание содержания учебного материала по разделам (темам)

Подпрограммы. Процедуры и функции. Понятие подпрограммы. Процедуры и функции, их сущность, назначение, различие. Функции: способы организации и описание в Python. Вызов функций, рекурсия. Программирование рекурсивных алгоритмов. Создание модулей.

Организация ввода-вывода данных при помощи файлов. Чтение информации из текстового файла. Запись информации в текстовый файл. Запись и чтение информации из двоичного файла.

Основные принципы объектно-ориентированного программирования (ООП). История развития ООП. Базовые понятия ООП: объект, его свойства и методы, класс, интерфейс. Основные принципы ООП: инкапсуляция, наследование, полиморфизм. Событий-но-управляемая модель программирования. Компонентно-ориентированный подход. Классы объектов. Компоненты и их свойства. Иерархия классов. Классы объектно-ориентированного языка программирования: виды, назначение, свойства, методы, события. Объявление класса, свойств и методов экземпляра класса. Наследование. Перегрузка методов.

Этапы разработки приложения. Постановка задачи. Создание программы. Тестирование, отладка приложения. Создание документации (документирование программы).

Визуальное событийно-управляемое программирование. Создание интерфейса пользователя. Основы работы с Tkinter. Создание формы и виджетов: «Кнопка», «Текстовое поле», «Надпись». Создание виджета «Флажок». Создание виджета «Переключатель».

Разработка оконного приложения. Настройка внешнего вида окон. Определение объектных связей между виджетами. Различные варианты компоновки виджетов. Обеспечение взаимодействия элементов управления посредством сигналов/слотов.

5. Образовательные технологии, в том числе перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине

В программе данного курса предусмотрено использование в учебном процессе активных форм проведения занятий и организации самостоятельной работы, рейтинговой технологии оценивания результатов обучения студентов. В учебном процессе используются как активные, так и интерактивные формы проведения занятий: технология учебной дискуссии, мозговой штурм.

Аудиторные занятия проводятся в интерактивной форме с использованием мультимедийного обеспечения (ноутбук, проектор) и технологии проблемного обучения.

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине: технологии смешанного обучения, мультимедиа технологии, облачные технологии, технологии визуализации (инфографика, презентации).

6. Учебно-методическое обеспечение самостоятельной работы обучающихся

Самостоятельная работа обучающихся по дисциплине осуществляется в следующих формах:

- установка и настройка программного обеспечения для работы с языком программирования;
- написание и тестирования программ на языке Python;
- изучение и анализ предложенных электронных образовательных ресурсов, содержащих информацию по алгоритмизации и программированию;
- повторение материала с использованием конспектов.

Полностью весь методический материал по обеспечению самостоятельной работы студентов приводится в приложении 1 к настоящей программе. Доступ к методическим материалам осуществляется через сайт Шуйского филиала ИвГУ, в том числе через систему Moodle.

7. Характеристика оценочных средств для текущего контроля успеваемости и промежуточной аттестации по дисциплине

Текущая аттестация осуществляется с использованием накопительного рейтингового оценивания.

Для текущего контроля успеваемости применяются:

- Практические работы на компьютере: 8 работ по 3 балла каждая (24 балла);
- Практические задания для самостоятельного выполнения: 28 заданий по 2 балла (56 балла);
- Опорный конспект по темам, предложенным преподавателем (10 баллов);

Помимо этого учитывается:

- посещаемость (отсутствие пропусков занятий) – 10 баллов.

Критерии оценки практических работ:

- 3 балла за практическую работу выставляется, если работа выполнена полностью правильно и самостоятельно, а студент способен уверенно отвечать на вопросы преподавателя о проделанной работе;

- 2 балла выставляется, если работа выполнена в целом корректна, но имеются неточности в ответах на вопросе преподавателя или некоторые действия выполнены неоптимально;
- 1 балл выставляется, когда работа не может быть завершена студентом самостоятельно, без помощи преподавателя.
- 0 баллов выставляется, если работа не была выполнена.

Критерии оценки практических заданий для самостоятельного выполнения:

- 2 балла за практическую работу выставляется, если работа выполнена полностью правильно и самостоятельно, а студент способен ответить на вопросы преподавателя о проделанной работе;
- 1 балл выставляется, если работа выполнена в целом корректна, но имеются неточности в ответах на вопросе преподавателя;
- 0 баллов выставляется, если работа не была выполнена.

Критерии оценки опорного конспекта по темам: Написание конспекта по каждой из пяти тем оценивается от 0 до 2 баллов: 2 балла – в конспекте отражены основные положения темы, имеются схемы, выделения особо значимой информации; 1 балл - в конспекте отражены основные положения темы, но отсутствуют признаки попытки переработки информации со стороны студенты; 0 баллов – основные положения темы не были отражены в конспекте.

Шкала количества баллов, выставляемых за посещение занятий:

Процент посещения занятий от их общего количества	**Количество баллов, выставляемое за посещаемость**
85%-100%	10 баллов
80%-84%	9 баллов
75%-79%	8 баллов
70%-74%	7 баллов
65%-69%	6 баллов
60%-64%	5 баллов
55%-59%	4 балла
50%-54%	3 балла
45%-49%	2 балла
40%-44%	1 балл
0%-39%	0 баллов

Оценка результатов освоения дисциплины при рейтинговой системе оценивания формируется по 100-балльной шкале из результатов текущего контроля успеваемости и результатов, полученных на зачёте. Зачёт проводится в форме выполнения практической работы (максимально 10 баллов) и интегрального тестового задания, включающего вопросы по всем темам курса (максимально 20 баллов).

Оценка на зачёте выставляется по следующей шкале:

Шкала оценки результатов освоения дисциплины «Программирование» при рейтинговой системе оценивания

Повышенный уровень	Базовый уровень	Пороговый уровень	Ниже порогового уровня

отлично	хорошо	удовлетворительно	неудовлетворительно
85-100 баллов	71-84 балла	61-70 баллов	60 и менее баллов

8. Учебно-методическое и информационное обеспечение дисциплины

Учебная и учебно-методическая литература:

– основная литература:

1. Аникеева Л. А. Программирование на алгоритмическом языке КуМир : учебное пособие / Л. А. Аникеева, О. Б. Гусева. – Москва : СОЛОН-ПРЕСС, 2012. – 48 с. – («Элективный курс. Профильное обучение»). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=226976 (дата обращения: 08.03.2023). – ISBN 978-5-91359-098-5. – Текст : электронный.
2. Лубашева, Т.В. Основы алгоритмизации и программирования : учебное пособие / Т.В. Лубашева, Б.А. Железко. - Минск : РИПО, 2016. - 378 с. : ил. - Библиогр. в кн. - ISBN 978-985-503-625-9 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=463632
3. Незнанов, А. А. Программирование и алгоритмизация : учебник / А. А. Незнанов ; науч. ред. В. П. Кутепов. - Москва : Академия, 2010. - 304 с.

– дополнительная литература:

1. Зюзьков, В.М. Программирование : учебное пособие / В.М. Зюзьков ; Министерство образования и науки Российской Федерации, Томский Государственный Университет Систем Управления и Радиоэлектроники (ТУСУР). - Томск : Эль Контент, 2013. - 186 с. : ил. - Библиогр. в кн. - ISBN 978-5-4332-0141-5 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=480616
2. Информатика : учебное пособие / Е. Н. Гусева, И. Ю. Ефимова, Р. И. Коробков [и др.]. – 5-е изд., стер. – Москва : ФЛИНТА, 2021. – 260 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=83542 – Библиогр. в кн. – ISBN 978-5-9765-1194-1. – Текст : электронный.
3. Карякин, М. И. Технологии программирования и компьютерный практикум на языке Python : учебное пособие : [16+] / М. И. Карякин, К. А. Ватульян, Р. М. Мнухин ; Южный федеральный университет. – Ростов-на-Дону ; Таганрог : Южный федеральный университет, 2022. – 244 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=698687
4. Кирнос, В.Н. Информатика II. Основы алгоритмизации и программирования на языке C++ : учебно-методическое пособие / В.Н. Кирнос ; Министерство образования и науки Российской Федерации, Томский Государственный Университет Систем Управления и Радиоэлектроники (ТУСУР). - Томск : Эль Контент, 2013. - 160 с. : ил.,табл., схем. - ISBN 978-5-4332-0068-5 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=208651
5. Основы алгоритмизации и программирования : лабораторный практикум / Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Северо-Кавказский федеральный университет» ; сост. Е.И. Николаев. - Ставрополь : СКФУ, 2015. - 211 с. : ил. - Библиогр. в кн. ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=457504
6. Хахаев, И.А. Практикум по алгоритмизации и программированию на Python : курс / И.А. Хахаев. - 2-е изд., исправ. - Москва : Национальный Открытый Университет «ИНТУИТ»,

2016. - 179 с. : ил. - Библиогр. в кн. ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=429256

Интернет-ресурсы:

1. Самоучитель Python 3 для начинающих. Режим доступа: https://pythonworld.ru/
2. Официальный сайт языка программирования Python. Режим доступа: https://www.python.org/
3. Официальный сайт среды программирования PyCharm. Режим доступа: https://www.jetbrains.com/pycharm/
4. Кулибин — среда программирования виртуальных роботов. Режим доступа: https://kulibin.app/
5. Единая коллекция цифровых образовательных ресурсов http://school-collection.edu.ru/
6. Образовательные сайты и порталы. http://school2100.com/
7. Видеоматериалы для учителей. https://video.1sept.ru/subject/programmirovanie/archive
8. Профессиональный сайт учителей «Методисты». http://nsportal.ru/

Современные профессиональные базы данных и информационно-справочные системы:
ЭБС «Университетская библиотека онлайн»;
Коллекция Шуйского филиала ИвГУ на национальном цифровом ресурсе «Руконт».
Консультант Плюс http://www.consultant.ru/
Методическое хранилище Ивановского института развития образования].- http://www.iroio.ru/;

Программное обеспечение:

В процессе изучения дисциплины используются офисные пакеты Microsoft Office, Open Office, интерпретатор языка Python 3.5 (или выше) и среды разработки IDLE и PyCharm, программа для просмотра и чтения файлов PDF Adobe Acrobat Reader DC, программа для воспроизведения флэш-анимации в браузерах Adobe Flash Player, вэб-обозреватели Google Chrome, Opera, Антивирус Касперского, 1С университет.

9. Материально-техническое обеспечение дисциплины

1. Учебная аудитория для проведения занятий лекционного типа, укомплектованная специализированной мебелью и техническими средствами обучения, служащими для предоставления учебной информации большой аудитории.
2. Учебная аудитория для проведения занятий семинарского типа, укомплектованная специализированной мебелью и техническими средствами обучения.
3. Учебная аудитория для проведения текущего контроля и промежуточной аттестации, укомплектованная специализированной мебелью и техническими средствами обучения.
4. Помещение для самостоятельной работы, оснащенное специализированной мебелью, компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду организации.
5. Учебно-наглядные пособия для занятий лекционного типа, обеспечивающие тематические иллюстрации: презентации по программированию на языке Python.

Авторы рабочей программы дисциплины: А.В. Ермошин, доцент кафедры математики, информатики и методики обучения, кандидат технических наук, доцент; В.А. Смирнов, преподаватель (аспирант) кафедры математики, информатики и методики обучения.

Программа рассмотрена на заседании кафедры математики, информатики и методики обучения

«____» _____ 20__ г., протокол № ____

Программа обновлена
протокол заседания кафедры № _____ от «_____» _____20 __ г.
Согласовано:
Руководитель ОП _____ С.В. Белов
(подпись)

Программа обновлена
протокол заседания кафедры № _____ от «_____» _____20 __ г.
Согласовано:
Руководитель ОП _____ С.В. Белов
(подпись)

Программа обновлена
протокол заседания кафедры № _____ от «_____» _____20 __ г.
Согласовано:
Руководитель ОП _____ С.В. Белов
(подпись)

<u>Приложение 1.</u> Методические указания для обучающихся по освоению дисциплины

Сопровождение самостоятельной работы студентов по курсу «Программирование» может быть организовано в следующих формах:

– согласование индивидуальных заданий (виды и темы заданий, сроки представления результатов) самостоятельной работы студента в пределах часов, отведенных на самостоятельную работу;

– консультации (индивидуальные и групповые), в том числе с применением дистанционных технологий взаимодействия;

– промежуточный контроль хода выполнения заданий, строящихся на основе различных способов самостоятельной деятельности в открытой информационной среде.

Преподаватель определяет содержание самостоятельной работы, график её выполнения; создает сетевую информационную и коммуникационную среду для организации самостоятельной работы.

Одной из важнейших особенностей заданий для самостоятельной работы по курсу «Программирование» является их ориентация на формирование способности оценивать приоритеты и ограничения при выборе средств информационных технологий для решения управленческих задач, а также на мотивирование самообразовательной деятельности с использованием ИКТ.

К типовым заданиям для самостоятельной работы студентов относятся:

– Конспектирование и реферирование литературных источников по тематике учебного курса.
– Решение задач, связанных с установкой программного обеспечения для написания программ, создание программ на языке Python для решения прикладных задач;
– Изучение и анализ предложенных электронных образовательных ресурсов, содержащих информацию по программированию на языке Python;
– Подготовка к тестированию на занятиях по каждой теме курса;
– Повторение материала с использованием конспектов.

Перечень вопросов для самостоятельного изучения

1. Формальные и фактические параметры в Python.
2. Понятие о локальных и глобальных переменных в Python.
3. Функции в Python.
4. Рекурсивный метод программирования. Разница между «циклическим» и «рекурсивным» способами определения. Явная и косвенная рекурсия.
5. Создание графического приложения с использованием tkinter.
6. Элементы управления в tkinter.
7. Обработка событий в tkinter.
8. Установка сторонних модулей (pip install). Пакетная установка.
9. Функция open. Чтение данных из файла.
10. Основные типы данных NumPy.
11. Функции автозаполнения, создания матриц и числовых диапазонов в NumPy.
12. Свойства и представления массивов NumPy, создание их копий.
13. Индексация, срезы, итерирование массивов NumPy.
14. Базовые математические операции над массивами NumPy.
15. Функция plot для построения и оформления двумерных графиков в Matplotlib.

16. Отображение нескольких координатных осей в одном окне Matplotlib.
17. Граничные значения осей и локаторы для расположения меток на них в Matplotlib.
18. Настройка формата отображения меток у координатных осей в Matplotlib.
19. Отрисовка ступенчатых, стековых, stem и точечных графиков Matplotlib.
20. Отрисовка гистограмм, столбчатых и круговых диаграммы в Matplotlib.

Задания для организации практических занятий:

Практическая работа №1 «Программирование с использованием словарей»

Цель работы: научиться правильно использовать словари при написании программ на языке Python.

Общие сведения: Словари в Python - неупорядоченные коллекции произвольных объектов с доступом по ключу.

Задача: Если записать числа от 1 до 5 английскими словами (one, two, three, four, five), то используется 3 + 3 + 5 + 4 + 4 = 19 букв. Сколько букв понадобится для записи всех чисел от 1 до 1000 (one thousand) включительно? Напишите программу, реализующую данные вычисления.

Пояснение:
Для решения удобно использовать словарь, заданный следующим образом:

```
nums={
     1:'one',
     2:'two',
     3:'three',
     4:'four',
     5:'five',
     6:'six',
     7:'seven',
     8:'eight',
     9:'nine',
     10:'ten',
     11:'eleven',
     12:'twelve',
     13:'thirteen',
     15:'fifteen',
     18:'eighteen',
     20:'twenty',
     30:'thirty',
     40:'forty',
     50:'fifty',
     60:'sixty',
     70:'seventy',
     80:'eighty',
     90:'ninety',
}
```

Практическая работа №2 «Программирование функций»

Цель работы: изучить понятие подпрограмм, рекурсии; научиться правильно создавать и использовать собственные функции при написании программ на языке Python.

Задачи:

1. Даны действительные числа a, b, c, d, e - стороны выпуклого пятиугольника. Найти площадь пятиугольника. (Определить функцию вычисления площади треугольника по его сторонам.)
2. Даны действительные числа x1, y1, x2, y2, …,x10, y10. Найти периметр десятиугольника, вершины которого имеют соответственно координаты (x1, y1), (x2, y2), …, (x10, y10). (Определить функцию вычисления расстояния между двумя точками, заданными своими координатами.)
3. Даны координаты вершин двух треугольников. Определить, какой из них имеет большую площадь. (Определить функцию вычисления площади треугольника по его вершинам.)
4. Написать программу, которая выводит в порядке возрастания все правильные несократимые дроби, знаменатели которых не превосходят N (2<=N<=500).
5. Написать функцию сложения двух чисел, используя только прибавление единицы.
6. Написать функцию умножения двух чисел, используя только операцию сложения.
7. Найти первые N чисел Фибоначчи двумя способами: с помощью рекурсии и с помощью итерации. Сравнить эффективность алгоритмов.
8. Рекурсивно подсчитать количество цифр в заданном числе.

Практическая работа №3 «Работа с файлами»

Цель работы: познакомить с понятием файлового типа данных (бинарные и текстовые); выработать навыки работы с файловым типом данных в языке программирования Python. научиться считывать информацию из файлов, записывать информацию в файл; научится решать задачи с использованием файлов.

Общие сведения: Файл представляет собой структурированный тип данных, содержащий последовательность компонентов одного типа и одной длины. Число элементов в файле (длина файла) не фиксировано. Это является основным отличием файла от списка или массива. Файл можно представить как ленту, у которой есть начало, а конец не фиксирован. Элементы файла записываются на эту ленту последовательно, друг за другом с помощью некоторого устройства - указателя файла. При чтении или записи этот указатель перемещается к следующему элементу и делает его доступным для обработки. В каждый момент доступен для чтения или записи только тот элемент файла, на который установлен указатель.

Задачи:

1. Дан текстовый файл f. Группы символов, разделенные пробелами (одним или несколькими) и не содержащие пробелов внутри себя, будем называть словами. Удалить из файла все однобуквенные слова и лишние пробелы. Результат записать в файл g.

2. Дан файл f, в котором записаны целые числа. Записать в файл g наибольшее значение первых пяти чисел из файла f, затем - следующих пяти и т.д. Если в последней группе окажется менее пяти чисел, то последнее число файла g должно быть равно наибольшему из чисел файла f, образующих последнюю (неполную) группу.
3. Дан файл f, в котором записаны целые числа. Никакое из них не равно нулю. Числа в файле идут в следующем порядке: десять положительных, десять отрицательных, десять положительных, десять отрицательных и т.д. Переписать числа файла f в файл g так, чтобы в файле g числа шли в следующем порядке: а) пять положительных, пять отрицательных, пять положительных, пять отрицательных и т.д.; б) двадцать положительных, двадцать отрицательных, двадцать положительных, двадцать отрицательных и т.д. (предполагается, что количество чисел в файле f делится на 40).
4. Творческое задание. Написать программы для сжатия и распаковки фалов.
5. Задание повышенной сложности. Дан текстовый фал, содержащий программу на языке Python. Проверить эту программу на соответствие числа открывающихся и закрывающихся скобок (любых).

Практическая работа №4 «Основы объектно-ориентированного программирования»

Цель работы: познакомить с понятием "класс"; выработать навыки работы со структурами данных в языке программирования Python. Научится решать задачи на использование структурированных типов данных.

Задачи:

1. Багаж пассажира характеризуется количеством вещей и общим весом вещей. Сведения о багаже каждого пассажира представляют собой класс с двумя полями: одно поле целого типа (количество вещей) и одно - действительное (вес в килограммах).
    a. Найти багаж, средний вес одной вещи в котором отличается не более, чем на 0.3 кг от общего среднего веса одной вещи.
    b. Найти число пассажиров, имеющих более двух вещей и число пассажиров, количество вещей которых превосходит среднее число вещей.
    c. Определить, имеются ли два пассажира, багажи которых совпадают по числу вещей и различаются по весу не более чем на 0,5 кг.
    d. Выяснить, имеется ли пассажир, багаж которого превышает багаж каждого из остальных пассажиров и по числу вещей, и по весу.
    e. Выяснить, имеется ли пассажир, багаж которого состоит из одной вещи весом менее 30 кг.
2. После поступления в ВУЗ о студентах собрана информация: фамилия, нуждается ли в общежитии, стаж, работал ли учителем, что окончил, какой язык изучал. Составить программу, определяющую:
    a. Сколько человек нуждаются в общежитии;
    b. Списки студентов, проработавших 2 и более лет учителем;
    c. Списки окончивших педучилище;
    d. Списки языковых групп.
3. В библиотеке для каждого заказывающего книгу читателя заполняется карточка: фамилия, дата заказа, дата выдачи книги. Определить:
    a. Самый маленький срок, за который нашли книгу;
    b. Сколько заказов было не удовлетворено;
    c. Кто чаще всего берет книги;

   d. Кому выдали книги 15.09.19;
   e. Сколько человек заказывали книги 25.04.19.
4. В школе было три 9 класса, в августе каждый классный руководитель имел сведения о своих учениках: фамилия, куда поступал, поступил или нет. Определить сколько учеников хотели пойти в 10 класс, кто хотел поступать в училище и техникум, кто поступил в училище или техникум, сколько учеников будет учится в 10 классе, сколько необходимо создать 10 классов и по сколько человек.
5. На олимпиаде по информатике на школьников заполнялись анкеты: фамилия, номер школы, класс, занятое место. Напечатать:
   a. Списки школ, занявших призовые места (их количество вводит пользователь);
   b. Какая из школ заняла больше всех призовых мест;
   c. Списки учеников занявших первое место, указать их класс.
6. В деканате хранится информация о зимней сессии на 1 курсе (фамилия, номер группы, оценка 1 по геометрии, оценка 2 по алгебре, оценка 3 по информатике). Составить программу, печатающую фамилии студентов, имеющих задолженность хотя бы по одному предмету, качество успеваемости (т.е. процент студентов, сдавших экзамены на 4 и 5), название предмета, который был сдан лучше всего, номера групп в порядке убывания средней успеваемости их студентов.
7. В отделе кадров студентов хранится следующая информация о каждом студенте: фамилия, имя, отчество, пол, возраст, курс. Составить программу которая печатает номер курса, на котором наибольший процент мужчин, самые распространенные мужские и женские имена, фамилии в алфавитном порядке и инициалы всех студенток, отчество и возраст которых являются одновременно самыми распространенными.
8. В счете указано: название игрушки, стоимость в копейках, возрастные границы. Получить следующие сведения:
   a. Название игрушек, цена которых не превышает 4 руб. и которые подходят детям до пяти лет;
   b. Цену самого дорогого конструктора;
   c. Названия наиболее дорогих игрушек, цена которых отличается от цены самой дорогой игрушки не более, чем на 1 руб.;
   d. Название игрушек, которые подходят как детям 4 лет, так и детям 10 лет;
   e. Можно ли подобрать игрушку, любую, кроме мяча, подходящую ребенку 3 лет, и дополнительно мяч так, чтобы стоимость игрушек не превосходила 5 руб.
9. Хранятся сведения о лесе: вид дерева, общая численность, численность здоровых деревьев. Составить программу вычисления:
   a. Суммарного числа деревьев на контрольном участке;
   b. Суммарного числа здоровых деревьев;
   c. Относительную численность (%) больных деревьев;
   d. Относительную численность (%) различных видов, в том числе больных (%) для каждого вида.
10. Написать программу, осуществляющую приведение подобных членов многочлена. Для хранения данных создать класс одночлена.

Практическая работа №5 «Разработка графического интерфейса»

Цель работы: рассмотрение способов создания классов, объектов и их использования; обучение созданию графического приложения с использованием стандартного средства tkinter.

Ход работы:

Графический пользовательский интерфейс GUI (Graphical User Interface) – это разновидность пользовательского интерфейса, в котором элементы интерфейса (меню, кнопки, значки, списки и т. п.), представленные пользователю на дисплее, исполнены в виде графических изображений. В отличие от интерфейса командной строки (типа старого DOS системы Windows), в GUI пользователь имеет свободный доступ ко всем элементам интерфейса экрана и может ими управлять с помощью клавиатуры или мыши.

Библиотека tkinter (от англ. Tk interface) – это встроенная графическая библиотека на основе средств Tk, широко распространенная в мире GNU/Linux и других UNIX-подобных систем, которая портирована (переведена), в частности, на Microsoft Windows и Apple Mac OS). Tk (от англ. Toolkit – «набор инструментов», «инструментарий») – кроссплатформенная библиотека базовых элементов графического интерфейса, распространяемая с открытыми исходными текстами.

Библиотека tkinter входит в стандартный дистрибутив языка Python (до версии Python 3.0 именовалась Tkinter). В tkinter стандартизированные компоненты графического интерфейса, с которыми взаимодействует пользователь, называются виджетами (widget, от англ. window gadget).

Создание окна

Для подключения модуля tkinter используется форма

```
import tkinter from *
```

позволяющая использовать все имена модуля без указания префикса tkinter.

Tk() является базовым классом любого tkinter приложения. При создании объекта этого класса запускается интерпретатор tcl/tk и создается виджетами наивысшего уровня. Обычно им является основное окно приложения. tkinter является событийно-ориентированной библиотекой. В приложениях такого типа имеется главный цикл обработки событий. В tkinter такой цикл запускается методом mainloop(). Для явного выхода из интерпретатора и завершения цикла обработки событий используется метод quit().

Таким образом минимальное приложение на Tkinter будет таким:

```
import tkinter from *
root = Tk()
root.mainloop()
```

Создание окна, рамки и кнопок

В качестве первого примера создания пользовательского приложения с использоваением графической библиотеки tkinter рассмотрим программу gui1.py, в которой создается пользовательское окно, а в нем – рамка с тремя кнопками:

```
from tkinter import *
class My_frame (Frame):
    ''' Рамка с тремя кнопками '''
    def __init__ (self,master):
        super ().__init__(master)
        self.grid()
        self.create_widgets()
    def create_widgets (self):
```

```
        ''' Создает три кнопки '''
        self.bt1=Button(self, text='Кнопка №1', fg='blue',
                        font='12' )
        self.bt1.grid()
        self.bt2=Button(self)
        self.bt2.grid()
        self.bt2['text']= 'Кнопка №2'
        self.bt2['fg']= 'blue'
        self.bt2['font']= '12'
        self.bt3=Button(self)
        self.bt3.grid()
        self.bt3.config(text='Кнопка №3', fg='blue',
                        font='12' )

root=Tk() #Создание базового окна
root.title( 'Рамка с кнопками' )
root.geometry( '230x95+800+200' )
app=My_frame(root)
root.mainloop()
```

В программе gui1.py подключение модуля tkinter осуществляется с использованием формы from tkinter import *. При этом имена всех классов и методов модуля помещаются в пространство имен программы и обращение к ним происходит без добавления префикса tkinter.

На базе класса Frame модуля tkinter создается пользовательский класс My_frame, который описывает рамку, содержащую три кнопки. При создании конструктора этого класса используется встроенная функция super(), которая предназначена для наследования методов родительского класса при их перегрузке. Таким образом, к методам конструктора класса Frame() объекта master добавляются методы: grid() – для размещения рамки, и create_widgets() – для создания кнопок в рамке.

Кнопки в виде объектов bt1, bt2 и bt3 создаются с использованием класса Button модуля tkinter. Хотя кнопки создаются в этом примере одинаковые, способы задания параметров кнопки: text – указывающего название кнопки, fg и font – задающих соответственно цвет (blue) и размеры (12 пикселей) символов, используются разные:

- для кнопки №1 – непосредственно при ее создании;
- для кнопки №2 – задаются с помощью ключей словаря экземпляра класса Button;
- для кнопки №3 – с помощью метода config().

После описания класса в программе gui1.py:

- создается пользовательское окно, обычно именуемое root;
- указываются имя этого окна – 'Рамка с кнопками' и его размеры и расположение на экране:
  - длина – 230 пикселей;
  - ширина – 85 пикселей;
  - смещение по горизонтали – 800 пикселей;
  - смещение по вертикали – 200 пикселей;
- в этом окне создается рамка, свойства которой описаны в классе My_frame();
- запускается цикл обработки событий (хотя в этом примере события не обрабатываются).

Обработка событий

В качестве примера обработки событий, т.е. действий пользователя над виджетами, приведена программа gui2.py, в которой также, как и в программе gui1.py, создается пользовательский класс My_frame2, описывающий рамку, которая, однако, имеет следующие отличия:

- содержит только одну кнопку
- для этой кнопки кроме параметра text, указывающего ее начальное название, задаются также следующие параметры:
  - command – указывает метод – обработчик события для кнопки (self.count);
  - width – задает длину кнопки (30);
  - bg – задает цвет фона (#efffef');
  - fg – задает цвет символов;(blue)
  - font – задает размер символов (6mm);
- содержит метод count(), который по каждому щелчку по кнопке увеличивает значение переменной bt_clicks на единицу и изменяет в названии кнопки число выполненных по ней щелчков.

```
from tkinter import *
class My_frame2 (Frame):
    '''Рамка с кнопкой, подсчитывающей события '''
    def __init__ (self,master):
        super ().__init__(master)
        self.grid()
        self.bt_clicks=0
        self.create_widgets()
    def create_widgets (self):
        ''' Создает кнопку '''
        self.bt1=Button(self,text= 'Число щелчков: 0' ,
                        command=self.count,
                        width= '20' ,
                        height= '2' ,
                        bg= '#efffef' ,
                        fg= 'blue' ,
                        font= 'arial 14' )
        self.bt1.grid()
    def count (self):
        self.bt_clicks+=1
        self.bt1['text'] = 'Число щелчков: ' \
                           + str (self.bt_clicks)

root=Tk()
root.title( 'Обработка событий' )
root.geometry( '240x60+800+300' )
app2=My_frame2(root)
root.mainloop()
```

Работа с текстом

```
from tkinter import *
class My_frame3 (Frame):
    ''' Рамка для работы с текстом '''
    def __init__ (self,master):
        super ().__init__(master)
        self.grid()
        self.create_widgets()
    def create_widgets(self):
        ''' Создает 2 метки, текстовое поле,
            текстовую область и кнопку '''
        #метка-инструкция
        self.ins_lb=Label(self,
                          text='Как прожить до 100 лет?',
                          fg= 'blue' , font= 'arial 14' )
        self.ins_lb.grid(row=0, column=0,
                         columnspan=2,sticky=W)
        #метка-пароль
        self.ps_lb=Label(self, text= 'Пароль',
                         fg='blue' , font='arial 10' )
        self.ps_lb.grid(row=1, column=0, sticky=W)
        #текстовое поле для ввода пароля
        self.ps_ent=Entry(self, bg='#f0f0ff',
                          font='arial 10' )
        self.ps_ent.grid(row=1, column=1, sticky=E)
        #создание текстовой области, куда будет помещен ответ
        self.txt=Text(self, width=30, height=4, wrap=WORD,
                      bg='#f0f0ff' , font='arial 10' )
        self.txt.grid(row=2, column=0, columnspan=2)
        #кнопка отправки сообщения
        self.bt=Button(self, text='Узнать секрет', fg='blue',
                       bd='5' , font='arial 12',
                       command=self.reveal)
        self.bt.grid(row=3, column=0, sticky=W)
    def reveal(self):
        contents=self.ps_ent.get()
        if (contents== 'secret'):
            message='Чтобы прожить до 100 лет, надо ' \
            'сначала прожить 99 лет и 11 месяцев, а потом ' \
            'вести себя очень-очень осторожно.'
        else:
            message= 'Вы ввели неправильный пароль.'
        self.secret_txt.delete(0.0,END)
        self.secret_txt.insert(0.0,message)

root=Tk()
root.title( 'Работа с текстом' )
root.geometry( ''250x160+850+300' )
app3=My_frame3(root)
```

```
root.mainloop()
```

В программе gui3.py описан класс My_frame3, в котором с помощью метода create_widgets() создаются следующие виджеты:

- неизменяемые сообщения (метки) на базе класса Label:
  - 'Чтобы узнать секрет долголетия введите пароль' (объект ins_lb);
  - 'Пароль' (объект ps_lb);
- текстовое поле для ввода пароля на базе класса Entry (объект ps_ent);
- кнопка с сообщением (надписью) 'Узнать секрет' на базе класса Button (объект submit_bt);
- текстовая область для помещения ответа на базе класса Text (объект secret_txt), которая имеет следующие параметры:
  - ширина (width) – 35 символов;
  - высота (height) – 5 рядков;
  - перенос по границе слева (WORD))
- Помимо метода create_widgets() в классе My_frame3 также описан обработчик щелчка по кнопке 'Узнать секрет' – метод reveal(), который:
  - получает с помощью метода get() из текстового поля ps_ent введенный пользователем пароль;
  - сравнивает его с паролем 'secret' и формирует сообщение, соответствующее результату сравнения: либо приводится "секрет долголения", либо сообщается о том, что пароль введен неправильно;
  - очищает текстовую область ответа (secret_txt);
  - помещает туда сформированное сообщение.

Отметим использование в этой программе метода grid(), который осуществляет размещение (упаковку) графических элементов на рамке. В отличие от программ gui1.py и gui2.py, где метод grid() использовался без параметров и, следовательно, виджеты располагались по усматрению интерпретатора, в программе gui3.py явным образом указано расположение элементов. При использовании метода grid() поле рамки изображается в виде решетки, состоящей из рядков и колонок. Для указания местоположения виджета использованы следующие параметры:

- row – указывает номер рядка;
- column – указывает номер колонки;
- columnspan – указывает размер ячейки в колонках (по умолчанию – 1);
- sticky – указывает выравнивание данных относительно границ ячейки:
  - N – по верхней границе;
  - E – по правой границе;
  - S – по нижней границе;
  - W – по левой границе.

Использование флажков

```
from tkinter import *
class My_frame4 (Frame):
    ''' Использование флажков '''
    def __init__ (self,master):
        super (My_frame4, self).__init__(master)
        self.grid()
```

```
        self.create_widgets()
    def create_widgets (self):
        ''' Создает три флажка '''
        #метка-описание
        Label(self, text='Укажите любимые телепередачи',
              font='arial 13' , fg='blue'
              ).grid(row=0, column=0,sticky=W)
        #метка-инструкция
        Label(self, text='Выберете все, что нравится',
              font='arial 11' , fg='blue'
              ).grid(row=1, column=0,sticky=W)
        #флажок 'Футбол'
        self.likes_football=BooleanVar()
        Checkbutton(self, text='Футбол',
                    variable=self.likes_football,
                    command=self.update_text
                    ).grid(row=2, column=0,sticky=W)
        #флажок 'История'
        self.likes_history=BooleanVar()
        Checkbutton(self, text='История',
                    variable=self.likes_history,
                    command=self.update_text
                    ).grid(row=3, column=0,sticky=W)
        #флажок 'Природа'
        self.likes_nature=BooleanVar()
        Checkbutton(self, text='Природа',
                    variable=self.likes_nature,
                    command=self.update_text
                    ).grid(row=4, column=0,sticky=W)
        #текстовая область с результатом
        self.results_txt=Text(self, fg='blue',
                              font='arial 11', width=40,
                              height=5, wrap=WORD)
        self.results_txt.grid(row=5, column=0, columnspan=3)
    def update_text (self):
        likes= ''
        if self.likes_football.get():
            likes += 'Вам нравится футбол\n'
        if self.likes_history.get():
            likes += 'Вас привлекает история\n'
        if self.likes_nature.get():
            likes += 'Вам небезразлична природа\n'
        self.results_txt.delete(0.0,END)
        self.results_txt.insert(0.0,likes)

root=Tk()
root.title( 'Использование флажков' )
root.geometry( '270x185+800+200' )
app4=My_frame4(root)
```

```
root.mainloop()
```

В отличие от программ gui1.py, gui2.py и gui3.py в программе gui4.py при создании объектов меток (класс Label) используется более короткая форма – без указания имен объектов. Это допускается делать в тех случаях, когда имена этих объектов в дальнейшем не используются в описании класса.

Другой особенностью программы gui4.py является использование флажков – графических элементов класса Checkbutton, при помощи которых осуществляетя выбор (от одной до трех) телепередач:

- о футболе;
- об истории;
- о природе.

При этом для каждого флажка:

- указывается переменная класса BooleanVar, соответственно:
  - likes_football;
  - likes_history;
  - likes_nature.
- создается с использованием короткой формы (т.е. без указания имени) объект класса Checkbutton и задаются следующие значения его параметров:
  - для text – наименование флажка (соответственно, "Футбол", "История" и "Природа");
  - для command – наименование функции (update_text), которая обрабатывает события, связанные с изменением состояния флажка (т.е. выбран флажок или нет);
  - для variable, отражающего текущее состояние флажка, – имя переменной, с которой этот параметр связывается(соответственно likes_football, likes_history или likes_nature).

Обработчик update_text() выполняет следующие действия:

- последовательно анализирует состояние переменных likes_football, likes_history и likes_nature и в зависимости от значения переменной (True или False) добавляет или не добавляет к переменной likes значение, соответственно, 'Вам нравится футбол', 'Вас привлекает история' или 'Вам небезразлична природа';
- очищает текстовое поле results_txt;
- выводит в текстовое поле results_txt значение переменной likes.

При выполнении программы gui4.pyw на экране появляется окно со следующими виджетами:

- меткой-описанием "Укажите любимые телепередачи";
- меткой-инструкцией "Выберете все, что нравится";
- флажками:
  - "Футбол";
  - "История";
  - "Природа";
- текстовой областью для вывода результатов выбора.

Использование переключателей

```
from tkinter import *
class My_frame5 (Frame):
    ''' Использование переключателей '''
```

```
    def __init__ (self,master):
        super (My_frame5, self).__init__(master)
        self.grid()
        self.create_widgets()
    def create_widgets (self):
        ''' Создает переключатель '''
        #метка-инструкция
        Label(self, text='Укажите любимую телепередачу',
              fg= 'blue' , font= 'arial 13'
              ).grid(row=0, column=0,sticky=W)
        self.favorite=StringVar()
        self.favorite.set( None )
        #положение переключателя 'Футбол'
        Radiobutton(self, text='Футбол',
                    variable=self.favorite,
                    value= 'о футболе',
                    command=self.update_text
                    ).grid(row=1, column=0,sticky=W)
        #положение переключателя 'История'
        Radiobutton(self, text= 'История',
                    variable=self.favorite,
                    value= 'об истории',
                    command=self.update_text
                    ).grid(row=2, column=0,sticky=W)
        #положение переключателя 'Природа'
        Radiobutton(self, text= 'Природа',
                    variable=self.favorite,
                    value= 'о природе',
                    command=self.update_text
                    ).grid(row=3, column=0,sticky=W)
        #текстовая область для результата
        self.results_txt=Text(self, fg='blue',
                              font='arial 10', width=40,
                              height=5, wrap=WORD)
        self.results_txt.grid(row=4, column=0, columnspan=3)
    def update_text (self):
        message= "Ваша любимая телепередача - "
        message += self.favorite.get()
        self.results_txt.delete(0.0, END)
        self.results_txt.insert(0.0, message)

root=Tk()
root.title( 'Использование переключателя' )
root.geometry( '310x120+800+200' )
app=My_frame5(root)
root.mainloop()
```

В отличие от программы gui4.py, в которой может осуществляться выбор нескольких телепередач, в программе gui5.py можно выбрать только одну телепередачу. Поэтому вместо

трех флажков класса Checkbutton используется один переключатель с тремя положениями, наименования которых соответствуют наименованиям флажков.

При этом выполняются следующие действия:

- указывается атрибут self.favorite класса StringVar для хранения наименования выбранного положения переключателя (при его создании указывается значение None);
- для каждого положения переключателя создается объект класса Radiobutton и указываются значения его параметров:
    - для text – наименование положения переключателя (соответственно, "Футбол", "История" и "Природа");
    - для command – наименование функции (update_text), которая обрабатывает события, связанные с изменением состояния переключателя;
    - для variable, отражающего текущее состояние переключателя, – self.favorite (имя атрибута, с которыи этот параметр связывается;
    - для value – значение выбранного положения переключателя, соответственно, 'о футболе', 'об истории' или 'о природе'.

Обработчик update_text() выполняет следующие действия:

- задает начальное значение переменной message ("Ваша любимая телепередача - ");
- используя атрибут self.favorite и метод get() получает значение выбранного положения переключателя и добавляет его к значению переменной message;
- очищает текстовое поле results_txt;
- выводит в текстовое поле results_txt значение переменной message.

При выполнении программы IE: gui5.pyw на экране появляется окно со следующими виджетами:

- меткой-инструкцией "Укажите любимую телепередачу";
- переключателем, имеющем три состояния:
    - "Футбол";
    - "История";
    - "Природа";
- текстовой областью для вывода результата выбора.

Индивидуальные задания

Разработать программу на языке Python, в которой на базе модуля tkinter разработан графический интерфейс, имеющий несколько окон:

- окно с кнопкой (объект Button), надпись на которой меняется по каждому щелчку и содержит следующую информацию – номер щелчка и число, которое определяется согласно колонке "Число" табл. №1:
    - 1 – очередное простое число (начиная с 1);
    - 2 – очередное число Фибоначчи (начиная с 1);
    - 3 – очередное простое число вида 2 n - 1 (начиная с n = 1);
- окно, содержащее:
    - текстовое поле (объект Text), куда выводится начало некоторого небольшого сообщения (рассказа и т.п.);
    - поле ввода пароля (объект Entry);
    - кнопку для продолжения вывода сообщения (объект Button);

  - второе текстовое поле (объект Text), куда выводится неожиданное (смешное) завершение сообщения (при правильном вводе пароля) или сообщение о неправильном вводе пароле;
  - метки (объект Label) – при необходимости;
- окно, содержащее:
  - пять флажков (объекты Checkbutton), задающие согласно колонке "Флажки" табл. №1:
    - 1 – числа 1, 2, 4, 8 и 16;
    - 2 – 5 букв кириллицы (на усмотрение студента);
    - 2 – 5 английских букв (на усмотрение студента);
  - текстовое поле, куда помещается результат:
    - для чисел – сумма заданных чисел;
    - для букв – слово, состоящее из выбранных букв;
- окно, содержащее:
  - переключатели (объекты Radiobutton), задающие согласно колонке "Переключатели" табл. №1:
    - 1 – изменяемые типы;
    - 2 – неизменяемые типы;
    - 3 – контейнеры;
    - 4 – последовательности;
  - текстовое поле, куда помещается результат – пример литерала для выбранного класса.

Таблица 1 – Перечень индивидуальных заданий

Номер п/п	**Число**	**Флажки**	**Переключатели**
1	1	3	1
2	2	1	2
3	3	2	3
4	1	3	4
5	2	1	1
6	3	2	2
7	1	3	3
8	2	1	4
9	3	2	1
10	1	3	2
11	2	1	3
12	3	2	4
13	1	3	1
14	2	1	2
15	3	2	3
16	1	3	4
17	2	1	1
18	3	2	2
19	1	3	3
20	2	1	4

Практическая работа №6 «Использование API сайтов»

Задание: Ознакомиться с примерами программ, взаимодействующих с Web-ресурсами. Выполнить задания и представить файлы формата .py в качестве ответа.

Код №1: Сайт ipify.org, позволяющий узнать текущий публичный IP-адрес:

```
# подключение модуля для чтения и открытия URL-адресов
import urllib.request

# создаём объект запроса к URL-адресу с помощью urllib
# в виде словаря: https://api.ipify.org?format=json
req = urllib.request.Request('https://api.ipify.org')

# выполняем запрос
request_url = urllib.request.urlopen(req)

# читаем данные с URL-адреса
ip = request_url.read()

# выводим данные в кодировке UTF-8
print('My public IP address is: ' + ip.decode('UTF-8'))
```

Код №2: Получение информации об IP-адресе:

```
import urllib.request

ip = '8.8.8.8'
req = urllib.request.Request('https://ipapi.co/' + ip + '/json/')

request_url = urllib.request.urlopen(req)
data = request_url.read()
print(data.decode('UTF-8'))
```

Код №3: Создание короткой ссылки:

```
import urllib.parse  # подключение модуля обработки строк с URL
# подключение модуля для чтения и открытия URL-адресов
import urllib.request

# Вводим ссылку для сокращения
line = input('Введите сокращаемый URL:')
# Кодируем URL-адрес (замена специальных символов на escape-символы)
```

```
    url = urllib.parse.quote(line, safe='')
    # Создание запроса
    req = urllib.request.Request('https://clck.ru/--?url=' + url)

    request_url = urllib.request.urlopen(req)
    data = request_url.read()
    print(data.decode('UTF-8'))
```

Код №4: Вывод ответов на поисковый запрос системы Searx (свободная поисковая система, объединяющая результаты других поисковых систем):

```
    import urllib.request
    import json
    import random

    # Набор ресурсов для получения данных
    urls = [
        'https://searx.mistli.net/search?q=',
        'https://etsi.me/search?q=',
        'https://searx.tyil.nl/search?q=',
        'https://searx.bissisoft.com/search?q=',
        'https://searx.ru/search?q=',
        'https://www.simspot.de/search?q='
    ]

    # Получение данных об ответе
    req = urllib.request.Request(
        random.choice(urls) + 'python&format=json&safesearch=1')
    req.add_header('User-Agent',
                   'Mozilla/5.0 (Windows NT 6.1; Win64; '
                   +      'x64;      rv:102.0)      Gecko/20100101
Firefox/102.0')
    req.add_header('Accept',

'text/html,application/xhtml+xml,application/xml;'
                   + 'q=0.9,image/avif,image/webp,*/*;q=0.8')
    req.add_header('Accept-Encoding', 'gzip, deflate, br')
    req.add_header('Accept-Language', 'ru-RU,ru;q=0.8,'
                   + 'en-US;q=0.5,en;q=0.3')

    request_url = urllib.request.urlopen(req)
    data = request_url.read()
    data = data.decode('UTF-8')

    # Перевод данных в формат словаря
    sl = json.loads(data)
```

```
# Вывод ответов и адресов к ним
for answer in sl["results"]:
    print(answer["url"])
    print(answer["title"])
    print('=' * 10)
```

Задания к лабораторной работе:

1. Дописать все вышеперечисленные программы так, чтобы вывод данных осуществлялся в файл, а не на консоль.

2. Переписать код №4 так, чтобы вместо ключевого слова «python» пользователь мог искать любые слова (ввод слов для поиска производить из файла или консоли).

Практическая работа №7 «Многострочное текстовое поле»

1. Настроить для программы собственную цветовую схему (отредактировать параметры `bg` и `fg` у виджетов).

2. Доработать программу так, чтобы в ней присутствовала кнопка «Backup» для быстрого сохранения файла в папке с программой. Имя файла задаётся текущей датой и временем.

3. Добавить на форму надпись (`Label`) с информацией об авторе программы (ФИО, курс, группа).

```
from tkinter import *
import datetime

root = Tk()
root.title("Блокнот")

fr1 = Frame(root)
fr1.pack(fill=BOTH, expand=True)

# Виджет - полоса прокрутки
scrollbar = Scrollbar(fr1)
scrollbar.pack(side=RIGHT, fill=Y)

# Виджет - многострочное текстовое поле
# wrap=WORD - автоматический перенос слов на следующую строчку
# wrap=CHAR - посимвольный перенос на следующую строчку
# wrap=NONE - без переносов (в этом случае желательно
# также подключение полосы прокрутки через xscrollcommand)
# yscrollcommand - установка вертикальной прокрутки
text = Text(fr1, wrap=WORD,
            yscrollcommand=scrollbar.set)
```

```
    text.pack(side=LEFT, fill=BOTH, expand=True)

    scrollbar.config(command=text.yview)

    fr2 = Frame(root)
    fr2.pack(expand=True)

    def create_new():
        # 1.0 – координаты начала фрагмента текста, на который
будет
        # действовать команда. Формат: y.x, где
        # y (в команде: 1) – номер строки, первая строка имеет
номер 1
        # x (в команде: 0) – номер столбца, первый имеет номер 0
        text.delete(1.0, END)

    def open_file():
        # Вызов диалога открытия файла
        # и получение информации об адресе открытого
        # файла в переменную file_name
        file_name = filedialog.askopenfilename()
        # Если выбран файл
        if file_name != '':
            # Открываем файл с помощью стандартной команды
            f = open(file_name)
            # Чтение текста из файл
            s = f.read()
            # Очистка предыдущего текста
            text.delete(1.0, END)
            # Вставка считанного текста
            text.insert(1.0, s)
            # Закрытие файла
            f.close()

    def save_file():
        file_name = filedialog.asksaveasfilename(
            defaultextension='.txt')
        if file_name != '':
            f = open(file_name, 'w')
            s = text.get(1.0, END)
            f.write(s)
            f.close()

    def insert_date():
        paste = str(datetime.datetime.now().year)
        # Получение номера месяца
        # с добавлением нуля слева до двух символов
        paste += str(datetime.datetime.now().month).rjust(2, "0")
        paste += str(datetime.datetime.now().day).rjust(2, "0")
```

```
        paste += " "
        paste += str(datetime.datetime.now().hour).rjust(2, "0")
        paste += str(datetime.datetime.now().minute).rjust(2, "0")
        paste += str(datetime.datetime.now().second).rjust(2, "0")
        # Вставка сформированной строки paste
        # в позицию курсора - text.index("insert")
        text.insert(text.index("insert"), paste)

    btnNew = Button(fr2, text='Новый', command=create_new)
    btnNew.pack(side=LEFT, padx=10, pady=5)

    btnOpen = Button(fr2, text='Открыть', command=open_file)
    btnOpen.pack(side=LEFT, padx=10, pady=5)

    btnSave = Button(fr2, text='Сохранить', command=save_file)
    btnSave.pack(side=LEFT, padx=10, pady=5)

    btnDate = Button(fr2, text='Вставить дату',
command=insert_date)
    btnDate.pack(side=LEFT, padx=10, pady=5)

    root.mainloop()
```

Практическая работа №8 «Работа с графикой в tkinter»

1. Ознакомиться с программой для отрисовки примитивов.

```
    # Подключение модуля для создания окна
    from tkinter import *

    root = Tk()
    root.geometry('500x600+100+100')

    # Создание холста для рисования
    canv = Canvas(root, bg='grey')
    # размещение виджета с помощью «упаковщика»
    # Виджет будет занимать весь размер формы
    # и расширяться при масштабировании
    canv.pack(fill=BOTH, expand=True)

    def keypressed(event):
        # Получение доступа к переменным y, x и rect
        # из основного скрипта
        global y
        global x
        global rect
        # удаление всех нарисованных фигур
        canv.delete("all")
```

```
        # проверка кода нажатой клавиши
        # стрелки: 38 – вверх, 40 – вниз, 37 – влево, 39 - вправо
        if event.keycode == 38:
            y -= 5
        elif event.keycode == 37:
            x -= 5
        elif event.keycode == 40:
            y += 5
        elif event.keycode == 39:
            x += 5
        # Создание прямоугольника
        rect = canv.create_rectangle(x - r, y - r,
                                     x + r, y + r, fill='#FF0000')
        # Создание овала
        #  canv.create_oval(x  -  r,  y  -  r,  x  +  r,  y  +  r,
fill='blue')
        # Создание крестика
        # canv.create_line(x - r, y - r, x + r,
        #                  y + r, fill='red', width=3)
        # canv.create_line(x + r, y - r, x - r,
        #                  y + r, fill='red', width=3)

    x = 200
    y = 300
    r = 100

    rect = canv.create_rectangle(x - r, y - r,
                                 x + r, y + r, fill='#FF0000')
    # canv.create_oval(x - r, y - r, x + r, y + r, fill='blue')
    # canv.create_line(x  -  r,  y  -  r,  x  +  r,  y  +  r,  fill='red',
width=3)
    # canv.create_line(x  +  r,  y  -  r,  x  -  r,  y  +  r,  fill='red',
width=3)

    # Обработка нажатия на кнопку на клавиатуре подпрограммой
keypressed
    root.bind('<Key>', keypressed)

    root.mainloop()
```

2. Доработать программу так, чтобы в ней перемещался смайлик.



Задания для самостоятельной работы студентов:

1. Требуется определить, является ли год, номер которого введен, високосным. (Напомним, что год является високосным, если его номер кратен 4, но не кратен 100, а также если он кратен 400.)

2. Бизнесмен Василий после прочтения известной книги решил открыть новый бизнес – отгружать апельсины бочками. Партнерам важно знать сколько именно бочек апельсинов отгружается каждый день. Мобильный телефон Василия поддерживает только транслит, поэтому он передает сообщения вида "N bochek" . Например, "3 bochki" или "1 bochka". Напишите программу, которая дописывает к числу слово в правильной форме (выбирает из "bochka" , "bochek" , "bochki") в зависимости от N.

3. Помимо составления последовательностей, летом Вася очень любил смотреть в окно. Напротив его дома расположился офис некоторой строительной фирмы. В течение всего месяца Вася наблюдал за его служащими. Про каждый из 31 дня месяца он знает, сколько сотрудников пришло на работу. Ему также известно, что каждый из служащих берет ровно по 4 выходных в месяц. Теперь он ломает голову над загадкой – сколько всего сотрудников работает в этом офисе. Напишите программу, которая ответит Васе на этот вопрос.
Входные данные: Вводится 31 целое неотрицательное число. Эти числа описывают количество работников, пришедших в офис в соответствующие дни месяца. Гарантируется, что входные данные корректны.
Выходные данные: Выведите единственное число – общее количество работников офиса. Гарантируется, что ответ не превышает 100.

4. Дано натуральное k. Напечатать k-ю цифру последовательности 1123581321..., в которой выписаны подряд все числа Фибоначчи.

5. Дано натуральное k. Напечатать k-ю цифру последовательности 149162536..., в которой выписаны подряд квадраты всех натуральных чисел.

6. Дана дата (число, месяц, год и день недели). Определить правильность заданной даты, вычислить последний день данного месяца, сколько дней прошло от 1-го января 1-го года нашей эры, определить день недели, приходящийся на заданную дату, если 1-е января 1-го года нашей эры было понедельником.

7. Написать программу вычисления длин сторон, углов, площади, радиусов вписанной и описанной окружностей для треугольника, заданного координатами вершин на плоскости в декартовых или полярных координатах.

8. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке:

- все гласные буквы, которые входят в каждое слово;
- все согласные буквы, которые не входят ни в одно слово;
- все звонкие согласные буквы, которые входят хотя бы в одно слово;
- все глухие согласные буквы, которые не входят хотя бы в одно слово;
- все согласные буквы, которые входят только в одно слово;
- все глухие согласные буквы, которые не входят только в одно слово;
- все звонкие согласные буквы, которые входят более чем в одно слово;
- все гласные буквы, которые не входят более чем в одно слово;

9. Подсчитать общее количество цифр и знаков '+', '-', и '*', входящих в строку s.

10. Написать программу раздачи карт при игре в дурака, количество игроков задается с клавиатуры.

11. Составить программу случайного выбора дежурного из списка, в котором 4 мальчика и 4 девочки, причем для девочек вероятность выбора в два раза ниже, чем для мальчиков.

12. Необходимо составить программу проверки равенства треугольников по их элементам. Пользователь поочерёдно вводит данные об углах и сторонах треугольников, а программа сохраняет введённые данные или выдаёт сообщение о том, что такой же треугольник уже был введён. Предусмотреть защиту от неверного ввода данных.

13. Аналогично задаче 12, сравнивать трапеции.

14. Составить программу перемещения по экрану вводимой предварительно с клавиатуры текстовой строки. Перемещение осуществляется по восьми направлениям, направление должно меняться через определённое количество шагов. Причем при смене направления движения должен меняться цвет надписи.

15. Создать окно с тестом по информатике. Тест должен включать как минимум один вопрос с выбором варианта ответа и вопрос с числовым ответом. Реализовать проверку правильности ответов при нажатии на кнопку «ОК».

16. Создать оконное приложение с кроссвордом по информатике. Кроссворд должен включать как минимум четыре слова. Ввод слова осуществляется по одной букве в каждое текстовое поле. Проверка правильности и корректности (ровно одна буква в поле) – при нажатии на кнопку «ОК». Тексты вопросов размещать в надписях (Label).

17. Дан список, состоящий из целых чисел. Напишите программу, которая определяет, есть ли в списке пара соседних элементов с одинаковыми знаками.

18. Дан список, состоящий из целых чисел. Известно, что числа упорядочены по неубыванию (то есть каждый следующий элемент не меньше предыдущего). Напишите программу, которая определит количество различных чисел в этом списке.

19. Задана строка S (длина не превышает 255 символов). Удалите из нее все цифры 1 и 2 и выведите результат.

20. Напишите программу, которая циклически сдвигает элементы списка вправо (например, если элементы нумеруются, начиная с нуля, то 0-й элемент становится 1-м, 1-й становится 2-м, ..., последний становится 0-м, то есть список {3, 5, 7, 9} превращается в список {9, 3, 5, 7}).

21. Дана последовательность из N ($1 \le N \le 100000$) целых чисел и число K ($|K| \le 100000$). Сдвинуть всю последовательность (сдвиг - циклический) на |K| элементов вправо, если K – положительное и влево, если отрицательное. В данной задаче нельзя использовать дополнительные массивы (списки). Обратите внимание, что нужно именно преобразовать имеющийся список и распечатать его целиком, а не создать новый, даже назвав его тем же самым именем (это возможно в языке Python).

22. В заданном квадратном массиве, содержащем целые числа, обнулить только те строки и столбцы, в которых встречаются нули.

23. Решить задачу: Пользователь вводит число N, а затем N строк, каждая из которых состоит из N целых чисел (т.е. задана матрица размера N x N). Изменить матрицу так, чтобы:
a. Все четные элементы матрицы были возведены в квадрат.
b. Все элементы матрицы, кратные трём, были умножены на 10.
c. Все отрицательные элементы матрицы были умножены на 2.
d. Все положительные элементы матрицы были заменены их остатком от деления на 2022.
Итоговую матрицу вывести на экран в том же формате.

24. Решить задачу: Пользователь вводит число N. Создать матрицу размера N x N и заполнить случайными целыми числами в диапазоне от 0 до 100. Вывести на экран полученную матрицу и:
a. Произведение элементов в каждой строке.
b. Произведение элементов в каждом столбце.
c. Сумму всех чётных элементов в каждом столбце.
d. Сумму всех элементов, кратных 10, в каждой строке.

25. Когда Антон прочитал «Войну и мир», ему стало интересно, сколько слов и в каком количестве используется в этой книге.
Помогите Антону написать упрощённую версию такой программы, которая сможет подсчитать слова, разделённые пробелом и вывести получившуюся статистику.
Программа должна считывать одну строку со стандартного ввода и выводить для каждого уникального слова в этой строке число его повторений (без учёта регистра) в формате "слово количество".
Порядок вывода слов может быть произвольным, каждое уникальное слово должно выводиться только один раз.

26. Посчитать количество букв в строке и вывести на экран ту букву, которая попадается в ней максимальное количество раз. Если таких букв несколько, то вывести меньшую в лексикографическом (алфавитном) порядке.

27. Пользователь вводит число N и текст, состоящий из N строчек. Вывести все слова, которые входят в данный текст только один раз.

28. Напишите программу, которая принимает текст и выводит два слова: наиболее часто встречающееся и самое длинное.

Критерии оценки:
Всего баллов – 56 (Максимально)
По 2 балла за каждое правильное самостоятельно выполненное задание.