Контур Школа

АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ
ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«УЧЕБНЫЙ ЦЕНТР СКБ КОНТУР»

Утверждаю
Директор АНО ДПО
«Учебный центр СКБ Контур»

_____ Т.В. Рубан

1 сентября 2023 г.

ДОПОЛНИТЕЛЬНАЯ ПРОФЕССИОНАЛЬНАЯ ПРОГРАММА
повышения квалификации

ЯЗЫК ПРОГРАММИРОВАНИ PYTHON

(профстандарт «Программист», код А)

Москва, 2023 г.

ОГЛАВЛЕНИЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Настоящая программа (далее — Программа) представляет собой совокупность требований, обязательных при реализации программы дополнительного профессионального образования повышения квалификации по теме «Язык программирования Python».

Настоящая программа разработана на основании федеральных требований к программам переподготовки и повышения квалификации специалистов специалистами Автономной некоммерческой организации дополнительного профессионального образования «Учебный центр СКБ Контур» (далее — АНО ДПО «Учебный центр СКБ Контур»).

Программа разработана в соответствии с:

— профессиональным стандартом «Программист» (код А), утвержденным приказом Министерства труда и социальной защиты РФ от 20.07.2022 г. № 424Н.

— Федеральным государственным образовательным стандартом среднего профессионального образования по специальности 09.02.07 «Информационные системы и программирование» (утв. приказом Министерства образования и науки РФ от 9 декабря 2016 г. № 1547)

Право на реализацию дополнительной образовательной программы повышения квалификации по теме «Повышение квалификации «Язык программирования Python», разработанной на основании федеральных стандартов, имеет образовательный центр при наличии соответствующей лицензии.

Цели:

- формирование знаний по разработке программного обеспечения (далее - ПО) с использованием языка программирования Python;
- формирование навыков разработки и отладки программного кода с использованием языка программирования Python.

Категория слушателей:

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

Организационно-педагогические условия:

Образовательный процесс осуществляется на основании учебного плана и регламентируется расписанием занятий для каждой учебной группы.

Срок обучения: 54/3/1 (час., нед., мес.).

Режим занятии: 28 академических часов самостоятельного обучения, 26 академических часов – работа на образовательной онлайн-платформе.

Форма обучения: заочная с использованием дистанционных образовательных технологий, электронного обучения.

Возраст слушателей: 18 лет и старше.

Характеристика профессиональной деятельности слушателей

Область профессиональной деятельности слушателей: разработка компьютерного программного обеспечения.

Слушатель готовится к следующим видам деятельности:

— в соответствии с ФГОС СПО и требованиями профессионального стандарта «Программист» (код А), утвержденного Приказом Министерства труда и социальной защиты РФ от 20.07.2022г. № 424Н.

Требования к результатам освоения дополнительной профессиональной образовательной программы

Специалист должен обладать общепрофессиональными компетенциями, включающими в себя способность:

— Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам;

— Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности;

— Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по финансовой грамотности в различных жизненных ситуациях;

— Эффективно взаимодействовать и работать в коллективе и команде;

— Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста;

— Пользоваться профессиональной документацией на государственном и иностранном языках.

Специалист должен обладать профессиональными компетенциями, соответствующими основным видам профессиональной деятельности:

— Формализация и алгоритмизация поставленных задач для разработки программного кода;

— Написание программного кода с использованием языков программирования;

— Оформление программного кода в соответствии с установленными требованиями;

— Проверка и отладка программного кода.

Для реализации программы задействован следующий кадровый потенциал:

- **Преподаватели учебных дисциплин** — обеспечивается необходимый уровень компетенции преподавательского состава, включающий высшее образование в области соответствующей дисциплины программы или высшее образование в иной области и стаж преподавания по изучаемой тематике не менее трех лет; использование при изучении дисциплин программы эффективных методик преподавания, предполагающих выполнение слушателями практических заданий.
- **Административный персонал** — обеспечивает условия для эффективной работы педагогического коллектива, осуществляет контроль и текущую организационную работу.

- **Информационно-технологический персонал** — обеспечивает функционирование информационной структуры (включая ремонт техники, оборудования, макетов иного технического обеспечения образовательного процесса, поддержание сайта Контур.Школы и т.п.).

Содержание программы повышения квалификации определяется учебным планом и календарным учебным графиком программы дисциплин (модулей), требованиями к итоговой аттестации и требованиями к уровню подготовки лиц, успешно освоивших Программу.

Текущий контроль знаний проводится в форме наблюдения за работой слушателей и контроля их активности на образовательной платформе, проверочного тестирования.

Промежуточный контроль знаний, полученных слушателями посредством самостоятельного обучения (освоения части образовательной программы), проводится в виде тестирования.

Итоговая аттестация по Программе проводится в форме тестирования.

Слушатель допускается к итоговой аттестации после самостоятельного изучения дисциплин Программы в объеме, предусмотренном для обязательных самостоятельных занятий и подтвердивший самостоятельное изучение сдачей поурочных тестов.

Слушатели, освоившие Программу и успешно прошедшие итоговую аттестацию, получают удостоверение о повышении квалификации.

Оценочными материалами по Программе являются блоки контрольных вопросов по дисциплинам, формируемые образовательной организацией и используемые при текущем контроле знаний (тестировании) и итоговой аттестации.

Методическими материалами к Программе являются нормативные правовые акты, положения которых изучаются при освоении дисциплин Программы. Перечень методических материалов приводится в рабочей программе образовательной организации.

УЧЕБНЫЙ ПЛАН

ПО
ДОПОЛНИТЕЛЬНОЙ ПРОФЕССИОНАЛЬНОЙ ПРОГРАММЕ
повышения квалификации

ЯЗЫК ПРОГРАММИРОВАНИ PYTHON

(профстандарт «Программист», код А)

№ п/п	Наименование разделов, дисциплин	Всего часов	В том числе		Форма контроля
			Самостоятельная работа	Работа на образовательной онлайн-платформе	
1	Базовый Python	31	16	15	Зачет
2	Продвинутый Python	21	12	9	Зачет
ИТОГОВАЯ АТТЕСТАЦИЯ		2	–	2	Зачет
Всего:		54	28	26	–

УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН
ПО
ДОПОЛНИТЕЛЬНОЙ ПРОФЕССИОНАЛЬНОЙ ПРОГРАММЕ
повышения квалификации

ЯЗЫК ПРОГРАММИРОВАНИ PYTHON

(профстандарт «Программист», код А)

№ п/п	Наименование разделов, дисциплин	Всего часов	В том числе		Форма контроля
			Самостоятельная работа	Работа на образовательной онлайн-платформе	
1	**Базовый Python**	**31**	**16**	**15**	**Зачет**
1.1	Переменные и типы данных	4	2	2	Тестирование
1.2	Условный оператор и отладка	4	2	2	Тестирование
1.3	Циклы	4	2	2	Тестирование
1.4	Строки, регулярные выражения, обработка ошибок	4	2	2	Тестирование
1.5	Коллекции	3	2	1	Тестирование
1.6	Функции	4	2	2	Тестирование
1.7	Итераторы, генераторы, анонимные функции	4	2	2	Тестирование
1.8	Файловая система	4	2	2	Тестирование
2	**Продвинутый Python**	**21**	**12**	**9**	**Зачет**
2.1	Декораторы и перегрузка	3	2	1	Тестирование
2.2	Введение в ООП	4	2	2	Тестирование
2.3	Инкапсуляция и полиморфизм	4	2	2	Тестирование
2.4	Наследование и абстрактные классы	3	2	1	Тестирование
2.5	Дескрипторы	3	2	1	Тестирование
2.6	Дата классы и Метаклассы	4	2	2	Тестирование
ИТОГОВАЯ АТТЕСТАЦИЯ		**2**	**–**	**2**	**Зачет**
Всего:		**54**	**28**	**26**	**–**

КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Календарный график обучения является примерным, составляется и утверждается для каждой группы.

Срок освоения программы — 3 недели. Начало обучения — по мере набора группы. Примерный режим занятий: не более 8 академических часов в день, до 40 часов в неделю. Промежуточная и итоговые аттестации проводятся согласно графику.

№	Темы / дни	ВР	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1.	Базовый Python	РП	3	3	3	3	3																
		СР						3	3	3	3	2	2										
2.	Продвинутый Python	РП												3	3	3							
		СР															2	2	2	2	2	2	
Итоговая аттестация		РП																					2

АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ
ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«УЧЕБНЫЙ ЦЕНТР СКБ КОНТУР»

Утверждаю
Директор АНО ДПО
«Учебный центр СКБ Контур»
_____Т.В. Рубан
1 сентября 2023 г.

Рабочая программа учебной дисциплины
«Базовый Python»

образовательной программы дополнительного профессионального образования
повышения квалификации

ЯЗЫК ПРОГРАММИРОВАНИ PYTHON

(профстандарт «Программист», код А)

Москва, 2023 г.

Цель: овладение знаниями по основам языка Python на базовом уровне.

Задачи:

- Понимать основы работы с разными типами данных языка программирования Python на базовом уровне.
- Применять в работе знания по основным типам и структурам данных используемым в языке Python на базовом уровне.

Место дисциплины в структуре программы

Дисциплина позволяет слушателям получить знания по основам языка Python на базовом уровне для выполнения разных задач при разработке программного обеспечения с учетом требований профессионального стандарта «Программист» (код А), утвержденным приказом Министерства труда и социальной защиты РФ от 20.07.2022г. №424Н.

Требования к результатам освоения дисциплины

В результате обучения дисциплине слушатели должны:

Знать:

- Типы данных в языке Python.
- Структуры данных в языке Python.
- Особенности работы с разными типами данных, используемых в языке Python.
- Основы использования условных операторов.
- Циклы в языке Python
- Основные операции со словарем**,** с коллекцией.
- Операции с функциями в Python.
- Основные методы строк.

Уметь:

- Разрабатывать программы с использованием основных конструкций языка Python.
- Использовать правильные имена переменных и констант чтобы улучшить читаемость кода.
- Работать с разными типами и структурами данных при разработке программ на языке Python.
- Работать с отладчиком в Pycharm.
- Получать доступ к словарям, управлять ими.
- Разрабатывать программы используя словари.
- Использовать коллекции данных при разработке ПО.
- Работать с функциями, в том числе с анонимными функциями.
- Выполнять работу с работу с текстовыми, json файлами и табличными данными.

Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 31 академический час (из них самостоятельная работа — 16 ак. часов, работа на образовательной онлайн-платформе — 15 ак. часов).

№ п/п	Наименование разделов и дисциплин	Всего часов	В том числе		Форма контроля
			Самостоятельная работа	Работа на образовательной онлайн-платформе	
1	**Базовый Python**	**31**	**16**	**15**	**Зачет**
1.1	Переменные и типы данных	4	2	2	Тестирование
1.2	Условный оператор и отладка	4	2	2	Тестирование
1.3	Циклы	4	2	2	Тестирование
1.4	Строки, регулярные выражения, обработка ошибок	4	2	2	Тестирование
1.5	Коллекции	3	2	1	Тестирование
1.6	Функции	4	2	2	Тестирование
1.7	Итераторы, генераторы, анонимные функции	4	2	2	Тестирование
1.8	Файловая система	4	2	2	Тестирование

Урок 1.1. Переменные и типы данных

— Понятие переменной, объекта и класса

— Знакомство с базовыми функциями

— Основные арифметические операции. Явные и неявные преобразования

— Форматированный ввод и вывод

— Практическое задание: напишите программу, которая будет запрашивать у пользователя ввод трехзначного числа и выводить на экран следующую информацию:

1. Первую цифру числа.
2. Вторую цифру числа.
3. Третью цифру числа.
4. Сумму всех цифр.

Вывод должен быть форматированным.

Сохраните это практическое задание в одном пространстве или файле. Полностью готовое задание необходимо сдать в уроке 4.

Урок 1.2. Условный оператор и отладка

— Условный оператор if-elif-else

— Операторы сравнения

— Примеры использования условных операторов

— Работа с отладчиком в Pycharm
— Практическое задание:

1. Дано целое число в диапазоне от 0 до 999. Напишите программу, которая определяет, из какого количества разрядов состоит это число. Измените диапазон от −999 до 999 и научите программу правильно определять разрядность отрицательных чисел.
2. Дано целое положительное число из трех чисел, не равных друг другу. Напишите программу, которая выводит «Да», если числа увеличиваются слева направо, и «Нет», если это не так.
3. Дано целое положительное четырехразрядное число. Число может быть палиндромом, то есть читаться одинаково слева направо и справа налево. Напишите программу, которая печатает «Да», если число является палиндромом, и «Нет», если не является.

Сохраните это практическое задание в одном пространстве или файле. Полностью готовое задание необходимо сдать в уроке 4.

Урок 1.3. Циклы

— Цикл while и бесконечные циклы
— Примеры использования цикла while
— Цикл for и интервалы
— Особенности использования циклов for
— Работа с отладчиком в Pycharm
— Практическое задание: задачи на цикл for и на цикл while.

Урок 1.4. Строки, регулярные выражения, обработка ошибок

— Особенности класса <str>
— Основные методы строк
— Регулярные выражения и для чего они нужны
— Операторы регулярных выражений
— Обработка ошибок
— Практическое задание: задача на строки и на регулярные выражения

Урок 1.5. Коллекции

— Списки как универсальный контейнер
— Методы списков
— Особенности работы со списками
— Генераторы списков
— Кортежи и множества
— Работа со словарями
— Практическое задание: задачи на списки и задачи на словари

Урок 1.6. Функции

— Создание и вызов функций
— Параметры и область видимости
— Возврат значений и функций
— Работа с рекурсивными функциями

— Практическое задание:

1. Напишите функцию, которая возводит число N в степень M. Сделайте значение степени параметром по умолчанию равным 2 (если степень не указываем, то число возводится в квадрат). Решите её 2мя способами - обычным и с помощью рекурсии.
2. Напишите функцию, которая при каждом вызове будет выводить на экран количество раз, которое она вызвалась ранее. (Подразумевается количество вызовов за 1 запуск программы).
3. Напишите функцию, которая принимает в себя неограниченное количество параметров, ищет среди них только продукты (задаются отдельным списком заранее) и выводит их на экран.

Урок 1.7. Итераторы, генераторы, анонимные функции

— Понятие итератора и итерируемого объекта
— Понятие генератора
— Генерато-функции
— Анонимные функции
— Работа с анонимными функциями
— Практическое задание: задачи на итераторы, генераторы и на лямбда-функции

Урок 1.8. Файловая система

— Работа с текстовыми и json файлами
— Работа с табличными данными
— Модуль CSV
— Модули OS и SYS для взаимодействия с OS
— Создание своих библиотек
— Практическое задание: создайте отдельный модуль из функций.

АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ
ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«УЧЕБНЫЙ ЦЕНТР СКБ КОНТУР»

Утверждаю
Директор АНО ДПО
«Учебный центр СКБ Контур»

_____ Т.В. Рубан

1 сентября 2023 г.

Рабочая программа учебной дисциплины «Продвинутый Python»

образовательной программы дополнительного профессионального образования

повышения квалификации

ЯЗЫК ПРОГРАММИРОВАНИ PYTHON

(профстандарт «Программист», код А)

Москва, 2023 г.

Цель: применение знаний по основным конструкциям языка Python на продвинутом уровне.

Задачи:

- Понимать основы работы с основными конструкциями языка Python.
- Понимать и применять конструкции языка Python при разработке ПО.

Место дисциплины в структуре программы

Дисциплина позволяет слушателям получить знания по основным конструкциям языка Python для выполнения разных задач при разработке программного обеспечения с учетом требований профессионального стандарта «Программист» (код А), утвержденным приказом Министерства труда и социальной защиты РФ от 20.07.2022г. №424Н.

Требования к результатам освоения дисциплины

В результате обучения дисциплине слушатели должны:

Знать:

- Основы работы с декораторами, передача параметров
- Методы и свойства классов и объектов
- Особенности работы с классами, дата и метаклассами
- Создание классов и их объектов
- Организация структуры классов
- Создание иерархии классов с помощью наследования
- Создание и применение дескрипторов, хранение данных в экземплярах дескриптора
- Назначение метаклассов и необходимость их использования

Уметь:

- Разрабатывать программы с использованием основных конструкций языка Python
- Выполнять работу с декораторами
- Работать с классами, дата и метаклассами
- Использовать дискрипторы при создании классов
- Использовать метаклассы для постройки сложной логики взаимодействия объектов

Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 21 академический час (из них самостоятельная работа — 12 ак. часов, работа на образовательной онлайн-платформе — 9 ак. часов).

№ п/п	Наименование разделов и дисциплин	Всего часов	В том числе		Форма контроля
			Самостоятельная работа	Работа на образовательной онлайн-платформе	
2	**Продвинутый Python**	**21**	**12**	**9**	**Зачет**
2.1	Декораторы и перегрузка	3	2	1	Тестирование
2.2	Введение в ООП	4	2	2	Тестирование
2.3	Инкапсуляция и полиморфизм	4	2	2	Тестирование
2.4	Наследование и абстрактные классы	3	2	1	Тестирование
2.5	Дескрипторы	3	2	1	Тестирование
2.6	Дата классы и Метаклассы	4	2	2	Тестирование

Урок 2.1. Декораторы и перегрузка

— Понятие декоратора. Области применения

— Работа с декораторами. Передача параметров

— Понятие перегрузки. Перегрузка функций

— Практическая работа: создание и модернизация декоратора по заданным условиям.

Урок 2.2. Введение в ООП

— Понятие класса и объект класса

— Создание классов и объектов

— Функции классов. Методы объектов

— Конструктор класса.

— Инициализация объектов

— Работа с классами. Разбор примера

— Практическая работа: реализуйте класс по заданным параметрам, который позволит хранить деньги на счете.

Урок 2.3. Инкапсуляция и полиморфизм

— Приватные методы и свойства

— Свойства property, геттеры и сеттеры

— Статические методы @staticmethod и методы класса @classmethod

— Полиморфизм и перегрузка операторов

— Практическая работа. Создайте класс Fraction, который будет хранить в себе два целых числа - числитель и знаменатель дроби.

Урок 2.4. Наследование и абстрактные классы

— Организация наследования.

— Родительские и дочерние классы

— Примеры и польза наследования
— Переопределение родительских методов. super-классы
— Множественное наследование и миксины
— Абстрактные классы
— Практическая работа:

1. Разработать базовый класс Commodity, описывающий товар.
2. Разработать абстрактный базовый класс AbstaractPurchase, описывающий покупку товара.
3. Разработать первый производный класс от AbstaractPurchase, в котором продажа товара осуществляется со скидкой от цены. Переопределить нужные методы.
4. Разработать второй производный класс от AbstaractPurchase, в котором скидка присутствует, если кол-во единиц товара не меньше некоторого числа, заданного константой в классе. Переопределить нужные методы.
5. Разработать третий производный класс от AbstaractPurchase, в котором к стоимости товара добавляются дополнительные транспортные расходы. Переопределить и добавить нужные методы.
6. Протестировать созданные классы, создав их объекты

Урок 2.5. Дескрипторы

— Понятие и использование дескрипторов
— Дескрипторы данных. Non-data дескрипторы
— Слабые ссылки weakref
— Проблемы хранения данных в дескрипторах
— Практическая работа.

Создайте класс **Car** в котором реализуйте свойства в виде объектов дескрипторов. Реализуйте метод Move(km), который будет увеличивать значение пробега (mileage) и уменьшать кол-во топлива(fuel) в зависимости от параметра km. Машина не должна двигаться дальше, чем есть топлива в баке.
Создайте несколько автомобилей, параметры для которых будут генерироваться случайным образом. И напишите функцию, которая будет определять какой автомобиль проедет дальше всех при текущих параметрах

Урок 2.6. Дата классы и Метаклассы

— Способы создания классов
— Работа метаклассов
— Датаклассы и их назначения
— Создание и использование датаклассов
— Практическая работа.

1 задание. Реализуйте метакласс, который позволит создать только один экземпляр пользовательского класса.

2 задание. Реализуйте класс данных, с помощью которого можно будет реализовать хранение почтовой переписки людей.

ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ ПРОГРАММЫ

Формы аттестации

Для проведения промежуточной и итоговой аттестации программы разработан фонд оценочных средств по программе, являющийся неотъемлемой частью учебно-методического комплекса.

Объектами оценивания выступают:

- степень освоения теоретических знаний;
- уровень овладения практическими умениями и навыками по всем видам учебной работы, активность на занятиях.

Текущий контроль знаний слушателей проводится преподавателем, ведущим занятия в учебной группе, на протяжении всего обучения по программе.

Текущий контроль знаний включает в себя наблюдение преподавателя за учебной работой слушателей и проверку качества знаний, умений и навыков, которыми они овладели на определенном этапе обучения посредством выполнения упражнений на практических занятиях и в иных формах, установленных преподавателем.

Промежуточная аттестация — оценка качества усвоения слушателями содержания учебных блоков непосредственно по завершении их освоения, проводимая в форме зачета посредством тестирования или в иных формах, в соответствии с учебным планом и учебно-тематическим планом.

Итоговая аттестация — процедура, проводимая с целью установления уровня знаний, слушателей с учетом прогнозируемых результатов обучения и требований к результатам освоения образовательной программы. Итоговая аттестация слушателей осуществляется в форме зачета посредством тестирования.

Слушатель допускается к итоговой аттестации после изучения тем образовательной программы в объеме, предусмотренном для лекционных и практических занятий.

Слушателям, освоившим образовательную программу повышения квалификации «Язык программирования Python» (профстандарт «Программист», код А), и успешно прошедшим итоговую аттестацию, выдается удостоверение о повышении квалификации установленного образца с указанием названия программы, календарного периода обучения, длительности обучения в академических часах.

Для аттестации слушателей на соответствие их персональных достижений требованиям соответствующей ОП созданы фонды оценочных средств, включающие типовые задания, тесты и методы контроля, позволяющие оценить знания, умения и уровень приобретенных компетенций.

Фонды оценочных средств соответствуют целям и задачам программы подготовки специалиста, учебному плану и обеспечивают оценку качества общепрофессиональных и профессиональных компетенций, приобретаемых слушателями.

Критерии оценки слушателей

Предмет оценивания (компетенции)	**Объект оценивания (навыки)**	**Показатель оценки (знания, умения)**
Специалист должен обладать общими компетенциями, включающими в себя способность: – Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам. – Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности. – Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по финансовой грамотности в различных жизненных ситуациях. – Эффективно взаимодействовать и работать в коллективе и команде. – Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста.	***Специалист должен обладать профессиональными компетенциями, соответствующими основным видам профессиональной деятельности:*** – Формализация и алгоритмизация поставленных задач для разработки программного кода; – Написание программного кода с использованием языков программирования; – Оформление программного кода в соответствии с установленными требованиями; – Проверка и отладка программного кода.	**Знания:** – Типы и структуры данных в языке Python. – Основные конструкции языка Python. – Функции, классы и объекты в языке Python. – Использование словарей и коллекций в Python. – Модификаторы доступа, свойства и дескрипторы. – Принципы объектно-ориентированного программирования. – Рефакторинг кода и создание функций. – Написание библиотеки с функциями по управлению файлами в ОС. – Создание перегруженных функций. – Построение иерархии классов с применением дескрипторов. – Создание датаклассов и их эффективное использование. **Умения:** – Использовать библиотеки языка для повышения производительности кода – Создавать код с обработкой исключений, чтобы программы работали быстро и не падали – Работать в парадигме объектно-ориентированного программирования – Писать программы на языке Python используя различные типы данных. – Читать код, созданный на языке

Предмет оценивания (компетенции)	Объект оценивания (навыки)	Показатель оценки (знания, умения)
– Пользоваться профессиональной документацией на государственном и иностранном языках.		программирования Python. – Создавать и использовать собственные классы и методы, хранить состояние объектов в атрибутах. – Использовать функции при разработке ПО. – Описывать публичные и внутренние интерфейсы. – Использовать свойства и дескрипторы для доступа к атрибутам объектов. – Получать доступ к словарям, управлять ими. Разрабатывать программы используя словари. – Использовать коллекции данных при разработке ПО. – Использовать метаклассы для постройки сложной логики взаимодействия объектов. – Использовать полученные знания в практической работе. – Владеть навыками профессионально и эффективно применять на практике приобретенные в процессе обучения знания и умения.

Оценка качества освоения учебных модулей проводится в процессе промежуточной аттестации в форме зачета.

Оценка	Критерии оценки
Зачтено	Оценка **«Зачтено»** выставляется слушателю, если он твердо знает материал курса, грамотно и по существу использует его, не допуская существенных неточностей в ответе на тестовые вопросы, правильно применяет теоретические положения при решении практических вопросов, владеет необходимыми навыками и приемами их выполнения. Не менее 70% правильных ответов при решении тестов
Не зачтено	Оценка **«Не зачтено»** выставляется слушателю, который не знает значительной части программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями решает практические вопросы или не справляется с ними самостоятельно. Менее 70% правильных ответов при решении тестов

Оценка качества освоения учебной программы проводится в процессе итоговой аттестации в форме тестирования.

Оценка (стандартная)	**Требования к знаниям**
Зачтено	Оценка **«Зачтено»** выставляется слушателю, продемонстрировавшему твердое и всесторонние знания материалы, умение применять полученные в рамках занятий практические навыки и умения. Достижения за период обучения и результаты текущей аттестации демонстрировали отличный уровень знаний и умений слушателя. Не менее 70% правильных ответов при решении тестов.
Не зачтено	Оценка **«Не зачтено»** выставляется слушателю, который в недостаточной мере овладел теоретическим материалом по дисциплине, допустил ряд грубых ошибок при выполнении практических заданий, а также не выполнил требований, предъявляемых к промежуточной аттестации. Достижения за период обучения и результаты текущей аттестации демонстрировали неудовлетворительный уровень знаний и умений слушателя. Менее 70% правильных ответов при решении тестов

Фонд оценочных средств

Тест к уроку 1.1

1. Данные какого типа возвращает функция input()?
a. Целого
b. Вещественного
c. Строку
d. Булев тип
2. С помощью какого оператора можно изменить порядок вычисления в Python?
a. Скобки
b. Умножение
c. Возведение в степень
d. Остаток от деления
3. Какой параметр надо изменить, чтобы функция print() вывела переданные ей аргументы в столбик?
a. end
b. sep
c. arg
d. next
4. Чтобы результат вычисления не сильно исказился, какой функцией лучше всего преобразовывать вещественное число в целое?
a. floor()
b. ceil()
c. int()
d. round()
5. Каким образом можно получить значение типа int после выполнения операции деления?
a. Делить только целые числа
b. Использовать целочисленное деление
c. Взять модуль получившегося числа
d. Такое невозможно

Тест к уроку 1.2

1. Сколько условий можно проверить внутри одной условной конструкции?
a. 1
b. 3
c. Любое количество
d. 4
2. В чем важное отличие операторов = и ==?
a. = присваивает значения, == сравнивает
b. = — менее точный, == — более точный
c. Отличий нет, это один и тот же оператор
d. В количестве черточек
3. Сколько значений содержит булев тип?
a. 1
b. 2
c. 3
d. Бесконечное количество
4. Что делает оператор pass?
a. Пропускает выполнение текущей строки
b. Пропускает выполнение следующей строки
c. Выполняет строку самой последней
d. Ничего, это просто пустая команда
5. Сколько вложенных блоков вы можете создать внутри другого вложенного блока?
a. Сколько угодно

b. 1
c. 2
d. 3
6. С помощью какого оператора можно проверить одновременное выполнение нескольких условий?
a. and
b. +
c. or
d. not
7. В какой ситуации будут работать точки останова?
a. Если запустим программу в обычном режиме
b. Если запустим программу в режиме отладки
c. Если произойдет ошибка
d. После второго и всех последующих запусков программы
8. Где логичнее всего поставить точку останова?
a. В начале программы
b. В конце программы
c. В месте предполагаемой ошибки
d. За несколько строк до места предполагаемой ошибки
9. Как узнать внутреннее содержимое переменной во время отладки?
a. Увидеть в окне просмотра созданных переменных
b. Увидеть в той строке кода, где она объявлена
c. Написать свое выражение с использованием этой переменной
d. Все вышеперечисленное
10. Как можно безопасно проверить корректность выполнения следующей строчки кода в режиме отладки?
a. Нажать на кнопку «Следующий шаг»
b. Скопировать строчку кода в интерактивную консоль
c. Проверить нельзя
d. Нажать на кнопку «Следующий шаг» со входом внутрь функций и методов

Тест к уроку 1.3

1. В каком цикле мы можем реализовать выход по условию?
a. В цикле for
b. В цикле while
c. В обоих циклах
d. Выход из цикла по условию реализовать нельзя
2. В чем отличие оператора break от continue?
a. break прерывает цикл, а continue переносит на следующую итерацию
b. break можно использовать без цикла, а continue только в цикле
c. break не позволяет вернуться в цикл после прерывания, а continue позволяет
d. Отличий нет
3. В какой ситуации нужны бесконечные циклы?
a. Когда неизвестно условие выхода из цикла
b. Когда неизвестно в какой момент выполнится условие выхода из цикла
c. Когда известно точное количество итераций цикла
d. Когда нам необходимо вызывать еще один цикл в текущем цикле
4. Будет ли выполняться код, расположенный после цикла while?
a. Не будет, так как while всегда работает бесконечно
b. Будет только в случае если указать break в цикле
c. Будет только если есть оператор continue в цикле
d. Будет когда, while закончит свою работу
5. Какое максимальное количество раз может выполниться цикл while?
a. Всегда бесконечно

b. Не известно, если нет условия выхода по достижению счетчиком определенного значения
c. Неопределенное количество раз но строго до момента, пока пользователь не введет команду “выход”
d. Пока не произойдет переполнение памяти
6. В чем принципиальное отличие цикла for от while?
a. Количество повторений цикла for всегда известно перед его выполнением
b. Количество повторений цикла for никогда не зависит от ввода пользователя
c. В цикле for всегда обязательно указание условия выхода
d. Оператор break в цикле for никогда не нужен
7. Какая последовательность чисел будет сгенерирована в range(1,15, 5) ?
a. 1, 5, 15
b. 1, 6, 11
c. 6, 11, 15
d. 10, 15, 20
8. Найдите неверное выражение
a. Счетчик цикла не является обязательным для циклов
b. Цикл while может заменить цикл for во всех случаях
c. Оператор continue заставляет python игнорировать команды стоящие после него
d. Цикл for может заменить цикл while во всех случаях
9. Что можно использовать вместо функции range() в цикле for?
a. Любое число
b. Любую другую функцию
c. Ничего кроме range()
d. Любую последовательность или функцию её возвращающую
10. Что является обязательным при указании убывающего полуинтервала?
a. правое значение полуинтервала должно быть больше левого
b. 3 аргумента в функции range
c. Отрицательный шаг
d. Все вышеперечисленное

Тест к уроку 1.4.

1. Какие типы кавычек лучше использовать при создании строк в Python
a. Одинарные
b. Двойные
c. Любые, главное чтобы были одинаковыми
d. Тройные
2. Можно ли размещать длинный текст на несколько строк в программе?
a. True
b. False
3. Как мы можем вывести на экран слово в кавычках?
a. Можем экранировать внутренние кавычки символом /
b. Любой вариант
c. Можем использовать другой тип кавычек внутри основной строки
d. Можем конкатенировать отдельно слово с кавычками
4. Какова основная цель применения сырых строк?
a. Преобразование из числа в строку
b. Форматированный вывод содержимого строки на экран
c. Удаление пробелов в строке
d. Выводить и обрабатывать строку “как она есть”
5. Какой метод позволит заменить все символы ‘o’ на ‘0’?
a. s.split(‘o’)
b. s.replace(‘o’,’0’)
c. s.join(‘0’)

d. s.strip(‘o’)
6. Как можно изменить содержимое строки?
a. Только если создадим новую строку на основании старой
b. Обратиться к символу по номеру элемента и записать новый на его место
c. Использовать методы строк
d. Использовать цикл для перебора строки посимвольно
7. Какой метод модуля re позволит найти все точки в тексте?
a. re.find(‘.’)
b. re.findall(‘.’)
c. re.sub(‘.’)
d. re.split(‘.’)
8. Какой спецсимвол регулярных выражений позволяет выбрать все символы кроме цифр?
a. ^D
b. [^w]
c. d
d. D
9. Какого типа команды должны использоваться в блоке except?
a. Код, который пытаемся выполнить чтобы отловить ошибку
b. Код, который должен быть выполнен после проверки на ошибку
c. Код, который должен быть выполнен перед проверкой на ошибку
d. Код, который должен быть выполнен, если случится ошибка
10. Как мы можем заменить стандартный вывод сообщения об ошибке на свой?
a. Вызвать функцию print() в блоке try
b. Вызвать функцию print() в блоке except
c. Нельзя заменять стандартные сообщения об ошибках
d. Вызвать функцию print() в блоке finally

Тест к уроку 1.5.

1. Какие объекты можно хранить в списках?
a. Только объекты, принадлежащие одному классу
b. Только строки
c. Любые
d. Только числа
2. Какой номер индекса необходимо использовать первым, если выводить список по одному элементу с конца?
a. -1
b. 0
c. 1
d. Нельзя обращаться сразу к последнему элементу списка
3. За что отвечает третье число в срезе списка?
a. Начало среза
b. Шаг внутри среза
c. Конец среза
d. У срезов нет третьего параметра
4. В чем отличие между удалением элементов через pop() и remove()?
a. pop() удаляет по индексу, а remove() — по значению
b. pop() работает быстрее, чем remove()
c. pop() возвращает удаленный элемент, а remove() — нет
d. Все варианты верны
5. Какой результат будет у данного кода?

```
numbers = (1, 2, 3, 4, 5, 6)
numbers.append(7)
```

a. Число 7 добавится в начало
b. Ошибка
c. Число 7 добавится в конец
d. numbers никак не изменится
6. С помощью какой функции можно отсортировать элементы в множестве?
a. Множество — неупорядоченная последовательность, а значит, сортировать его нельзя
b. Метод sort()
c. Нет готовой функции, необходимо писать собственную
d. Воспользоваться готовыми алгоритмами (сортировка пузырьком, методом вставки и т.п.)
7. В чем преимущество метода get() перед оператором выбора [] при обращении к значению словаря?
a. Преимущества нет, результат будет всегда одинаковый
b. В том, что в случае отсутствии ключа ошибки не будет
c. Работает быстрее
d. Не изменяет словарь
8. Что произойдет, если по не существующему в словаре ключу запишем новое значение?
a. Программа выдаст исключение
b. Словарь останется неизмененным
c. Создастся ключ со значением None
d. Создастся новая пара «ключ — значение»
9. Какой результат вернет метод items() у словаря?
a. Общий список всех ключей и значений
b. Набор ключей
c. Набор значений
d. Набор кортежей «ключ — значение»
10. Необходимо пользователю при указании неверного ключа словаря my_dict выдавать сообщение «Invalid». Как мы это можем сделать?
a. my_dict[‘Invalid’]
b. my_dict.get(key, ‘Invalid’)
c. my_dict.get(‘Invalid’)
d. my_dict[key] = ‘Invalid’

Тест к уроку 1.6.

1. Сколько раз может быть вызвана созданная функция (не рекурсивная) за один запуск программы?
a. Только 1
b. Пока не переполнится стек вызова
c. Бесконечное
d. 2 раза (в начале и в конце работы программы)
2. Как создать функцию с параметрами, которые не обязательно передавать при вызове?
a. Сделать все параметры со значениями по умолчанию
b. Присвоить значения параметрам внутри тела функции
c. Так сделать нельзя
d. Присвоить значения параметрам до вызова функции
3. Что будет если при вызове функции передать свое значение в параметр, для которого определено значение по умолчанию?
a. Останется значение указанное по умолчанию
b. Значение заменится на указанное
c. Ошибка вызова функции
d. Переданное значение будет следующим в списке параметров
4. В чем отличие между *args и **kwargs?
a. args нужен для обычных функций, kwargs - для рекурсивных

b. Через args невозможно передать словари, через kwargs невозможно передать списки
c. args позволяет передать одно значение, kwargs - сразу два
d. args передает значения в виде кортежа, kwargs в виде словаря
5. Что делает оператор global?
a. указывает, что будет использована переменная, объявленная в глобальной области видимости
b. Изменяет тип области видимости функции на глобальный
c. Создает переменную в глобальной области видимости
d. создает глобальную функцию, доступную для любых программ
6. Что будет, если мы присвоим новое имя функции?
a. Можно будет вызывать функцию как по новому так и по старому имени
b. Можно будет использовать только новое имя
c. Исключение, функции после создания переименовывать нельзя
d. Ничего, функция будет доступна только по старому имени
7. Данные какого типа мы можем возвращать из функций?
a. Только кортежи
b. Любого
c. Только в виде других функций
d. Только словари
8. В каких случаях функции возвращают значение?
a. Только функции с оператором return
b. Только если возвращаем данные стандартного типа
c. Столько если возвращаются коллекции
d. По всех
9. Сколько максимально рекурсивных вызовов может выполнить функция?
a. 995
b. 1000
c. 10
d. Будет выполнять, пока не заполнится стек
10. Что нужно сделать, первым делом, чтобы рекурсия не была бесконечной
a. Выполнить рекурсивный вызов
b. Продумать и реализовать способ завершения рекурсии
c. Вернуть значение из рекурсивной функции
d. Постараться избежать рекурсии и реализовать обычным способом

Тест к уроку 1.7.

1. В чем отличие итератора от итерируемого объекта?
a. Итератор - это часть итерируемого объекта
b. Итератор нельзя использовать в цикле
c. Элементы итератора можно получить только один раз
d. Отличий нет
2. Итератор содержит последовательность из 5 чисел. Какой результат получим, если применим функцию next() 6 раз?
a. Исключение StopIteration
b. 0
c. Пустую строку
d. None
3. Какой объект будет меньше по размеру?
a. Список четных чисел от 1 до 500 полученный с помощью генератора списков
b. Генератор всех чисел от 1 до 10000
c. Список нечетных чисел от 1 до 500, полученный с помощью цикла for
d. Список всех чисел от 1 до 500, полученный с помощью цикла while?
4. Сколько раз мы можем посчитать сумму чисел, получаемых с помощью одного генератора?

a. Бесконечное количество раз
b. Сможем считать пока не получим исключение Memory Error
c. Только два раза
d. Только один раз
5. В каких случаях будет выполняться код после оператора yield?
a. При каждом вызове функции генератора начиная со второго раза
b. Будет выполняться всегда
c. Никогда не выполнится, так как yield возвращает значение из функции
d. Выполнится только один раз при первом вызове
6. Чем yield отличается от return?
a. yield запоминает состояние функции, чтобы при следующем вызове продолжить выполнять код функции дальше
b. return возвращает конкретное значение, yield ничего не возвращает
c. yield может использоваться не только в функциях
d. Ничем. Это абсолютно одинаковые операторы
7. Сколько выражений может выполнять анонимная функция?
a. Произвольное количество
b. Только одно
c. Ни одного, так как у функции нет имени
d. Только два
8. Занимает ли анонимная функция пространство в памяти?
a. Занимает также как и обычная функция
b. Никогда не занимает память. В этом её преимущество
c. Занимает в 2 раза меньше чем обычная функция
d. Занимает только тогда, когда исполняется
9. Как чаще всего используются лямбда функции?
a. Как самостоятельные функции
b. Как элементы последовательностей
c. Как композиции функций
d. Как параметры для функций высшего порядка
10. Найдите неверное утверждение
a. Лямбда функцию можно превратить в обычную присвоив ей имя
b. Лямбда функции ничего не возвращают, так как в них отсутствует оператор return
c. Лямбда функции можно возвращать как результат работы
d. Лямбда функции могут принимать только один параметр

Тест к уроку 1.8.

1. Как можно открыть файл одновременно на чтение и запись?
a. Так сделать нельзя
b. Использовать режим ‘a’
c. Использовать режим ‘w+’ или ‘r+’
d. Использовать режим ‘x’
2. При каком условии можно не закрывать файл после выполнения операций
a. Если открывается файл с использованием оператора with
b. Если файл был открыт внутри функции
c. Файл нужно всегда закрывать самостоятельно
d. Если файл открывался только для чтения
3. В чем отличие методов json.dump() и json.dumps()?
a. dumps() - для записи в файл json, dump() - для его чтения
b. dumps() конвертирует данные в строку, dump() - записывает как есть
c. Отличий нет
d. dumps() возвращает результат операции, dump() - не возвращает
4. Какой символ может быть использован в качестве разделителя в CSV файлах?
a. Только запятая

b. Только запятая, но если она есть в данных, то точку
c. Точка. Если есть точки в данных, то запятая
d. любой, но предпочтительно использовать запятую
5. В какой ситуации удобно применять DictReader вместо reader?
a. Когда большое количество столбцов в таблице
b. Когда большое количество строк в таблице
c. Когда работаем с несколькими csv файлами
d. Разницы нет
6. Для чего указывать параметр fieldnames при записи CSV с использованием класса DictWriter?
a. Чтобы явно задать порядок следования столбцов
b. Чтобы можно было указать свои наименования столбцов
c. Чтобы не нарушать типы данных в столбцах
d. Чтобы преобразовать значения столбцов в строки
7. В чем разница работы os.listdir(“.”) и os.walk(“.”)?
a. listdir - возвращает список, walk - возвращает генератор
b. listdir - позволяет узнать содержимое текущего каталога, walk - содержимое текущего каталога и всех вложенных
c. результат listdir можно использовать несколько раз. Результат walk - только один
d. Все вышеперечисленное
8. С какого элемента списка мы начинаем чтение параметров, переданных в sys.args?
a. С первого
b. Со второго
c. С третьего
d. С четвертого
9. Каким образом мы можем изменить имя импортируемого объекта?
a. Изменять нельзя
b. Воспользоваться функцией os.path.basename()
c. Воспользоваться функцией os. rename()
d. Использовать оператор as
10. Какую роль играет содержимое файла __init__.py
a. Позволяет интерпретатору python понять, что файлы внутри относятся к библиотеке
b. Код внутри выполняется при импорте любого файла из директории, в которой он находится
c. Позволяет не выполнять код, который написан внутри при импорте этого файла
d. Позволяет среде разработки видеть папку, в которой он находится

Тест к теме 1. «Базовый Python»

1. Каким образом можно получить значение типа int после выполнения операции деления?
a. Делить только целые числа
b. Использовать целочисленное деление
c. Взять модуль получившегося числа
d. Такое невозможно
2. С помощью какого оператора можно изменить порядок вычисления в Python?
a. Скобки
b. Умножение
c. Возведение в степень
d. Остаток от деления
3. Сколько значений содержит булев тип?
a. 1
b. 2
c. 3
d. Бесконечное количество

4. С помощью какого оператора можно проверить одновременное выполнение нескольких условий?
a. and
b. +
c. or
d. not
5. В каком цикле мы можем реализовать выход по условию?
a. В цикле for
b. В цикле while
c. В обоих циклах
d. Выход из цикла по условию реализовать нельзя
6. Какая последовательность чисел будет сгенерирована в range(1,15, 5) ?
a. 1, 5, 15
b. 1, 6, 11
c. 6, 11, 15
d. 10, 15, 20
7. Какой метод позволит заменить все символы ‘o’ на ‘0’?
a. s.split(‘o’)
b. s.replace(‘o’,’0’)
c. s.join(‘0’)
d. s.strip(‘o’)
8. Можно ли размещать длинный текст на несколько строк в программе?
a. True
b. False
9. Какие объекты можно хранить в списках?
a. Только объекты, принадлежащие одному классу
b. Только строки
c. Любые
d. Только числа
10. Какой результат вернет метод items() у словаря?
a. Общий список всех ключей и значений
b. Набор ключей
c. Набор значений
d. Набор кортежей «ключ — значение»
11. Что будет если при вызове функции передать свое значение в параметр, для которого определено значение по умолчанию?
a. Останется значение указанное по умолчанию
b. Значение заменится на указанное
c. Ошибка вызова функции
d. Переданное значение будет следующим в списке параметров
12. Сколько максимально рекурсивных вызовов может выполнить функция?
a. 995
b. 1000
c. 10
d. Будет выполнять, пока не заполнится стек
13. Как чаще всего используются лямбда функции?
a. Как самостоятельные функции
b. Как элементы последовательностей
c. Как композиции функций
d. Как параметры для функций высшего порядка
14. При каком условии можно не закрывать файл после выполнения операций
a. Если открывается файл с использованием оператора with
b. Если файл был открыт внутри функции

c. Файл нужно всегда закрывать самостоятельно
d. Если файл открывался только для чтения

Тест к уроку 2.1.

1. Что подразумевается под термином «замыкание»?
a. Возможность вызова функций внутри других функций
b. Возможность вернуть вызов функции
c. Функция, которая запоминает значения из своей внешней области видимости, даже если эта область уже недоступна
d. Функция, которая возвращает лямбду-функцию
2. В чем заключается основной смысл декораторов?
a. В возможности изменять поведение функции без изменения ее кода
b. В возможности вызвать несколько функций одновременно
c. В возможности изменить имя функции
d. В возможности измерять время работы функций
3. В какой последовательности применяются декораторы к функции, когда их несколько?
a. В обратном от порядка их указания в коде (сверху вниз)
b. В порядке следования (снизу вверх)
c. Все одновременно
d. Применяется только один декоратор, который объявлен ранее всех
4. Как можно передать параметр в декорируемую функцию?
a. Указать этот параметр в декораторе
b. Создать дополнительную функцию внутри, которая примет этот параметр
c. В декорируемую функцию нельзя передавать параметры
d. Указать этот параметр в функции, возвращаемой через декоратор
5. Сколько параметров по умолчанию принимает декоратор?
a. Один параметр — функцию, которую он декорирует
b. Два параметра: функцию и пользовательские параметры в виде *args
c. Ни одного параметра
d. Один параметр — пользовательский
6. Как можно передать пользовательские параметры в декоратор?
a. Создать в декораторе дополнительную функцию, которая будет принимать этот параметр
b. Указать дополнительные параметры при создании декоратора
c. В декоратор нельзя передавать пользовательские параметры
d. Воспользоваться *args
7. Что может возвращать декоратор?
a. Только обычные функции
b. Только анонимные функции
c. Только адрес декорируемой функции
d. Любой объект
8. В чем смысл перегрузки?
a. В возможности создать функцию, в которую можно передавать разное количество параметров
b. В возможности работы операторов и функций с разным количеством параметров и разными типами данных
c. В возможности функции по-разному реагировать на разные данные
d. В возможности функции возвращать несколько объектов
9. Каким образом можно проверить принадлежность объекта к определенному классу?
a. С помощью функции isinstance()
b. С помощью функции type()
c. Сравнить результат работы функции type с одним из существующих классов
d. Все вышеперечисленное

10. В чем разница между singledispatch и multipledispatch
a. singledispatch можно применить только к одной функции, multipledispatch — к любому количеству
b. Разные модули для реализации одиночной отправки (singledispatch) и множественной отправки (multipledispatch)
c. singledispatch — декоратор, который принимает только один параметр, multipledispatch — любое количество параметров
d. multipledispatch — это декоратор, который расширяет возможности singledispatch

Тест к уроку 2.2.

1. Что подразумевается под термином ООП?
a. Способность Python создавать разные пространства имен для объектов
b. Название всех классов, объектов, их методов и свойств
c. Способ написания кода, при котором задачи решаются через классы и их объекты
d. Возможность разделять объекты на методы и свойства
2. В чем разница между функцией dir() и свойством __dict__?
a. dir() показывает набор методов и свойств объекта, __dict__ отображает содержимое пространства имен у указанного объекта
b. dir() содержит методы объекта, __dict__ свойства объекта
c. dir() работает только с классами, __dict__ только с объектами классов
d. Разницы нет, результат один и тот же
3. Как можно узнать какому классу принадлежит объект?
a. type(obj)
b. isinstance(obj, Class)
c. obj.__class__
d. Все вышеперечисленное
4. Каким образом мы можем изменить содержимое словаря __dict__
a. Обратившись напрямую по ключу, в виде имени атрибута
b. Обратившись через класс по ключу, в виде имени атрибута
c. __dict__ является неизменяемым словарем
d. Через дот-нотацию, указав новое значение атрибуту
5. Что будет, если класс вызвать как функцию?
a. Вернется экземпляр класса
b. Исключение - классы не являются вызываемыми объектами
c. Вернется экземпляр класса, или None если в классе будет отсутствовать return
d. Ничего не произойдет
6. Какой результат получим, если обратимся к свойству, которого нет у объекта?
a. Python произведет поиск свойства у класса и если не найдет его и там, то выдаст исключение AttributeError
b. Исключение AttributeError
c. Python произведет поиск свойства у класса? затем в глобальной области видимости и если ничего не найдет, то выдаст исключение AttributeError
d. Получим None
7. Сколько параметров по умолчанию принимает функция класса?
a. Один параметр - ссылку на объект у которого она будет вызвана
b. Два параметра - ссылку на объект и пользовательские параметры в виде *args
c. Ни одного параметра
d. *args и **kwargs
8. Как мы можем назвать переменную, содержащую ссылку на объект?
a. Только self
b. Любым именем, но рекомендуется использовать self
c. Только object
d. Только instance
9. В чем основное назначение метода __init__()?

a. Создать экземпляр класса
b. Связать класс с объектами
c. Инициализировать пространство имен класса
d. Присвоить свойствам объекта указанные значения или реализовать логику проверки данных для объекта
10. Сколько параметров может принимать __init__()?
a. 1
b. 2
c. 3
d. Бесконечное количество

Тест к уроку 2.3.

1. Что подразумевается под термином “Инкапсуляция”?
a. Создание методов и свойств класса доступных только для чтения
b. Переопределение методов класса с целью возможности выполнения математических операций с объектами
c. Процесс скрытия методов и свойств класса
d. Возможность использовать объекты класса в качестве ключей словаря
2. В чем отличие атрибутов self._name и self.__name?
a. К self._name можно получить доступ из объекта, к self.__name - нельзя
b. self._name можно сделать доступным через getter, self.__name - нельзя
c. Для self._name можно определить setter, для self.__name - нельзя
d. Разницы нет - это оба приватные свойства с одинаковым поведением
3. В чем польза Name Mangling?
a. Позволяет создавать несколько имен одному свойству
b. Делает класс read-only
c. Делает свойства объектов доступными только для чтения
d. Защищает приватные свойства от случайной перезаписи другим значением
4. Что будет, если мы присвоим значение приватному свойству через объект?

```
person.__name = 'Ivan'
```

a. Получим исключение, нельзя изменять приватные свойства напрямую через объект
b. Сможем изменить значение приватного свойства
c. Объект останется не измененным
d. Python просто создаст новое свойство __name у объекта person
5. В чем разница между классом property и декоратором @property ?
a. Разницы нет. Декоратор - просто синтаксический сахар использования класса
b. Класс позволяет объявить и сеттеры и геттеры. Декоратор - только геттеры
c. Декоратор можно использовать только для одного свойства класса. Класс property - для любого количества
d. Декоратор позволяет создавать свойства только для чтения. Класс property - не позволяет так делать
6. В чем главная особенность статических методов класса?
a. Они могут быть вызваны без создания экземпляра класса.
b. Они могут быть преобразованы в обычные функции.
c. Они не имеют доступа к свойствам экземпляра класса.
d. Все вышеперечисленное.
7. Сколько одинаковых статических методов инициализируется во время работы?
a. Один метод на все экземпляры класса
b. По одному для каждого экземпляра класса
c. Один метод для самого класса + столько сколько есть экземпляров класса
d. Статические методы не инициализируются вообще, так как не работают с экземплярами класса
8. В чем особенность методов класса?

a. В том, что они не имеют доступа к атрибутам ни класса ни объектов
b. В том, что они имеют доступ только к свойствам и функциям класса
c. В том, что они могут создавать другие классы
d. В том, что с ними можно определить несколько конструкторов
9. В каких ситуациях нужно определять магический метод __add__() в пользовательском классе?
a. Когда нужно выполнять операции сложения свойств внутри класса
b. Когда нужно определить правила сложения двух классов
c. Когда в классе определены функции , выполняющие любые математические вычисления с оператором “+”
d. Когда часто происходит математическое сложение объектов этого класса по определенным правилам
10. Какие свойства могут принимать участие при формировании результата, возвращаемого из метода __hash__()?
a. Приватные свойства
b. Только свойства класса
c. только свойства объектов
d. Неизменяемые свойства

Тест к уроку 2.4.

1. В чем заключается основной принцип наследования?
a. Можно создать методы и свойства, которые будут только у заранее определенных дочерних классов
b. Можно скрыть реализацию методов в родительском классе
c. Дочерние классы получают все функции и свойства родительского класса
d. Родительские классы получают доступ ко всем методам и свойствам дочерних классов
2. Какое количество родительских классов может быть у пользовательского класса?
a. Любое
b. Только один
c. Всегда два родительских класса
d. Максимум три класса
3. Представьте такую иерархию классов. Что выведет данный код?

```
class Food:
   pass

class Fruits(Food):
   pass

class Apple(Fruits):
   pass

a = Apple()
print(isinstance(a, Food))
```

a. False
b. True
c. TypeError
d. None
4. В чем преимущество функции super() перед указанием напрямую родительского класса?
a. Позволяет получить доступ к пространству имен родительского класса
b. Изолирует содержимое родительского класса от дочернего класса
c. Создает дополнительный объект для взаимодействия с родительским классом

d. Избавляет от зависимости от имени родительского класса
5. С каким классом связывается объект, у которого был вызван метод, обращающийся к super() классу?
a. С тем классом, экземпляром которого он является
b. С родительским классом, на который указывает super()
c. С самым старшим в цепочке наследования классов
d. В этом случае не связывается вообще
6. В каком классе функция super() ищет определение метода, который был вызван?
a. Во всех классах-предках
b. В родительском классе
c. В самом старшем классе цепочки наследования
d. В текущем классе
7. Какое поведение предусмотрено в случае возникновения конфликта имен при множественном наследовании?
a. Выкидывается исключение AttributeError
b. Идет обращение к объекту любого случайного класса, который указан при наследовании
c. Идет обращение к объекту класса, который указан последним в списке наследования
d. Идет обращение к объекту класса, который указан первым в списке наследования
8. В каких случаях нужно применять миксины?
a. Когда у всех дочерних классов есть одинаковые методы
b. Когда есть общий функционал разных классов, не требующий сложной реализации
c. Когда создаем много дочерних классов
d. Когда применяем множественное наследование
9. Чем абстрактный класс отличается от обычного?
a. Нельзя создавать объекты абстрактного класса
b. Методы абстрактных классов не имеют конкретной реализации
c. Абстрактные методы обязательно должны быть определены во всех дочерних классах без исключения
d. Все вышеперечисленное
10. В каких ситуациях абстрактные классы являются более корректным решением?
a. Когда количество наследуемых методов становится очень большим
b. Когда необходимо отвязаться от конкретных реализаций с целью улучшения масштабируемости кода
c. Когда мы пока не знаем, как будем реализовывать методы класса
d. Все вышеперечисленное

Тест к уроку 2.5.

1. Когда нужно использовать дескрипторы?
a. Когда количество классов превышает три
b. Когда у класса много разных свойств
c. Когда у класса много свойств с одинаковым поведением
d. Когда у класса много методов
2. Какие основные методы всегда используются в дескрипторах?
a. __get__ и __set__
b. __set__ и __set_name__
c. __set_name__ и _delete__
d. __get__ и __delete__
3. Сколько всего методов образуют прокол дескриптора?
a. Три: __get__, __set__ и __delete___
b. Четыре: __get__, __set__, __delete___ и __set_name__
c. Два: __get__, __set__
d. Один: только __get__
4. В чем отличие data-дескрипторов от non-data дескрипторов?

a. Non-data дескрипторы только для чтения, data-дескрипторы — для чтения и записи
b. Non-data дескрипторы поддерживают только метод __get__, data-дескрипторы — методы __get__ и __set__
c. Non-data дескрипторы не поддерживают изменение значений пользователем, data-дескрипторы поддерживают
d. Все вышеперечисленное
5. Где должны храниться значения свойств объекта при использовании дескрипторов?
a. В экземплярах пользовательского класса
b. В экземплярах дескриптора
c. В пользовательском классе
d. В классе дескриптора
6. Как можно передать пользовательские параметры в декоратор?
a. Создать дополнительную функцию в декораторе, которая будет принимать этот параметр
b. Указать дополнительные параметры при создании декоратора
c. В декоратор нельзя передавать пользовательские параметры
d. Воспользоваться *args
7. Сколько ссылок на объект создается в случае использования экземпляров дескрипторов в качестве свойств объектов?
a. Столько же, сколько свойств у объекта
b. Одна ссылка
c. Ссылки на объект не создаются, так как дескриптор содержит их в себе
d. На одну больше, чем свойств у объекта
8. Что происходит с объектом — слабой ссылкой, когда удаляется объект, на который она указывала?
a. Слабая ссылка полностью удаляется
b. Слабая ссылка изменяет состояние на dead
c. Слабая ссылка продолжает хранить ссылку на удаленный объект
d. Вместо слабой ссылки начинает возвращаться значение None
9. В какой момент вызывается метод дескриптора __set_name__?
a. В момент чтения свойства, являющегося дескриптором
b. В момент записи нового значения в свойство, являющееся дескриптором
c. В момент удаления свойства, являющегося объектом-дескриптором
d. В момент создания свойства, являющегося объектом-дескриптором
10. Какое основное назначение метода __set_name__?
a. Для обеспечения обращения к свойствам объекта всегда через методы дескриптора
b. Одинаковая обработка входных данных при создании разных свойств разных классов
c. Для сохранения данных всегда только в локальном словаре __dict__
d. Для создания слабых ссылок на свойства объектов в дескрипторе

Тест к уроку 2.6.

1. Что на самом деле представляет собой type()?
a. Отдельная функция, выводящая тип данных объекта
b. Объект метакласса
c. Класс, объектами которого являются пользовательские классы
d. Метод класса, возвращающий родительский класс
2. Укажите правильный способ передачи методов calc и remove в пользовательский класс.
a. type(‘MyClass’, (), {‘a’: 50, ‘calc’: calc, ‘remove’:remove})
b. type(‘MyClass’, (calc, remove), {‘a’:50})
c. type(‘MyClass’, methods = {‘calc’: calc, ‘remove’: remove}, {‘a’:50})
d. Методы передавать в класс нельзя
3. Какой метод отвечает за инициализацию параметров метакласса?
a. __init__()

b. __new__()
c. __cls__()
d. __super__()
4. Как можно создать свойство для метакласса, доступное для всех дочерних классов?
a. self.property = 50
b. cls.property = 50
c. property = 50
d. dict['property'] = 50
5. В чем заключается основное назначение классов данных?
a. В простой реализации хранения данных объектов
b. В реализации дополнительных обработок данных
c. В простой реализации хранения методов класса
d. В раздельном использовании методов и свойств классов
6. Что обязательно при объявлении полей классов данных?
a. Указание типов данных полей
b. Использование функции field()
c. Определение конструктора __init__, сохраняющего передаваемые значения
d. Определение функции постобработки __post_init__
7. В чем отличие работы метода __eq__ в обычных классах и классах данных?
a. В классах данных метод __eq__ переопределить нельзя, в обычных классах — можно
b. В классах данных метод __eq__ может принимать более двух объектов для сравнения, в обычных — только два.
c. В классах данных метод __eq__ генерируется автоматически, в обычных его необходимо определять вручную
d. В классах данных сравнение происходит по полям объектов, а в обычных — по адресам объектов
8. Почему нельзя создавать неинициализированные поля в мутабельных классах данных?
a. Потому что в классах данных все поля должны быть инициализированы
b. Потому что неинициализированные поля подразумевают изменение значения свойства, которое запрещено
c. Потому что нарушается порядок объявления параметров
d. Потому что вычисленное значение будет некорректным
9. Как можно изменять правила обработки отдельных полей класса данных?
a. Если наследоваться от метакласса
b. Если создать поле, являющееся объектом другого класса
c. Изменять поведение отдельных полей нельзя
d. С помощью параметров функции field()
10. При каком условии лучше использовать обычный класс вместо класса данных?
a. Если создается слишком много полей
b. Если приходится изменять слишком много стандартных реализаций классов данных
c. Если для полей данных могут быть использованы объекты разных типов
d. Если все поля должны быть только для чтения

Тест к теме 2. «Продвинутый Python»

1. Что подразумевается под термином «замыкание»?
a. Возможность вызова функций внутри других функций
b. Возможность вернуть вызов функции
c. Функция, которая запоминает значения из своей внешней области видимости, даже если эта область уже недоступна
d. Функция, которая возвращает лямбду-функцию
2. В чем смысл перегрузки?
a. В возможности создать функцию, в которую можно передавать разное количество параметров

b. В возможности работы операторов и функций с разным количеством параметров и разными типами данных
c. В возможности функции по-разному реагировать на разные данные
d. В возможности функции возвращать несколько объектов
3. Что подразумевается под термином ООП?
a. Способность Python создавать разные пространства имен для объектов
b. Название всех классов, объектов, их методов и свойств
c. Способ написания кода при котором задачи решаются через классы и их объекты
d. Возможность разделять объекты на методы и свойства
4. Сколько параметров по умолчанию принимает функция класса?
a. Один параметр - ссылку на объект у которого она будет вызвана
b. Два параметра - ссылку на объект и пользовательские параметры в виде *args
c. Ни одного параметра
d. *args и **kwargs
5. Что подразумевается под термином “Инкапсуляция”?
a. Создание методов и свойств класса доступных только для чтения
b. Переопределение методов класса с целью возможности выполнения математических операций с объектами
c. Процесс скрытия методов и свойств класса
d. Возможность использовать объекты класса в качестве ключей словаря
6. Какие свойства могут принимать участие при формировании результата, возвращаемого из метода __hash__()?
a. Приватные свойства
b. Только свойства класса
c. только свойства объектов
d. Неизменяемые свойства
7. Представьте такую иерархию классов. Что выведет данный код?

```
class Food:
   pass

class Fruits(Food):
   pass

class Apple(Fruits):
   pass

a = Apple()
print(isinstance(a, Food))
```

a. False
b. True
c. TypeError
d. None
8. В каких случаях нужно применять миксины?
a. Когда у всех дочерних классов есть одинаковые методы
b. Когда есть общий функционал разных классов, не требующий сложной реализации
c. Когда создаем много дочерних классов
d. Когда применяем множественное наследование
9. Где должны храниться значения свойств объекта при использовании дескрипторов?
a. В экземплярах пользовательского класса
b. В экземплярах дескриптора
c. В пользовательском классе
d. В классе дескриптора

10. Какое основное назначение метода __set_name__?
a. Для обеспечения обращения к свойствам объекта всегда через методы дескриптора
b. Одинаковая обработка входных данных при создании разных свойств разных классов
c. Для сохранения данных всегда только в локальном словаре __dict__
d. Для создания слабых ссылок на свойства объектов в дескрипторе
11. Что на самом деле представляет собой type()?
a. Отдельная функция, выводящая тип данных объекта
b. Объект метакласса
c. Класс, объектами которого являются пользовательские классы
d. Метод класса, возвращающий родительский класс
12. Почему нельзя создавать неинициализированные поля в мутабельных классах данных?
a. Потому что в классах данных все поля должны быть инициализированы
b. Потому что неинициализированные поля подразумевают изменение значения свойства, которое запрещено
c. Потому что нарушается порядок объявления параметров
d. Потому что вычисленное значение будет некорректным

Итоговое тестирование
1. Каким образом можно получить значение типа int после выполнения операции деления?
a. Делить только целые числа
b. Использовать целочисленное деление
c. Взять модуль получившегося числа
d. Такое невозможно
2. С помощью какого оператора можно изменить порядок вычисления в Python?
a. Скобки
b. Умножение
c. Возведение в степень
d. Остаток от деления
3. Сколько значений содержит булев тип?
a. 1
b. 2
c. 3
d. Бесконечное количество
4. Какой метод позволит заменить все символы ‘o’ на ‘0’?
a. s.split(‘o’)
b. s.replace(‘o’,’0’)
c. s.join(‘0’)
d. s.strip(‘o’)
5. Какой результат вернет метод items() у словаря?
a. Общий список всех ключей и значений
b. Набор ключей
c. Набор значений
d. Набор кортежей «ключ — значение»
6. Сколько максимально рекурсивных вызовов может выполнить функция?
a. 995
b. 1000
c. 10
d. Будет выполнять, пока не заполнится стек
7. В чем смысл перегрузки?
a. В возможности создать функцию, в которую можно передавать разное количество параметров

b. В возможности работы операторов и функций с разным количеством параметров и разными типами данных
c. В возможности функции по-разному реагировать на разные данные
d. В возможности функции возвращать несколько объектов
8. Сколько параметров по умолчанию принимает функция класса?
a. Один параметр - ссылку на объект у которого она будет вызвана
b. Два параметра - ссылку на объект и пользовательские параметры в виде *args
c. Ни одного параметра
d. *args и **kwargs
9. Какие свойства могут принимать участие при формировании результата, возвращаемого из метода __hash__()?
a. Приватные свойства
b. Только свойства класса
c. только свойства объектов
d. Неизменяемые свойства
10. Где должны храниться значения свойств объекта при использовании дескрипторов?
a. В экземплярах пользовательского класса
b. В экземплярах дескриптора
c. В пользовательском классе
d. В классе дескриптора
11. В каких случаях нужно применять миксины?
e. Когда у всех дочерних классов есть одинаковые методы
f. Когда есть общий функционал разных классов, не требующий сложной реализации
g. Когда создаем много дочерних классов
h. Когда применяем множественное наследование
12. Почему нельзя создавать неинициализированные поля в мутабельных классах данных?
a. Потому что в классах данных все поля должны быть инициализированы
b. Потому что неинициализированные поля подразумевают изменение значения свойства, которое запрещено
c. Потому что нарушается порядок объявления параметров
d. Потому что вычисленное значение будет некорректным

ОРГАНИЗАЦИОННО-ПЕДАГОГИЧЕСКИЕ УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ

Требования к квалификации педагогических кадров, представителей предприятий и организаций, обеспечивающих реализацию образовательного процесса

Требования к образованию и обучению лица, занимающего должность преподавателя: высшее образование — специалитет или магистратура, направленность (профиль) которого, как правило, соответствует преподаваемому учебному курсу, дисциплине (модулю).

Дополнительное профессиональное образование на базе высшего образования (специалитета или магистратуры) — профессиональная переподготовка, направленность (профиль) которой соответствует преподаваемому учебному курсу, дисциплине (модулю).

Педагогические работники обязаны проходить в установленном законодательством Российской Федерации порядке обучение и проверку знаний и навыков в области охраны труда.

Рекомендуется обучение по дополнительным профессиональным программам по профилю педагогической деятельности не реже чем один раз в три года.

Требования к опыту практической работы: при несоответствии направленности (профиля) образования преподаваемому учебному курсу, дисциплине (модулю) — опыт работы в области профессиональной деятельности, осваиваемой слушателями или соответствующей преподаваемому учебному курсу, дисциплине (модулю).

Преподаватель: стаж работы в образовательной организации не менее одного года; при наличии ученой степени (звания) — без предъявления требований к стажу работы.

Особые условия допуска к работе: отсутствие ограничений на занятие педагогической деятельностью, установленных законодательством Российской Федерации

Прохождение обязательных предварительных (при поступлении на работу) и периодических медицинских осмотров (обследований), а также внеочередных медицинских осмотров (обследований) в порядке, установленном законодательством Российской Федерации

Прохождение в установленном законодательством Российской Федерации порядке аттестации на соответствие занимаемой должности.

Требования к материально-техническим условиям

Организация проводит занятия по адресу: г. Москва, ул. Сущевский Вал, д. 18. Аудитории для занятий расположены на 11-м этаже здания.

Все занимаемые помещения соответствуют обязательным нормам пожарной безопасности и требованиям санитарно-эпидемиологических служб. Помещения имеют централизованные системы водоснабжения, отопления и канализации. Воздухообмен помещений обеспечивается современными системами кондиционирования, за счет приточно-вытяжной вентиляционной системы.

Учебным центром СКБ Контур заключен договор с организацией общественного питания о возможности обеспечения слушателей питанием.

В учебной аудитории проводятся лекции и практические занятия. Аудитория оснащена столами и стульями, в составе учебного оснащения маркерная доска и флипчарт, в случае необходимости подключается мультимедийный проектор, слушателям предоставляются компьютеры.

Компьютерная сеть учебного центра оснащена необходимым оборудованием для доступа в интернет по выделенному каналу. На каждом компьютере обеспечен постоянный доступ к компьютерной программе «Контур.Школа».

Для проведения вебинаров и онлайн-трансляций используется оснащенная современным оборудованием видеостудия:

- помещение оборудовано посадочными местами для спикера(ов);
- спикеру предоставляется персональный компьютер с соответствующими мультимедийными характеристиками (Intel Core i3 либо идентичные по характеристикам, оперативная память: от 4 Гб и выше для всех ОС), со стабильным соединением с сетью Интернет на скорости не менее 1 Мбит/с;
- видеокамера (максимальное разрешение видео — не менее 3840 x 2160).

Размещение материалов вебинаров и доступ к ним участников обеспечивает техническая платформа (сайт, система управления сайтом, другие технические средства):

1. Трансляция вебинара в режиме реального времени.
2. Хранение, систематизация записей вебинаров, с предоставлением участникам возможности просмотра записи онлайн.
3. Хранение, систематизация и доступ к скачиванию материалов учебных программ.
4. Напоминание участникам о предстоящем вебинаре за 1 час до начала мероприятия.
5. Использование защищенных соединений, передача и прием видео и звука по протоколам RTMP(S) или аналогичным.
6. Управление качеством и разрешением передаваемого/принимаемого видео вплоть до разрешения HD 720p на каждого участника мероприятия (адаптивный стриминг).
7. Обмен короткими текстовыми сообщениями (чат).
8. Осуществление записи мероприятий в формате, не требующем конвертации для проигрывания (mp4, AVI, WMA и т.д.).
9. Система регистрации на вебинар.
10. Техническое сопровождение проведения вебинара.
11. Отображение числа участников.
12. Техническая доступность услуги не менее 99,8% времени.
13. Устойчивость при проведении вебинара при единовременном подключении до 3000 участников.
14. Возможность участия пользователей на вебинарах в браузерах Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Apple Safari с установленным плагином Adobe Flash Player.
15. Передача аудио- и видеоинформации на персональные компьютеры участников реализована при скорости интернет-соединения не менее 134 кбит/с.

Основные функции программы Контур.Школа:
1. Размещение расписания и описания учебных программ и условий обучения.
2. Онлайн-трансляция учебных занятий с возможностью обратной связи.
3. Размещение тестов и проведение онлайн-тестирования.
4. Размещение и выбор образовательного контента и заданий для слушателей.
5. Хранение учебно-методических материалов.
6. Обратная связь слушателей к организаторам и преподавателям.

7. Автоматическая фиксация хода учебного процесса, промежуточных и итоговых результатов слушателей.
8. Хранение информации о ходе учебного процесса и результатов обучения в течение периода обучения.
9. Сбор и хранение заявок на обучение и сведений о слушателях.
10. Создание и актуализация контента и учебно-методических материалов.
11. Информационно-консультационное обслуживание слушателей.

Требования к информационным и учебно-методическим условиям

Образовательная программа обеспечивается учебно-методическими материалами по всем модулям образовательной программы.

Фонд учебно-научной библиотеки содержит основную и дополнительную учебную, учебно-методическую, научную литературу, справочно-библиографические и периодические издания (в том числе и на электронных носителях) по всем темам и дисциплинам реализуемой программы.

Нормативно-правовая база

1. Федеральный закон «Об информации, информационных технологиях и о защите информации» от 27.07.2006 №149-ФЗ
2. ГОСТ Р ИСО/МЭК 12207-2010 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств» (утвержден приказом Росстандарта от 30 ноября 2010 г. N 631-ст);
3. ГОСТ Р ИСО/МЭК 14764-2002 «Информационная технология. Сопровождение программных средств» (принят постановлением Госстандарта РФ от 25 июня 2002 г. N 248-ст);
4. ГОСТ Р ИСО/МЭК 90003-2014 «Разработка программных продуктов. Руководящие указания по применению ИСО 9001:2008 при разработке программных продуктов» (утвержден приказом Росстандарта от 23 октября 2014 г. N 1405-ст).

Список литературы

1. Python для гиков / [АзифМ.], пер. с англ. - СПб.: БХВ-Петербург, 2024г. - 432 с.
2. Python. Лучшие практики и инструменты. 4-е изд. / [Михаил Яворски, Тарек Зиаде]; - СПб.: Питер, 2024г. - 592 с.:
3. Python: большая книга примеров / [А. Л. Марченко]; — Москва: Издательство Московского университета, 2023г. — 361, [1] с.
4. Python и анализ данных: Первичная обработка данных с применением pandas, NumPy и Jupiter / [Уэс Маккинни]; М.: МК Пресс, 2023г. – 536 с.
5. Python. Полное руководство / [Кольцов Д. М.]; СПБ.: Издательство НАУКА и ТЕХНИКА, 2022г. - 480с.,
6. Безопасность веб-приложений на Python / [Бирн Д.], пер. с англ. С. С. Скобелева, А. Н. Киселева. – М.: ДМК Пресс, 2023г. – 334 с.
7. Основы программирования на языке Python: учеб .-метод. пособие / [Г. В. Зыкова, А. С. Попов, Т. Н. Сапуглецева]; Москва: ФЛИНТА, 2020г. – 135 с.

Периодические издания

1. Журнал «Вестник компьютерных и информационных технологий», №2, 2023г. http://www.vkit.ru/index.php/archive-rus/1228-02-february

2. Научно-практический журнал «Программные продукты и системы» №1, 2023г. http://www.swsys.ru/index.php

Интернет-ресурсы

1. Основы Python. Научитесь думать как программист. Аллен Б. Дауни https://library.samdu.uz/files/b0c333b5613b7c1710d62e0194dfc40e_%D0%94%D0%B0%D1%83%D0%BD%D0%B8%20%D0%90.%20-%20%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D1%8B%20Python%20-%202021.pdf
2. Учебник «Основы Python с нуля» https://okpython.net/python/python_uchebnik/python_uchebnik.html?yclid=15535983162980302847