МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ТУЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Утверждено решением Ученого совета Тульского государственного университета
от «27» июня 2022г.,
протокол №15;

Ректор _____ О.А.Кравченко
Подпись

М.П.

Дополнительная профессиональная программа (программа профессиональной переподготовки)

«Программирование на языке Python»
(наименование программы)

дополнительное профессиональное образование
(подвид дополнительного образования)

Тула 2022 год

I. Общие положения

1. Дополнительная профессиональная программа (программа профессиональной переподготовки) ИТ-профиля *«Программирование на языке Python»* (далее – Программа) разработана в соответствии с нормами Федерального закона РФ от 29 декабря 2012 года № 273-ФЗ «Об образовании в Российской Федерации», с учетом требований приказа Минобрнауки России от 1 июля 2013 г. № 499 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам», с изменениями, внесенными приказом Минобрнауки России от 15 ноября 2013 г. № 1244 «О внесении изменений в Порядок организации и осуществления образовательной деятельности по дополнительным профессиональным программам, утвержденный приказом Министерства образования и науки Российской Федерации от 1 июля 2013 г. № 499»; паспорта федерального проекта «Развитие кадрового потенциала ИТ-отрасли» национальной программы «Цифровая экономика Российской Федерации»; постановления Правительства Российской Федерации от 13 мая 2021 г. № 729 «О мерах по реализации программы стратегического лидерства «Приоритет-2030» (в редакции постановления Правительства Российской Федерации от 14 марта 2022 г. № 357 «О внесении изменений в постановление Правительства Российской Федерации от 13 мая 2021 г. № 729»); приказа Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации от 28 февраля 2022 г. № 143 «Об утверждении методик расчета показателей федеральных проектов национальной программы «Цифровая экономика Российской Федерации» и признании утратившими силу некоторых приказов Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации об утверждении методик расчета показателей федеральных проектов национальной программы «Цифровая экономика Российской Федерации» (далее – приказ Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации № 143); федеральных государственных образовательных стандартов высшего образования (далее вместе –

ФГОС ВО)), а также профессионального стандарта *«Программист», утвержденного приказом Министерства труда и социальной защиты РФ от 18 ноября 2013 г. № 679н*.

2. Профессиональная переподготовка заинтересованных лиц (далее – Слушатели), осуществляемая в соответствии с Программой (далее – Подготовка), имеющей отраслевую направленность «Информационно-коммуникационные технологии», проводится в Федеральном государственном бюджетном образовательном учреждении высшего образования «Тульский государственный университет» (далее – Университет) в соответствии с учебным планом в *очно-заочном форме обучения с применением дистанционных образовательных технологий*.

3. Разделы, включенные в учебный план Программы, используются для последующей разработки календарного учебного графика, учебно-тематического плана, рабочей программы, оценочных и методических материалов. Перечисленные документы разрабатываются Университетом самостоятельно, с учетом актуальных положений законодательства об образовании, законодательства в области информационных технологий и смежных областей знаний ФГОС ВО и профессионального стандарта *«Программист», утвержденного приказом Министерства труда и социальной защиты РФ от 18 ноября 2013 г. № 679н.*

4. Программа регламентирует требования к профессиональной переподготовке в области разработки алгоритмов и программ, пригодных для практического применения.

Срок освоения Программы составляет *296 академических часов*.

К освоению Программы в рамках проекта допускаются лица:

- получающие высшее образование по очной (очно-заочной) форме, лица, освоившие основную профессиональную образовательную программу (далее – ОПОП ВО) бакалавриата – в объеме не менее первого курса (бакалавры 2-го курса), ОПОП ВО специалитета – не менее первого и второго курсов (специалисты 3-го курса), а также магистратуры, обучающиеся по ОПОП ВО, не отнесенным к ИТ-сфере.

5. Область профессиональной деятельности 06, Связь, информационные и коммуникационные технологии.

II Цель программы профессиональной переподготовки

6. Целью подготовки слушателей по Программе является получение компетенции, необходимой для выполнения нового вида профессиональной деятельности в области информационных технологий обучающихся по специальностям и направлениям подготовки, не отнесенным к ИТ; приобретение новой квалификации «Программист» на языке Python.

III. Характеристика новой квалификации и связанных с ней видов профессиональной деятельности, трудовых функций и (или) уровней квалификации

7. Виды профессиональной деятельности, трудовая функция, указанные в профессиональном стандарте по соответствующей должности программиста представлены в таблице 1:

Таблица 1

Характеристика новой квалификации, связанной с видом профессиональной деятельности и трудовыми функциями в соответствии с профессиональным стандартом «Программист»

Область профессиональной деятельности	Тип задач профессиональной деятельности	Код и наименование профессиональной компетенции	Трудовые действия	Трудовая функция	Обобщенная трудовая функция	Вид профессиональной деятельности
06, Связь, информационные и коммуникационные технологии	Проектный	ПК -1. Способен разрабатывать и отлаживать программный код.	- Составление формализованных описаний решений поставленных задач в соответствии с требованиями технического задания или других принятых в организации нормативных документов. - Разработка алгоритмов решения поставленных задач в соответствии с требованиями технического задания или других принятых в организации нормативных документов.	Формализация и алгоритмизация поставленных задач (A/01.3).	Разработка и отладка программного кода (A)	Разработка программного обеспечения
			- Создание программного кода в соответствии с техническим заданием (готовыми спецификациями).	Написание программного кода с использованием языков программирования, определения и манипулирования данными (A/02.3).		

			- Комментирование и разметка программного кода в соответствии с установленными в организации требованиями. - Структурирование исходного программного кода в соответствии с установленными в организации требованиями. - Приведение наименований переменных, функций, классов, структур данных и файлов в соответствие с установленными в организации требованиями. - Форматирование исходного программного кода в соответствии с установленными в организации требованиями.	Оформление программного кода в соответствии с установленными требованиями (A/03.3).		
			- Регистрация изменений исходного текста программного кода в системе контроля версий. - Сохранение сделанных изменений программного кода в соответствии с регламентом контроля версий.	Работа с системой контроля версий (A/04.3).		
			- Анализ и проверка исходного программного кода. - Отладка программного кода на уровне программных модулей. - Отладка программного кода на уровне межмодульных взаимодействий и взаимодействий с окружением.	Проверка и отладка программного кода (A/05.3)		
06, Связь, информационные и коммуникационные технологии	Производственно-технологический.	ПК -2. Способен выполнять проверку работоспособности и осуществлять рефак-	- Подготовка тестовых наборов данных в соответствии с выбранной методикой.	Разработка тестовых наборов данных (B/02.4).	Проверка работоспособности и рефакторинг кода программного	Разработка программного обеспечения

		торинг кода программного обеспечения.	- Проверка работоспособности программного обеспечения на основе разработанных тестовых наборов данных. - Оценка соответствия программного обеспечения требуемым характеристикам. - Сбор и анализ полученных результатов проверки работоспособности программного обеспечения.	Проверка работоспособности программного обеспечения (B/03.4).	обеспечения (B)	

Таблица 2

Характеристика новой и развиваемой цифровой компетенции в ИТ-сфере, связанной с уровнем формирования и развития в результате освоения Программы «Программирование на языке Python»

Наименование сферы	**Код и наименование профессиональной компетенции**	**Пример инструментов**	**0 — способность не проявляется/ проявляется в степени, недостаточной для отнесения к 1 уровню сформированности компетенции**	**1 — способность проявляется под внешним контролем / при внешней постановке задачи/ обучающийся пользуется готовыми, рекомендованными продуктами**	**2 — способность проявляется, но обучающийся эпизодически прибегает к экспертной консультации/ самостоятельно подбирает и пользуется готовыми продуктами**	**3 — способность проявляется системно / обучающийся модифицирует способность под определенные задачи / создает новый продукт, обучает других**
Средства программной разработки	**ПК-1.** Применяет языки программирования для решения профессиональ-	Python,	не применяет	Применяет языки программирования (в т.ч. скрипты) для	Самостоятельно применяет языки программирования	Управляет процессом использования

	ных задач			решения профессиональных задач под контролем более опытных специалистов	(в т.ч. скрипты) и настраиваемые программные инструменты для автоматизации процессов в профессиональной деятельности	языков программирования (в т.ч. скриптов) и настраиваемых программных инструментов для автоматизации процессов в профессиональной деятельности организации. Обучает других
Средства программной разработки	**ПК-2.** Применяет принципы и основы алгоритмизации	Вычислительные алгоритмы, диалоговые, графические, обработки данных, управления объекта-ми/процессам и и т.д	Владеет базовыми принципами и основами алгоритмизации	Разрабатывает типовые алгоритмы под контролем опытных наставников	Самостоятельно разрабатывает алгоритмы любой сложности, использует доступный опыт других разработчиков (интернет, литература)	Применяет принципы и основы алгоритмизации системно на экспертном уровне. Контролирует программную разработку в части применения и эффективности использования алгоритмов. Обучает других
Средства программной разработки	**ПК-3.** Применяет СУБД	MySQL, MS SQL	Не применяет СУБД	Участвует в проектах по созданию ПО с использованием СУБД под контролем опытных специалистов	Участвует в проектах по созданию ПО с использованием СУБД. Разрабатывает отдельные модули ПО	На экспертном уровне применяет СУБД. Контролирует выбор, разворачивание и настройку, использование СУБД. Занимается вопросами скорости

						и оптимизации запросов. Обучает других
Средства программной разработки	**ПК-4.** Применяет интегрированные среды разработки (IDE)	PyCharm	Не применяет IDE. Использует в рамках стандартного функционала (написание кода, компиляция приложений)	Применяет IDE. Использует инструменты отладки и проверку синтаксиса под контролем опытных специалистов	Применяет IDE. Использует встроен-ные средства проверки кода	На экспертном уровне применяет IDE. Контролирует выбор, разворачивание и настройку, использование IDE. Обучает других
Прикладные программные комплексы и системы	**ПК-5.** Применяет системы контроля версий	Git	Не применяет системы контроля версий	Применяет под контролем базовый функционал систем контроля версий в части подключения к репозиторию и ведения совместной разработки	Применяет самостоятельно систе-мы контроля версий в ча-сти использования дополнительного функционала с ветвлениями	Применяет системно на экспертном уровне. Контролирует применение и эффективность систем контроля версий. Отвечает за использование инструментария распределенной разработки. Обучает других

IV. Характеристика новых и развиваемых цифровых компетенций, формирующихся в результате освоения программы

8. В ходе освоения Программы Слушателем приобретаются следующие профессиональные компетенции:

ПК -1. Способен разрабатывать и отлаживать программный код.

ПК -2. Способен выполнять проверку работоспособности и осуществлять рефакторинг кода программного обеспечения.

9. В ходе освоения Программы Слушателем совершенствуются следующие профессиональные компетенции:

- ПК-1. Применяет языки программирования для решения профессиональных задач.
- ПК-2. Применяет принципы и основы алгоритмизации.
- ПК-3. Применяет СУБД.
- ПК-4. Применяет интегрированные среды разработки (IDE).
- ПК-5. Применяет системы контроля версий.

V. Планируемые результаты обучения по ДПП ПП

10. Результатами подготовки слушателей по Программе является получение компетенции, необходимой для выполнения нового вида профессиональной деятельности в области информационных технологий «Разработка программного обеспечения»; приобретение новой квалификации «Программист» на языке Python.

11. В результате освоения Программы слушатель должен:

Знать:

- методы и приемы формализации задач;
- языки формализации функциональных спецификаций;
- методы и приемы алгоритмизации поставленных задач;

- нотации и программные продукты для графического отображения алгоритмов;
- алгоритмы решения типовых задач, области и способы их применения;
- синтаксис выбранного языка программирования, особенности программирования на этом языке, стандартные библиотеки языка программирования;
- методологии разработки программного обеспечения;
- методологии и технологии проектирования и использования баз данных;
- технологии программирования;
- особенности выбранной среды программирования и системы управления базами данных;
- компоненты программно-технических архитектур, существующие приложения и интерфейсы взаимодействия с ними;
- инструментарий для создания и актуализации исходных текстов программ;
- методы повышения читаемости программного кода;
- системы кодировки символов, форматы хранения исходных текстов программ;
- нормативные документы, определяющие требования к оформлению программного кода;
- возможности используемой системы контроля версий и вспомогательных инструментальных программных средств;
- методы и приемы отладки программного кода;
- типы и форматы сообщений об ошибках, предупреждений;
- современные компиляторы, отладчики и оптимизаторы программного кода;
- правила, алгоритмы и технологии создания тестовых наборов данных;
- методы и средства проверки работоспособности программного обеспечения;
- среду проверки работоспособности и отладки программного обеспече-

ния.

Уметь:

- использовать методы и приемы формализации задач;
- использовать методы и приемы алгоритмизации поставленных задач;
- использовать программные продукты для графического отображения алгоритмов;
- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования и средства системы управления базами данных;
- использовать возможности имеющейся технической и/или программной архитектуры;
- применять нормативные документы, определяющие требования к оформлению программного кода;
- применять инструментарий для создания и актуализации исходных текстов программ;
- применять имеющиеся шаблоны для составления технической документации;
- использовать выбранную систему контроля версий;
- использовать вспомогательные инструментальные программные средства для обработки исходного текста программного кода;
- выполнять действия, соответствующие установленному регламенту используемой системы контроля версий;
- выявлять ошибки в программном коде;
- применять методы и приемы отладки программного кода;
- применять современные компиляторы, отладчики и оптимизаторы программного кода;
- разрабатывать и оформлять контрольные примеры для проверки работоспособности программного обеспечения;
- разрабатывать процедуры генерации тестовых наборов данных с задан-

ными характеристиками;

- подготавливать наборы данных, используемых в процессе проверки работоспособности программного обеспечения;
- применять методы и средства проверки работоспособности программного обеспечения;
- анализировать значения полученных характеристик программного обеспечения.

Иметь навыки:

- составления формализованных описаний решений поставленных задач в соответствии с требованиями технического задания или других принятых в организации нормативных документов;
- разработки алгоритмов решения поставленных задач в соответствии с требованиями технического задания или других принятых в организации нормативных документов;
- создания программного кода в соответствии с техническим заданием (готовыми спецификациями);
- комментирования и разметки программного кода в соответствии с установленными в организации требованиями;
- структурирования исходного программного кода в соответствии с установленными в организации требованиями;
- приведения наименований переменных, функций, классов, структур данных и файлов в соответствие с установленными в организации требованиями;
- форматирования исходного программного кода в соответствии с установленными в организации требованиями;
- регистрации изменений исходного текста программного кода в системе контроля версий;
- сохранения сделанных изменений программного кода в соответствии с регламентом контроля версий;
- анализа и проверки исходного программного кода;
- отладки программного кода на уровне программных модулей;

- отладки программного кода на уровне межмодульных взаимодействий и взаимодействий с окружением;
- подготовки тестовых наборов данных в соответствии с выбранной методикой;
- проверки работоспособности программного обеспечения на основе разработанных тестовых наборов данных;
- оценки соответствия программного обеспечения требуемым характеристикам;
- сбора и анализа полученных результатов проверки работоспособности программного обеспечения.

VI. Организационно-педагогические условия реализации ДПП

12. Реализация Программы должна обеспечить получение компетенции, необходимой для выполнения нового вида профессиональной деятельности в области информационных технологий «Разработка программного обеспечения»; приобретение новой квалификации «Программист» на языке Python.

13. Учебный процесс организуется с применением электронного обучения, дистанционных образовательных технологий, инновационных технологий и методик обучения, способных обеспечить получение слушателями знаний, умений и навыков в области 06, Связь, информационные и коммуникационные технологии

14. Реализация Программы обеспечивается научно-педагогическими кадрами Университета, допустимо привлечение к образовательному процессу высококвалифицированных специалистов ИТ-сферы и/или дополнительного профессионального образования в части, касающейся профессиональных компетенций в области создания алгоритмов и программ, пригодных для практического применения, с обязательным участием представителей профильных организаций-работодателей. Возможно привлечение региональных руководителей цифровой трансформации (отраслевых ведомственных и/или корпоративных) к

проведению итоговой аттестации, привлечение работников организаций реального сектора экономики субъектов Российской Федерации.

VII. Учебный план ДПП

15. Объем Программы составляет *296 часов*.

16. Учебный план Программы определяет перечень, последовательность, общую трудоемкость разделов и формы контроля знаний.

Учебный план программы профессиональной переподготовки
«Программирование на языке Python»

№ п/п	Наименование раздела (модуля)	Общая трудоемкость (*296* часов)	Форма контроля
1.	Основы алгоритмизации и программирования. Язык программирования Python: синтаксис и семантика, особенности.	28	Оценка выполнения практических заданий.
2.	Работа с системой контроля версий при разработке программного обеспечения.	21	Тестирование.
3.	Технологии программирования. Объектно-ориентированное программирование на языке Python.	35	Оценка выполнения практических заданий.
4.	Базы данных и Python, работа с данными.	28	Оценка выполнения практических заданий.
5.	Возможности языка Python для разработки веб-приложений	28	Оценка выполнения практических заданий.
6.	Экосистема языка программирования Python. Области применения.	28	Оценка выполнения практических заданий.
7.	Промежуточная аттестация	42	Ассесмент, тестирование.
8.	*Практика/стажировка*	*72*	Оценка выполнения заданий/кейсов.
9.	Итоговая аттестация	14	Оценка итоговой работы
	Итого:	**296**	

VIII. Календарный учебный график

18. Календарный учебный график представляет собой график учебного процесса, устанавливающий последовательность и продолжительность обучения и итоговой аттестации по учебным дням.

Календарный учебный график программы профессиональной переподготовки

«Программирование на языке Python»

№ п п	Наименование раздела	Учебные недели																																	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
1.	Основы алгоритмизации и программирования. Язык программирования Python: синтаксис и семантика, особенности.																																				
2.	Работа с системой контроля версий при разработке программного обеспечения.																																				
3.	Технологии программирования. Объектно-ориентированное																																				

	программирование на языке Python.																																						
4.	Базы данных и Python, работа с данными.																																						
5.	Возможности языка Python для разработки веб-приложений																																						
6.	Экосистема языка программирования Python. Области применения																																						
7.	Промежуточная аттестация																																						
8.	Практика/стажировка																																						
9.	Итоговая аттестация																																						

IX. Рабочая программа учебных предметов, курсов, дисциплин (модулей)

19. Рабочая программа содержит перечень разделов и тем, а также рассматриваемых в них вопросов с учетом их трудоемкости.

Рабочая программа разрабатывается Университетом с учетом профессионального стандарта «*Программист», утвержденного приказом Министерства труда и социальной защиты РФ от 18 ноября 2013 г. № 679н.*

№ п/п	Наименование и краткое содержание раздела(модуля)	Объем, часов
1.	*Основные темы:* **Основы адгоритмизации и программирования. Язык программирования Python: синтаксис и семантика, особенности.** *Краткое содержание:* Основы алгоритмизации и программирования. Обзор инструментальных средств разработки программного обеспечения. Общий обзор языка программирования Python, его назначение и место среди других языков и систем программирования. Основные алгоритмы, функции, библиотеки и приложения языка программирования Python. Модели разработки программ. Структурное программирование. Базовые принципы, типовые структуры управления, конструкции данных. Подпрограммы (функции) как основные блоки кода. Описание функций в Python. Тестирование программного кода. Оптимизация программ и программного кода.	28
2.	*Основные темы:* **Работа с системой контроля версий при разработ-ке программного обеспечения.** *Краткое содержание:* Обзор существующих систем контроля версий. История развития. Классификации. Основные возможности системы контроля версий, понятия: репозиторий, ветка, коммит, форк, пул и пуш, мастер, кодревью. Основы Git. Ключевые особенности Git. Репозиторий Git. Работа с ветками.	21
3.	*Основные темы:* **Технологии программирования. Объектно-ориентированное программирование на языке Python.** *Краткое содержание:* Концепции, принципы и методы реализации. Классы и объекты, алгоритмы сортировки и поиска. Тестирование объектно-ориентированных приложений, инструменты тестирования.	35
4.	*Основные темы:* **Базы данных и Python, работа с данными.** Краткое содержание: Записи данных. Работа с данными СУБД. Хранение данных вне программы. Работа с данными JSON в Python.	28
5.	*Основные темы:* **Возможности языка Python для разработки веб-приложений.** *Краткое содержание:* Введение в Django, Flask, Pyramid, Web2Py, Turbogears и т.п.. Основные модели, представления, шаблоны, работа с формами. Аутентификация и авторизация, сессии, тестирование и отладка, безопасность.	28
6.	*Основные темы:* **Экосистема языка программирования Python. Области применения.** *Краткое содержание:* Установка и настройка программного обес-	28

	печения по работе с библиотеками. Обзор популярных библиотек для языка Python и основные принципы работы с ними. Первичная обработка данных. Статистика и анализ. Возможности использование языка Python для решения задач в профессиональной сфере.	
7.	*Основные темы:* **Промежуточная аттестация** *Краткое содержание:* Оценка выполненных заданий (кейсов), прохождение тестирования по модулям программы.	42
8.	*Основные темы:* **Практика/стажировка** *Краткое содержание:* Выполнение «живых» кейсов от ИТ-предприятий (партнеров).	*72*
9.	*Основные темы:* **Итоговая аттестация** *Краткое содержание:* Оценка выполнения итогового задание. Тестирование по материалам программы.	14

20. Учебно-тематический план Программы определяет тематическое содержание, последовательность разделов и (или) тем и их трудоемкость.

№ п/п	Наименование раздела(модуля)	Количество часов		
		аудиторных		самостоятельной работы (выполнение практических заданий и изучение дополнительного материла по программе)
		Лекции	Семинары	
1.	Основы алгоритмизации и программирования. Язык программирования Python: синтаксис и семантика, особенности.	4	14	10
2.	Работа с системой контроля версий при разработке программного обеспечения.	2	8	11
3.	Технологии программирования. Объектно-ориентированное программирование на языке Python.	6	18	11
4.	Базы данных и Python, работа с данными.	4	14	10
5.	Возможности языка Python для разработки веб-приложений	4	14	10
6.	Экосистема языка программирования Python. Области применения.	4	14	10
7.	**Промежуточная аттестация**	42		
8.	**Практика/стажировка**	*72*		
9.	**Итоговая аттестация**	14		

X. Формы аттестации

21. При планировании процедуры итоговой аттестации обучающихся целесообразно использовать соответствующие методические рекомендации Минобрнауки России (Письмо Минобрнауки России от 30 марта 2015 г. «О направлении методических рекомендаций по итоговой аттестации слушателей»).

Слушатели, успешно выполнившие все элементы учебного плана, допускаются к итоговой аттестации.

Итоговая аттестация по Программе проводится в форме защиты итоговой работы .

22. Лицам, успешно освоившим Программу (в области создания алгоритмов и программ, пригодных для практического применения, или навыков использования и освоения цифровых технологий, необходимых для выполнения нового вида профессиональной деятельности) и прошедшим итоговую аттестацию в рамках проекта «Цифровые кафедры», выдается документ о квалификации: диплом о профессиональной переподготовке.

При освоении ДПП ПП параллельно с получением высшего образования диплом о профессиональной переподготовке выдается не ранее получения соответствующего документа об образовании и о квалификации (за исключением лиц, имеющих среднее профессиональное или высшее образование).

23. Лицам, не прошедшим итоговую аттестацию или получившим на итоговой аттестации неудовлетворительные результаты, а также лицам, освоившим часть Программы и (или) отчисленным из Университета, выдается справка об обучении или о периоде обучения по образцу, самостоятельно устанавливаемому Университетом.

XI. Оценочные материалы

24. Контроль знаний, полученных слушателями при освоении разделов (модулей) Программы, осуществляется в следующих формах:

- текущий контроль успеваемости – обеспечивает оценивание хода освоения разделов Программы, проводится в формах оценки выполнения практиче-

ских заданий, тестирование;

- промежуточная аттестация – завершает изучение отдельного модуля Программы, проводится в форме ассесмента и (или) тестирования;

- итоговая аттестация – завершает изучение всей программы.

25. В ходе освоения Программы каждый слушатель выполняет следующие отчетные работы:

№ п/п	Наименование раздела (модуля)	Задание	Критерии оценки (0 – практическое задание не выполнено. 1- выполнено практическое задание с полной помощью наставника. 2- выполнено практическое задание с эпизадичной консультацией наставником. 3- выполнено практическое задание самостоятельно.)
1.	Основы алгоритмизации и программирования. Язык программирования Python: синтаксис и семантика, особенности.	Выполнение практических заданий по модулю (см. п.26)	Зачтено (1-3) /не зачтено (0)
2.	Работа с системой контроля версий при разработке программного обеспечения.	Прохождение тестирования по модулю (см. п.27)	Тестирование (зачтено/не зачтено)
3	Технологии программирования. Объектно-ориентированное программирование на языке Python.	Выполнение практических заданий по модулю (см. п.28)	Зачтено (1-3) /не зачтено (0)
4	Базы данных и Python, работа с данными.	Выполнение практических заданий по модулю (см. п.29)	
5	Возможности языка Python для разработки веб-приложений	Выполнение практических заданий по модулю (смп.30)	
6.	Экосистема языка программирования Python. Области применения.	Выполнение практических заданий по модулю	

		(см. п.31)	
7.	**Промежуточная аттестация**	Промежуточное тестирование по модулям (см. пп.32-37), ассесмент.	Тестирование (зачтено/не зачтено), прохождение ассесмента.
8.	**Практика/стажировка**	«Живые кейсы» от ИТ-компаний	Зачтено (1-3) /не зачтено (0)
9.	**Итоговая аттестация**	Выполнение итогового задания по программе (см. п.38)	Итоговая работа (зачтено/не зачтено). Определяется уровень сформированности компетенций по программе (0-3).

26. Текущий контроль. Перечень примерных практических заданий Модуль «Основы алгоритмизации и программирования. Язык программирования Python: синтаксис и семантика, особенности»

Практическое задание: Добавьте каждое предложение по смыслу.

1. Программа Python называется …
2. Расширение файла Python – as. …
3. Переменная в Python – это …
4. Регистр букв в идентификаторах значение …
5. Выражение в Python – это …
6. Символ # в Python обозначает …
7. … в Python это тип данных для вещественных чисел, встроенный в Python по-умолчанию.
8. Операция 3**4 - это
9. 345 - … тип данных.
10. Операция 46%10 – это …
11. Функция round(d) – это …
12. Функция input() – предначначена для …
13. Для вывода данных есть функция в Python - …
14. … в Python это логический тип данных, встроенный в Python по-умолчанию.
15. Строки – это …
16. A=’pri’, s=’vet’. A+s – это …
17. E=’no’. E*5 – это …
18. К элементу в строке можно обратиться по …
19. s='asdfgh'
print(s[-1]). Программа выведет …
20. s='asdfgh'
print(s[2:4]). Программа выведет …
21. Функция len(строка) – возвращает …
22. Списки – это …
23. Пример списка - …
24. Словари – это …
25. Пример словаря - …
26. Условный оператор в Python - …
27. Цикл for называется циклом …
28. Переведите конструкцию языка

```
S=[1,2,3]
for  I in s:
    print(I*4)
```

29. Функция range() переводится как …

30. Переведите конструкцию языка

```
S=0
While S<10:
    print(S)
    S=S+1
```

31. Функции — это …
32. Локальные переменные объявлены …

Практическое задание:
1. Создать произвольный список
2. Добавить новый элемент типа str в конец списка
3. Добавить новый элемент типа int на место с индексом
4. Добавить новый элемент типа list в конец списка
5. Добавить новый элемент типа tuple на место с индексом
6. Получить элемент по индексу
7. Удалить элемент
8. Найти число повторений элемента списка

Практическое задание:
1. Создать произвольный словарь
2. Добавить новый элемент с ключом типа str и значением типа int
3. Добавить новый элемент с ключом типа кортеж(tuple) и значением типа список(list)
4. Получить элемент по ключу
5. Удалить элемент по ключу
6. Получить список ключей словаря

Практическое задание:
1. Создать множество(set)
2. Создать неизменяемое множество(frozenset)
3. Выполнить операцию объединения созданных множеств
4. Выполнить операцию пересечения созданных множеств

Практическое задание:
Создать функцию calc(a, b, operation). Описание входных параметров:
1. Первое число
2. Второе число
3. Действие над ними:
   1) + Сложить
   2) - Вычесть
   3) * Умножить
   4) / Разделить
   5) В остальных случаях функция должна возвращать "Операция не поддерживается"

Практическое задание:
Дан список lst = [11, 5, 8, 32, 15, 3, 20, 132, 21, 4, 555, 9, 20].
Необходимо вывести элементы, которые одновременно 1) меньше 30 и 2) делятся на 3 без остатка. Все остальные элементы списка необходимо просуммировать и вывести конечный результат.

Практическое задание:

Написать функцию month_to_season(), которая принимает 1 аргумент - номер месяца - и возвращает название сезона, к которому относится этот месяц.
Например, передаем 2, на выходе получаем 'Зима'.

Практическое задание:
Дано имя файла. Необходимо вывести его расширение.

Практическое задание:
Замените заданное количество вхождений подстроки.

Практическое задание:
Изучите содержимое и импортируйте модуль this.py

27. Текущий контроль. Перечень примерных тестовых заданий Модуль «Работа с системой контроля версий при разработке программного обеспечения».

1. Проект, в котором была инициализирована система Git, называется …
a) репозиторием;
b) корневой папкой проекта;
c) хранилищем.

2. Рабочая область, в которой находятся все файлы и папки, необходимые для его работы называется …
a) репозиторием;
b) корневой папкой проекта;
c) хранилищем.

3. Содержимое скрытой папки `.git`. называется…
a) репозиторием;
b) корневой папкой проекта;
c) хранилищем.

4. В системе контроля версий Git можно сохранить текущее состояние проекта. Какая специальная команда для этого используется?
a) commit
b) git
c) save

5. Git – это распределённая и децентрализованная система управления версиями файлов?
a) нет:
b) да.

6. Если работа над проектом ведётся в команде, то перед тем как начать писать код, нужно получить последнюю версию проекта. Для этого нужно выполнить команду …
a) Pull
b) Push
c) Git

7. Чтобы отправить коллегам последнюю версию проекта выполняем команду
a) Pull

b) Push
c) Git

8. Как называется сайт, сервис и то самое облако, в котором можно хранить удалённые репозитории и через которое коллеги могут синхронизировать свои версии проектов?
a) GitHub
b) GUIHub
c) Hub

28. Текущий контроль. Перечень примерных практических заданий Модуль «Технологии программирования. Объектно-ориентированное программирование на языке Python».

Практическое задание:
Описание классовой структуры
Есть Алфавит, характеристиками которого являются:
1. Язык
2. Список букв

Для Алфавита можно:
1. Напечатать все буквы алфавита
2. Посчитать количество букв

Так же есть Английский алфавит, который обладает следующими свойствами:
1. Язык
2. Список букв
3. Количество букв

Для Английского алфавита можно:
1. Посчитать количество букв
2. Определить, относится ли буква к английскому алфавиту
3. Получить пример текста на английском языке
Задание
Класс Alphabet
1. Создайте класс Alphabet
2. Создайте метод __init__(), внутри которого будут определены два динамических свойства: 1) lang - язык и 2) letters - список букв. Начальные значения свойств берутся из входных параметров метода.
3. Создайте метод print(), который выведет в консоль буквы алфавита
4. Создайте метод letters_num(), который вернет количество букв в алфавите

Класс EngAlphabet
1. Создайте класс EngAlphabet путем наследования от класса Alphabet
2. Создайте метод __init__(), внутри которого будет вызываться родительский метод __init__(). В качестве параметров ему будут передаваться обозначение языка(например, 'En') и строка, состоящая из всех букв алфавита(можно воспользоваться свойством ascii_uppercase из модуля string).
3. Добавьте приватное статическое свойство __letters_num, которое будет хранить количество букв в алфавите.
4. Создайте метод is_en_letter(), который будет принимать букву в качестве параметра и определять, относится ли эта буква к английскому алфавиту.
5. Переопределите метод letters_num() - пусть в текущем классе классе он будет возвращать значение свойства __letters_num.

6. Создайте статический метод example(), который будет возвращать пример текста на английском языке.

Тесты:
1. Создайте объект класса EngAlphabet
2. Напечатайте буквы алфавита для этого объекта
3. Выведите количество букв в алфавите
4. Проверьте, относится ли буква F к английскому алфавиту
5. Проверьте, относится ли буква Щ к английскому алфавиту
6. Выведите пример текста на английском языке

Практическое задание:
Описание классовой структуры
Есть Помидор со следующими характеристиками:
1. Индекс
2. Стадия зрелости(стадии: Отсутствует, Цветение, Зеленый, Красный)

Помидор может:
1. Расти (переходить на следующую стадию созревания)
2. Предоставлять информацию о своей зрелости

Есть Куст с помидорами, который:
1. Содержит список томатов, которые на ней растут

И может:
1. Расти вместе с томатами
2. Предоставлять информацию о зрелости всех томатов
3. Предоставлять урожай

И также есть Садовник, который имеет:
1. Имя
2. Растение, за которым он ухаживает

И может:
1. Ухаживать за растением
2. Собирать с него урожай
Задание
Класс Tomato:
1. Создайте класс Tomato
2. Создайте статическое свойство states, которое будет содержать все стадии созревания помидора
3. Создайте метод __init__(), внутри которого будут определены два динамических protected свойства: 1) _index - передается параметром и 2) _state - принимает первое значение из словаря states
4. Создайте метод grow(), который будет переводить томат на следующую стадию созревания
5. Создайте метод is_ripe(), который будет проверять, что томат созрел (достиг последней стадии созревания)

Класс TomatoBush
1. Создайте класс TomatoBush
2. Определите метод __init__(), который будет принимать в качестве параметра количество томатов и на его основе будет создавать список объектов класса Tomato. Данный список будет храниться внутри динамического свойства tomatoes.

3. Создайте метод grow_all(), который будет переводить все объекты из списка томатов на следующий этап созревания
4. Создайте метод all_are_ripe(), который будет возвращать True, если все томаты из списка стали спелыми
5. Создайте метод give_away_all(), который будет чистить список томатов после сбора урожая

Класс Gardener
1. Создайте класс Gardener
2. Создайте метод __init__(), внутри которого будут определены два динамических свойства: 1) name - передается параметром, является публичным и 2) _plant - принимает объект класса Tomato, является protected
3. Создайте метод work(), который заставляет садовника работать, что позволяет растению становиться более зрелым
4. Создайте метод harvest(), который проверяет, все ли плоды созрели. Если все - садовник собирает урожай. Если нет - метод печатает предупреждение.
5. Создайте статический метод knowledge_base(), который выведет в консоль справку по садоводству.

Тесты:
1. Вызовите справку по садоводству
2. Создайте объекты классов TomatoBush и Gardener
3. Используя объект класса Gardener, поухаживайте за кустом с помидорами
4. Попробуйте собрать урожай
5. Если томаты еще не дозрели, продолжайте ухаживать за ними
6. Соберите урожай

Практическое задание:
Описание классовой структуры
Есть Человек, характеристиками которого являются:
1. Имя
2. Возраст
3. Наличие денег
4. Наличие собственного жилья

Человек может:
1. Предоставить информацию о себе
2. Заработать деньги
3. Купить дом

Также же есть Дом, к свойствам которого относятся:
1. Площадь
2. Стоимость

Для Дома можно:
1. Применить скидку на покупку

Также есть Небольшой Типовой Дом, обязательной площадью 40м2.
Задание
Класс Human

1. Создайте класс Human.
2. Определите для него два статических поля: default_name и default_age.
3. Создайте метод __init__(), который помимо self принимает еще два параметра: name и age.

Для этих параметров задайте значения по умолчанию, используя свойства default_name и default_age. В методе __init__() определите четыре свойства: Публичные - name и age. Приватные - money и house.
4. Реализуйте справочный метод info(), который будет выводить поля name, age, house и money.
5. Реализуйте справочный статический метод default_info(), который будет выводить статические поля default_name и default_age.
6. Реализуйте приватный метод make_deal(), который будет отвечать за техническую реализацию покупки дома: уменьшать количество денег на счету и присваивать ссылку на только что купленный дом. В качестве аргументов данный метод принимает объект дома и его цену.
7. Реализуйте метод earn_money(), увеличивающий значение свойства money.
8. Реализуйте метод buy_house(), который будет проверять, что у человека достаточно денег для покупки, и совершать сделку. Если денег слишком мало - нужно вывести предупреждение в консоль. Параметры метода: ссылка на дом и размер скидки

Класс House

1. Создайте класс House
2. Создайте метод __init__() и определите внутри него два динамических свойства: _area и _price. 3. Свои начальные значения они получают из параметров метода __init__()
4. Создайте метод final_price(), который принимает в качестве параметра размер скидки и возвращает цену с учетом данной скидки.

Класс SmallHouse

1. Создайте класс SmallHouse, унаследовав его функционал от класса House
2. Внутри класса SmallHouse переопределите метод __init__() так, чтобы он создавал объект с площадью 40м2

Тесты

1. Вызовите справочный метод default_info() для класса Human()
2. Создайте объект класса Human
3. Выведите справочную информацию о созданном объекте (вызовите метод info()).
4. Создайте объект класса SmallHouse
5. Попробуйте купить созданный дом, убедитесь в получении предупреждения.
6. Поправьте финансовое положение объекта - вызовите метод earn_money()
7. Снова попробуйте купить дом
8. Посмотрите, как изменилось состояние объекта класса Human

29. Текущий контроль. Перечень примерных практических заданий Модуль «Базы данных и Python, работа с данными».

Практическое задание (реализация с использованием Python): Задача БД Сотрудники, расчет зарплаты.

Требуется создать таблицу people («Сотрудники» с полями :первичный ключ, имя, фамилия, возраст, адрес, телефон. Данные вводятся через безопасный прием. Вывод и обращение к данным ведется с помощью класса PYMYSQL ; применяется цикл и обращение к данным по именам столбцов.

Практическое задание (реализация с использованием Python): Задача БД Автомашины

Поля: Id INT, Name TEXT, sila int, Price INT, colors, kuz int
Первичный ключ, марка, мощность, цена, цвет, коэффициент уценки
Создать и выполнить запрос: вывести данные по всем машинам
Создать и выполнить запрос: вывести данные по марке машишы.

Создать и выполнить запрос: список автомобилей с уценкой.
Создать и выполнить запрос: список автомобилей цена которых менее 600000 р.
Создать и выполнить запрос: список автомобилей с мощностью менее 150 лс.

Практическое задание (реализация с использованием Python): Задача БД Гостиницы

Создать 2 таблицы : директора и гостиницы
Поля БД директора dir(idd integer , fam text,syty text,telefon text ,kategor integer,lizens text)")
Поля БД гостиницы(gost(idg integer , name text, shmest integer,zena3 integer,zena2 integer,zena1 integer)")
(перв. Ключ, наименование, число мест, цены по категориям

Создать и выполнить запрос: Директора гостиниц
Создать и выполнить запрос: гостиницы и реквизиты

Практическое задание (реализация с использованием Python): Задача БД Склад
Поля таблицы- поставщики TABLE postawsh(id integer , name_last text, syty text, telefon text, fact text)")

Поля таблицы склад TABLE sklad(sid integer ,towar text,kpostaw integer,kolish integer ,zena integer)")

Создать и выполнить запрос: **данные по поставщикам**

postawsh.id,postawsh.name_last,sklad.sid,sklad.towar,sklad.kolish,sklad.zena

Создать и выполнить запрос: **данные по складу sid, towar, kolish, zena,**

30. Текущий контроль. Перечень примерных практических заданий Модуль «Возможности языка Python для разработки веб-приложений».

Практическое задание: Создание веб-приложения на языке Python с использованием Django.
Практическое задание: Создание веб-приложения на языке Python с использованием Flask.
Практическое задание: Создание веб-приложения на языке Python с использованием Pyramid.
Практическое задание: Создание веб-приложения на языке Python с использованием Web2Py.
Практическое задание: Создание веб-приложения на языке Python с использованием Turbogears.

31. Текущий контроль. Перечень примерных практических заданий Модуль «Экосистема языка программирования Python. Области применения».

Практическое задание:

1. Найдите пакет http(относится к Стандартной библиотеке), изучите модули, из которых он состоит.
2. Импортируйте модуль client из пакета http
3. Воспользуйтесь функционалом импортированного модуля:
   1. Создайте HTTP соединение по адресу www.google.com
   2. Отправьте GET запрос по адресу выше
   3. Проверьте ответ на отправленный запрос

Практическое задание:

Выполните следующие действия:
1. Изучите сайт репозитория пакетов PyPI (https://pypi.org/)

2. Запустите менеджер пакетов pip, изучите уже установленные модули.

3. Создайте виртуальное окружение через консоль.

4. Активируйте созданное окружение, установите любой из пакетов (например, requests), деактивируйте окружение.

5. Создайте виртуальное окружение через PyCharm.

6. Активируйте созданное окружение, установите пакет wget.

7. Воспользуйтесь функционалом пакета wget:

1) Импортируйте данный пакет

2) Скачайте с его помощью файл

Практическое задание:

Стандартная библиотека. Модуль calendar.

Выполните следующие действия:

1. Изучите справку для модуля calendar
2. Импортируйте модуль calendar
3. Найдите расположение файла модуля calendar и изучите его содержимое
4. Получите список доступных атрибутов модуля calendar
5. С помощью модуля calendar узнайте, является ли 2027 год високосным
6. С помощью модуля calendar узнайте, каким днем недели был день 25 июня 1995 года
7. Выведите в консоль календарь на 2023 год

Практическое задание:

Репозиторий PyPI. Пакет FuzzyWuzzy.

Пакет FuzzyWuzzy - это библиотека для нечеткого сравнения строк. Нечеткое сравнение строк позволяет не просто сказать - одинаковые строки или нет, а определить степень их похожести. В текущем задании предлагаем вам поработать с данной библиотекой:

1. Изучите документацию для пакета fuzzywuzzy на https://pypi.org/
2. Установите его с помощью менеджера пакетов pip
3. Определите, какие модули включает пакет fuzzywuzzy
4. Изучите модуль для базового сравнения строк fuzz(входит в пакет), получите список доступных атрибутов.
5. Изучите синтаксис метода для базового нечеткого сравнения строк ratio() (входит в состав модуля fuzz).

Прим.: Данный метод возвращает индекс схожести 2 срок

6. Воспользуйтесь методом ratio() для сравнения следующих срок:

1) 'Плохой код на самом деле не плохой.' и 'Его просто не так оптиимизировали.'

2) 'Работает? Протестируй.' и 'Работает? Оптимизируй.'

Практическое задание:

Написать программу, которая выводит на экран столбиковую диаграмму, представляющую оптовые и розничные цены на различные наименования кофе. Исходные данные сформировать в текстовом файле. Построение диаграммы оформить в виде процедуры. Параметры процедуры: количество наименований; массив значений оптовых цен; массив значений розничных цен; массив наименований. Наименования товаров разместить вертикально под осью абсцисс.

Практическое задание:

Написать программу, которая выводит на экран столбиковую диаграмму, представляющую максимальную и среднюю норму прибыли при реализации различных сортов шоколада. Исходные данные сформировать в текстовом файле самостоятельно. Построение диа-

граммы оформить в виде процедуры. Параметры процедуры: количество наименований; массив значений оптовых цен; массив значений розничных цен; массив наименований. Наименования товаров разместить вертикально под осью абсцисс

Практическое задание:

Написать программу, которая выводит на экран столбиковую диаграмму, представляющую максимальную и среднюю норму прибыли при реализации различных сортов шоколада. Исходные данные сформировать в текстовом файле самостоятельно. Построение диаграммы оформить в виде процедуры. Параметры процедуры: количество наименований; массив значений оптовых цен; массив значений розничных цен; массив наименований. Наименования товаров разместить вертикально под осью абсцисс

Практическое задание:

Написать программу, которая выводит на экран трехмерную столбиковую диаграмму курса немецкой марки по отношению к рублю за заданное количество дней. Исходные данные сформировать в текстовом файле самостоятельно. Построение диаграммы оформить в виде процедуры. Параметры процедуры: массив дат; количество дней; массив значений по оси Y; код заполнителя.

32. Промежуточный контроль. Перечень примерных тестовых заданий Модуль «Основы алгоритмизации и программирования. Язык программирования Python: синтаксис и семантика, особенности»

1. Тип переменной во время выполнения скрипта определяется по следующим правилам:

a) Тип переменной явно указывается при определении переменной и не изменяется в процессе выполнения скрипта.

b) Тип переменной явно указывается при определении переменной и изменяется только при приведении этой переменной к другому типу.

c) Тип переменной определяется типом первого значения, которое было ей присвоено, и далее не изменяется.

d) Тип переменной изменяется при приведении этой переменной к другому типу, а также может изменяться в зависимости от контекста использования этой переменной.

e) Тип переменной изменяется при присваивании, но не может изменяться в зависимости от контекста использования этой переменной.

2. Python поддерживает следующие простые скалярные типы:

a) Целое.
b) Беззнаковое целое.
c) Булево (логическое).
d) Число с плавающей точкой.
e) Строка.

3. Какие характеристики можно отнести к языку программирования Python?

a) Для быстрой разработки приложений.
b) Богатый и громоздкий синтаксис.
c) Поощряет повторное использование кода.

4. Какие характеристики можно отнести к языку программирования Python?

a) Программы на Python транслируются в машинные коды, которые затем исполняются.
b) Python использует промежуточный код.
c) Язык Python применяется для быстрой разработки приложений.

d) Python имеет обширную библиотеку стандартных модулей.

5. Что будет выведено следующей программой:

```
a = "A"
b = "B"
b = b + a
print a + b
```

a) сообщение об ошибке в третьей строке
b) ABA
c) BA
d) AB

6. Что выведет следующая программа:

```
    S = 0
for i in range(1, 10, 2):
  if i % 2 == 0:
    S = S + i
print S»
```

a) 0
b) 1
c) 10
d) 20

7. Какого типа значение получится в результате вычисления следующего выражения (, " ")?

a) str (строка)
b) tuple (кортеж)
c) это синтаксическая ошибка
d) unicode (Unicode-строка)

8. Какими операторами можно импортировать модуль?

a) import
b) from-import
c) exec
d) imp

9. Какие виды модулей есть в Python?

a) Стандартная библиотека Python
b) Сторонние модули
c) Пользовательские модули

10. В каких каталогах Python ищет модули?

e) В каталогах, указанных в переменной окружения PATH.
f) В текущем каталоге.
g) В каталогах, указанных в списке sys.path.
h) В каталоге, в который установлены стандартные модули.

33. Промежуточный контроль. Перечень примерных тестовых заданий Модуль «Работа с системой контроля версий при разработке программного обеспечения».

1. Что такое репозиторий Git?

a) Любая директория/папка;

b) Любая папка, находящаяся внутри Git

c) Репозиторий Git представляет собой каталог файловой системы, в котором находятся файлы конфигурации репозитория, файлы журналов, хранящие операции, выполняемые над репозиторием, индекс, описывающий расположение файлов, и хранилище, содержащее собственно файлы.

d) Папка .git/ и все входящие в нее.

2. Что делает команда git status?

a) Показывает состояние проекта: кол-во untracked, deleted, new и прочих файлов, количество коммитов, на которое отличается локальная версия репозитория от удаленного и так далее.

b) Показывает имя и email нашего пользователя, а также является ли он авторизованным в системе GitHub или нет.

c) Показывает место, занимаемое репозиторием на жестком диске и кол-во выделенного под репозиторий месте.

d) Такой команды нет, есть только команда git show.

3. Что означает статус файла new в выводе команды git status?

a) Что файл только что был создан и еще не отслеживается системой Git.

b) Что файл только начал отслеживаться Git и пока не имеет истории.

c) Что файл удаляли из Git и потом восстановили командой git return.

d) Такого статуса нет, есть только статус deleted file.

4. Что такое коммит?

a) Это единица состояния проекта в Git.

b) Это результат вывода команды git diff.

c) Это обобщающее название одного из статусов файла в выводе git status: untracked, new, deleted или modified.

d) Это слово ничего не означает, его ввели только для того, чтобы путать новичков.

5. Что такое ветка в репозитории Git?

a) Это то же самое, что и коммит.

b) Это минимум два коммита с одинаковым коммит-сообщением.

c) Это разные пути развития проекта, по сути разные последовательности коммитов.

d) Это механизм изменения конкретного файла.

6. Чем отличается master и origin master?

a) Это просто два разных названия одной ветки.

b) master принадлежит локальному репозиторию, а origin master – удаленному.

c) Это две разные ветки локального репозитория.

d) Ветки origin master не существует.

34. Промежуточный контроль. Перечень примерных тестовых заданий Модуль «Технологии программирования. Объектно-ориентированное программирование на языке Python».

1 Конструктор класса задается методом с именем:

a) _new__

b) __init__

c) __construct__

d) new

e) init

f) имя конструктора совпадает с именем класса

2 Если в классе определен деструктор с двумя и более параметрами, то:

a) Будет сгенерирована ошибка, т.к. Деструктор не может иметь более одного параметра.

b) Будет сгенерировано предупреждение, и такой деструктор должен вызываться только явно.

c) Не будет сгенерировано ни предупреждения, ни ошибки; при неявном вызове деструктора значение параметра будет равно none .

d) Предупреждения не будет сгенерировано, но такой деструктор должен вызываться только явно персональные данные.

3 Если в классе определены два метода с одинаковыми именами и разными списками параметров, то:

a) При выполнении скрипта будет сгенерирована ошибка

b) Будет сгенерировано предупреждение, второе определение заменит первое

c) Не будет сгенерировано ни предупреждения, ни ошибки; второе определение заменит первое

d) Не будет сгенерировано ни предупреждения, ни ошибки; вызов того или иного метода будет зависеть от типа и количества указанных при вызове параметров

e) Будет сгенерировано предупреждение; вызов того или иного метода будет зависеть от типа и количества указанных при вызове параметров персональные данные.

4 В языке Python объектами являются:

a) Экземпляры классов и переменные.
b) Экземпляры классов, переменные и функции.
c) Экземпляры классов, классы и переменные.
d) Все типы данных.

5 Значением поля класса по умолчанию может являться…

a) Значение переменной.
b) Константа.
c) Результат вызова функции.
d) Возможность указания значений полей по умолчанию в python не предусмотрена.

6 Деструктор класса задается методом с именем:

a) __del__
b) __delete__
c) __destr__
d) __destruct__

7 В языке Python доступ через instance._class__attribute:

a) разрешен к любым атрибутам
b) разрешен к любым атрибутам, кроме помеченных специальными идентификаторами
c) разрешен к любым атрибутам, кроме атрибутов со специальными именами
d) запрещен ко всем атрибутам, кроме атрибутов со специальными именами

35. Промежуточный контроль. Перечень примерных тестовых заданий Модуль «Базы данных и Python, работа с данными».

1. Укажите, как правильно подключать внешний модуль для работы с базой данных SQLite

a) import sqlite
b) import sqlite3
c) import SQLite

d) import SQLight

2. Укажите, какая команда добавляет в таблицу базы данных новую строку(запись)

a) SELECT
b) INSERT
c) VALUES
d) DROP

3. Укажите, какая команда извлекает строки(записи) из таблицы базы данных

a) SELECT
b) INSERT
c) VALUES
d) DROP

4. Укажите, какая команда обновляет строку(запись) в базе данных

a) SELECT
b) UPDATE
c) VALUES
d) DROP

36. Промежуточный контроль. Перечень примерных тестовых заданий Модуль «Возможности языка Python для разработки веб-приложений».

1. **Какие функции выполняет SWIG?**
   a) создает интерфейсные файлы
   b) интерпретирует заголовочные файлы C/C++
   c) интерпретирует интерфейсные файлы собственного формата
   d) компилирует и компонует модуль расширения для Python

2. **Какие парадигмы и стили программирования поддерживает Python?**
   a) логистическое программирование
   b) структурный стиль
   c) модульное программирование
   d) императивное программирование

3 В какой переменной окружения передается метод запроса (GET, POST)?

a) REQUEST_METHOD
b) REQUEST_URI
c) HTTP_CONNECTION
d) QUERY_STRING

4 Как с помощью модуля smtplib создать SMTP-соединение с сервером mail.server?

a) c = smtplib.sendmail('mail.server', from_, to_, message)
b) c = smtplib.connect('mail.server')
c) c = smtplib.SMTP('mail.server')
d) c = smtplib.SMTP(); c.connect('mail.server')

5 В каком модуле нужно искать функции, помогающие тестировать программу?

a) pdb
b) profile
c) unittest
d) dictutils

6 Каким из приведенных ниже способов можно убрать из строки s пробельные символы слева и справа?

a) string.strip(s)
b) string.isspace(s)
c) string.split(s)
d) string.trim(s)

7 Какой метод больше подходит для обработки XML, если при этом происходит нелинейное изменение структуры XML-документа?

a) SAX
b) DOM
c) текстовое редактирование текста XML-документа
d) все методы одинаково подходят

8 С помощью какого регулярного выражения можно "прочитать" из строки дату в формате ГГГГ-ММ-ДД? (требуется не только сопоставить строку с регулярным выражением, но и получить данные: год, месяц, день)?

a) (\d{4})-(\d{2})-(\d{2})
b) ([0-9]{4})(-([0-9]{2})){2}
c) ([0-9]{4})(-[0-9]{2}){2}
d) ([0-9]{4})-([0-9]{2})-([0-9]{2})

9 Как использовать XML-RPC сервер из Python-программы с помощью модуля xmlrpclib?

a) создать объект-подключение вызовом ServerProxy; вызывать метод call этого объекта
b) создать объект-подключение вызовом ServerProxy; вызывать метод этого объекта, соответствующий названию зарегистрированной на удаленном сервере функции
c) вызвать функцию xmlrpclib.make_call, аргументами которой указать URL, имя вызываемой функции и аргументы

10 В пакете email для чтения и записи поля сообщения используется синтаксис:

a) доступа к атрибуту
b) доступа к элементу словаря
c) итератора
d) именованного атрибута

37. Промежуточный контроль. Перечень примерных тестовых заданий Модуль «Экосистема языка программирования Python. Области применения».

1. Какой модуль стандартной библиотеки Python позволяет работать с WWW на более низком уровне:

a) httplib
b) urlparse
c) urllib2

2. Какой класс Tkinter соответствует виджету для поля ввода?

a) Text
b) Entry
c) Label
d) Frame

3. Какой класс Tkinter соответствует виджету для надписи?

a) Text
b) Entry
c) Label
d) Frame

4. Какой класс Tkinter соответствует виджету для вывода графических примитивов?

a) Frame
b) Canvas
c) Text
d) Label

5. Рассмотрите пример ниже…

```
Следующая программа производит замену одной подстроки на другую в тексте слева, записывая текст в виджете справа. Какие ошибки
в ней допущены?
from Tkinter import *
from ScrolledText import ScrolledText

def transl():
  txt = t1.get("1.0", END).replace(e1.get(), e2.get())
  t2.delete("1.0", END)
  t2.insert(END, txt)

tk = Tk()
f = Frame(tk)
e1 = Entry(f, background="White", width=32)
b = Button(f, text=">>", command=transl)
e2 = Entry(f, background="White", width=32)
f.grid(row=0, column=0, columnspan=2)
t1 = ScrolledText(tk, background="White", width=40)
t1.grid(row=1, column=0)
t2 = ScrolledText(tk, background="White", width=40)
t2.grid(row=1, column=1)
tk.mainloop()
```

a) e1.get() и e2.get() должны быть e1.get("1.0", END) и e2.get("1.0", END)
b) соответственно некоторые виджеты не появятся в окне, так как не применены менеджеры расположения
c) функция transl() должна иметь аргумент

38. Практика/стажировка.

Решение «живых» кейсов от ИТ-предприятий (партнеров) для прохождения практики/стажировки.

Пример кейса 1: Необходимо разработать приложение формирующее бухгалтерию, в частности учет расходов в разрезе различных категорий затрат, просмотр ежемесячной статистики в виде списка. Язык программирования Python.

Категории затрат:

- Зарплата
- Транспорт
- Мобильная связь
- Интернет

Должна быть возможность редактирования/добавления/удаления

Поля затрат:

- Категория *
- Стоимость *
- Комментарий (*)

*Обязательность поля

БАЗА ДАННЫХ: Данные могут храниться в любом удобном виде, желательно использование СУБД.

СРЕДСТВА И УСЛОВИЯ ВЫПОЛНЕНИЯ: Рекомендуется использование IDE для языка Python (например, PyCharm).

Пример кейса 2: Необходимо написать поисковик по текстам документов. Данные хранятся в БД по желанию, поисковый индекс в эластике. Ссылка на тестовый массива данных в формате [csv].

Структура БД:

- id - уникальный для каждого документа;
- rubrics - массив рубрик;
- text - текст документа;
- created_date - дата создания документа.

Структура Индекса:

- iD - id из базы;
- text - текст из структуры БД.

Необходимые методы

- сервис должен принимать на вход произвольный текстовый запрос, искать по тексту документа в индексе и возвращать первые 20 документов со всем полями БД, упорядоченные по дате создания;
- удалять документ из БД и индекса по полю id.

Пример кейса 3: Необходимо разработать каталог сотрудников для компании с более чем 50,000 сотрудников.

Информация о каждом сотруднике должна храниться в базе данных и содержать следующие данные:

- ФИО;
- Должность;
- Дата приема на работу;
- Размер заработной платы;

У каждого сотрудника есть 1 начальник;

База данных должна содержать не менее 50 000 сотрудников и 5 уровней иерархий.

Пример кейса 4: Необходимо разработать приложение, позволяющее управлять учетными данными сотрудников (создавать, редактировать, просматривать список и детали, удалять).

Функциональные требования к приложению:

1. Поддержка операций управления (создания, чтения, редактирования и удаления) данными сотрудников.

Данные сотрудника определяется следующими значениями:

- Имя;
- Фамилия;
- Отчество;
- Дата рождения;
- Адрес проживания;
- Отдел;
- Поле ввода “О себе”.

Нефункциональные требования:

- Система хранения данных: любая СУБД;
- Код приложения необходимо снабдить комментариями;
- Приложение должно собираться без установки или настройки каких-либо дополни-

тельных компонент;

- Архив с результатом тестового задания должен содержать текстовый файл readme.txt с инструкцией по настройке и конфигурированию приложения (если необходимо).

39. Итоговая аттестация. Перечень примерных итоговых (комлексных) заданий по программе профессиональной переподготовки.

1. Создать программу обслуживания телефонных абонентов, в которой создается база данных, содержащая сведения о номере абонента, Ф.И.О., лицевом счете. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.
2. Простой справочник. Программа должна реализовывать простой справочник, хранящий различные данные (общие или специализированные – например – номера телефонов). Должны обеспечиваться: ввод новых данных с проверкой их корректности (в зависимости от назначения справочника), просмотр справочника, поиск необходимых данных по образцу, корректировка данных, удаление ненужных записей, сохранение данных в файле и чтение данных из файла. Справочник можно реализовать в виде консольного либо экранного приложения. Тема справочника: "студенты курса".
3. Создать программу обслуживания клиентов банка, в которой создается база данных, содержащая код клиента, лицевой счет (величина вклада, проценты по вкладу, даты и суммы вложений и изъятий в течении года). Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.
4. Разработать экранное приложение "Игра 15" (так называемые "пятнашки")
5. Создать программу обслуживания клиентов заправочной станции, в которой создается база данных, содержащая марку и номер машины, дату заправки, номер и количество отпускаемого бензина, суммы оплаты. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.
6. Используя пакет PyGame разработать простую игру.
7. Создать программу регистрации нарушителей ПДД, в которой создается база данных, содержащая марку и номер машины, дату нарушения, Ф.И.О., вид нарушения, номер квитанции, суммы оплаты. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.
8. Программа "Игра в кости". Основа – рассмотренный в учебном курсе общий механизм реализации игры в кости. Программа может быть реализована в виде консольного или экранного приложения. В случае реализации в виде консольного приложения – разработать класс автомата, реализующего нетривиальный алгоритм стратегии игры, провести анализ его стратегии по сравнению с тривиальной стратегией (выбор всегда одинакового числа бросков). При реализации в виде экранного приложения – провести сравнительный анализ различных "тривиальных" стратегий.
9. Создать программу регистрации подписчиков журнала, в которой создается база данных, содержащая Ф.И.О., название журнала, номер квитанции, сумму оплаты, срок подписки. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.
10. Простой справочник. Программа должна реализовывать простой справочник, хранящий различные данные (общие или специализированные – например – номера телефонов). Должны обеспечиваться: ввод новых данных с проверкой их корректности (в зависимости от назначения справочника), просмотр справочника, поиск необходимых данных по образцу, корректировка данных, удаление ненужных записей, сохранение данных в файле и чтение данных из файла. Справочник можно реализовать в виде консольного либо экранного приложения. Тема справочника "моя фонотека".
11. Создать программу регистрации посетителей ресторана, в которой создается база данных, содержащая номер заказа, Ф.И.О. официанта, перечень заказанных блюд, сумму оплаты. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.
12. Простой справочник. Программа должна реализовывать простой справочник,

хранящий различные данные (общие или специализированные – например – номера телефонов). Должны обеспечиваться: ввод новых данных с проверкой их корректности (в зависимости от назначения справочника), просмотр справочника, поиск необходимых данных по образцу, корректировка данных, удаление ненужных записей, сохранение данных в файле и чтение данных из файла. Справочник можно реализовать в виде консольного либо экранного приложения. Тема справочника: "расписание занятий".

13. Создать программу регистрации сотрудников, в которой создается база данных, содержащая Ф.И.О. сотрудника предприятия, номер трудовой книжки, дата трудоустройства, дата увольнения, причина увольнения, а также поощрения и наказания. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.

14. Экранное приложение "календарь".

15. Создать программу регистрации посетителей поликлиники, в которой создается база данных, содержащая Ф.И.О. больного, принимающий специалист, время приема, номер полиса. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.

16. Простой справочник. Программа должна реализовывать простой справочник, хранящий различные данные (общие или специализированные – например – номера телефонов). Должны обеспечиваться: ввод новых данных с проверкой их корректности (в зависимости от назначения справочника), просмотр справочника, поиск необходимых данных по образцу, корректировка данных, удаление ненужных записей, сохранение данных в файле и чтение данных из файла. Справочник можно реализовать в виде консольного либо экранного приложения. Тема справочника: "адресная книга".

17. Создать программу учета материалов, в которой создается база данных, содержащая перечень расходуемых материалов на стройке. В перечень входят: наименование материала, количество, Ф.И.О. отпустившего, Ф.И.О. получившего. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.

18. Экранное приложение "калькулятор". Простой калькулятор для выполнения элементарных арифметических операций (типа стандартного калькулятора MS Windows, вид – "обычный").

19. Создать программу, в которой создается база данных, содержащая информацию об экзопланетах: название, масса и размеры планеты; радиус орбиты; название звезды вокруг которой обращается планета. Программа должна предоставлять возможность просматривать, добавлять, удалять, копировать, хранить данные.

20. Простой справочник. Программа должна реализовывать простой справочник, хранящий различные данные (общие или специализированные – например – номера телефонов). Должны обеспечиваться: ввод новых данных с проверкой их корректности (в зависимости от назначения справочника), просмотр справочника, поиск необходимых данных по образцу, корректировка данных, удаление ненужных записей, сохранение данных в файле и чтение данных из файла. Справочник можно реализовать в виде консольного либо экранного приложения. Тема справочника: "моя библиотека".

XII. Материально-техническое и учебно-методическое обеспечение Программы

Для проведения лекционных занятий требуется аудитория, оборудованная настенным экраном (переносным экраном), проектором, ноутбуком или персональным компьютером с выходом в Интернет, аудиосистемой.

Для проведения практических (семинарских) занятий требуется компьютерный класс с выходом в Интернет, оснащенный полным пакетом офисных

программ, инструментальной средой разработки программного обеспечения на языке Python (например, PyCharm), СУБД (например, MySQL, MS SQL), системой контроля версий (например, Git), системой видеоконференции и LMS-система для управления обучением (например, Moodle).

Для проведения промежуточной аттестации требуется компьютерный класс с программным обеспечением для проведения тестирования.

Для проведения итоговой аттестации требуется компьютерный класс с выходом в Интернет, оснащенный полным пакетом офисных программ, инструментальной средой разработки программного обеспечения на языке Python (например, PyCharm) с фреймворками Django, Flask, Pyramid, Web2Py, Turbogears, СУБД (например, MySQL, MS SQL), системой контроля версий (например, Git), системой видеоконференции.

XIII. Список литературы

Основная литература

1. Официальный сайт Python. URL: https://www.python.org/.
2. Документация Python. URL: https://docs.python.org/3/.
3. Python Software Foundation. URL: https://ru.wikipedia.org/wiki/Python_Software_Foundation.
4. PyPI - the Python Package Index. URL: https://pypi.python.org/pypi.
5. Full Grammar specification. URL: https://docs.python.org/3/reference/grammar.html.
6. Should I use Python 2 or Python 3 for my development activity?. URL: https://wiki.python.org/moin/Python2orPython3.
7. Python2 vs Python3: различия синтаксиса. URL: http://pythonworld.ru/osnovy/python2-vs-python3-razlichiya-sintaksisa.html.
8. Установка и использование Python (углубленное описание). URL: https://docs.python.org/3/using/index.html.
9. Первое сообщение о Python. URL: http://svn.python.org/view/***checkout***/python/trunk/Misc/HISTORY.
10. Guido van Rossum on the History of Python. URL: https://www.youtube.com/watch?v=ugqu10JV7dk.
11. Guido van Rossum: The Modern Era of Python. URL: https://youtu.be/rTTFh7HOlC0.

12. PEP 404 – Python 2.8 Un-release Schedule (почему Python 2.8 никогда не будет). URL: https://www.python.org/dev/peps/pep-0404/.
13. Персона. Гвидо Ван Россум — создатель Python. URL: https://habrahabr.ru/post/315974/.

Дополнительная литература

1. Introducing Getting Started with PyCharm video tutorials. URL: http://blog.jetbrains.com/pycharm/2016/01/introducing-getting-started-with-pycharm-video-tutorials/.
2. Key Features of Wing IDE. URL: https://wingware.com/wingide/key-features.
3. PyDev Video. URL: http://www.pydev.org/video_pydev_20.html.
4. Настройка и использование Geany с Python. URL: http://habrahabr.ru/post/198468/.
5. Дронов, Владимир Django: практика создания Web-сайтов на Python / Владимир Дронов. - М.: БХВ-Петербург, 2016. - 865 с.
6. Куусинен, М.Э. Говорим по-фински. Puhutaan suomea / М.Э. Куусинен. - М.: Петрозаводск: Карелия, 2017. - 356 с.
7. Россум, Г. Язык программирования Python / Г. Россум, Ф.Л.Дж. Дрейк, Д.С. Откидач и др.. - М.: [не указано], 2020. - 578 с.
8. Форсье, Джефф Django. Разработка веб-приложений на Python / Джефф Форсье , Пол Биссекс , Уэсли Чан. - М.: Символ-плюс, 2018. - 456 с.
9. Форсье, Джефф Django. Разработка веб-приложений на Python / Джефф Форсье. - М.: Символ-плюс, 2017. - 516 с.

XIV. Лист согласования программы профессиональной переподготовки

Разработчики программы профессиональной переподготовки:

Арефьева Е.А., к.т.н., доц., доц. каф. ИБ _____ *Подпись*

Сафронова М.А., к.т.н., доц., доц. каф. ИБ _____ *Подпись*

Сычугов А.А., д.т.н., доц., зав. каф. ИБ _____ *Подпись*

Французова Ю.В. к.т.н., доц. каф. ВТ _____ *Подпись*

Согласовано:
Руководитель проекта «Цифровые кафедры» _____ *Подпись* А.А.Сычугов

Согласовано с УМУ:

Специалист по УМР _____ *Подпись* С.В. Моржова

Начальник УМУ _____ *Подпись* А.В. Моржов

Программа согласована с директором
ЦПКиП ДУКО «Студенческий офис»: _____ *Подпись* М.О. Панферова

«20» июня 2022г.

На основании Протокола №4-ЗФ (от 25.08.2022г.) заочного голосования участников рабочей группы «Информационно-коммуникационные технологии» в рамках федерального проекта «Развитие кадрового потенциала ИТ-отрасли» национальной программы «Цифровая экономика Российской Федерации» и на основании Экспертного заключения АНО «Цифровая экономика» (от 17.08.2022 г.) по оценке дополнительной профессиональной программы профессиональной переподготовки (ДПП ПП) или программ обучения по модулям ИТ-профиля в пределах основной образовательной профессиональной программы высшего образования (модуль ИТ-профиля ОПОП ВО) внести в программу «Программирование на языке Python» (утверждена решением Ученого совета Тульского государственного университета от 27.07.2022г., протокол №15) внесены изменения и дополнения от 6.09.2022 г.

Лист согласования дополнений и изменений программы профессиональной переподготовки

Разработчики программы повышения квалификации:

Арефьева Е.А., к.т.н., доц., доц. каф. ИБ _____ *Подпись*

Сафронова М.А., к.т.н., доц., доц. каф. ИБ _____ *Подпись*

Сычугов А.А. д.т.н., доц., зав.каф. ИБ _____ *Подпись*

Французова Ю.В. к.т.н., доц. каф. ВТ _____ *Подпись*

Согласовано:
Руководитель проекта «Цифровые кафедры» _____ А.А.Сычугов
Подпись, дата 6.09.22.