	МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
	Федеральное государственное бюджетное образовательное учреждение высшего образования «Благовещенский государственный педагогический университет»
	ПРОГРАММА ПОДГОТОВКИ СПЕЦИАЛИСТОВ СРЕДНЕГО ЗВЕНА Рабочая программа дисциплины

УТВЕРЖДАЮ
И.о. декана физико-математического факультета ФГБОУ ВО «БГПУ»
_____ Т.А. Мередели на
«29» декабря 2021 г

Рабочая программа учебной дисциплины

ОПЦ.04 ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Программа подготовки специалистов среднего звена по специальности 09.02.07 Информационные системы и программирование

Квалификация выпускника
Программист

Принята на заседании кафедры
информатики и методики преподавания информатики
(протокол № 5 от «29» декабря 2021 г.)

Благовещенск 2021

СОДЕРЖАНИЕ

1 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

1.1 Цель дисциплины: формирование практических навыков применения алгоритмизации вычислительных процессов и программирования, ознакомление студентов с различными парадигмами проектирования и разработки программного обеспечения, формирование общего представления об эффективности алгоритмов и начального представления об анализе эффективности программ.

1.2 Место дисциплины в структуре ООП:

Учебная дисциплина «Основы алгоритмизации и программирования» принадлежит к общепрофессиональному циклу (ОПЦ.04).

1.3 Дисциплина направлена на формирование следующих компетенций:

- ОК 1. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.
- ОК 2. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности.
- ОК 9. Использовать информационные технологии в профессиональной деятельности.
- ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием.
- ПК 1.3. Выполнять отладку программных модулей с использованием специализированных программных средств.
- ПК 1.4. Выполнять тестирование программных модулей.

1.4 Перечень планируемых результатов обучения. В результате изучения дисциплины обучающийся должен

уметь:

- разрабатывать алгоритмы для конкретных задач;
- использовать программы для графического отображения алгоритмов;
- определять сложность работы алгоритмов;
- работать в среде программирования;
- реализовывать построенные алгоритмы в виде программ на конкретном языке программирования;
- оформлять код программы в соответствии со стандартом кодирования;
- выполнять проверку, отладку кода программы;

знать:

- понятие алгоритмизации, свойства алгоритмов, общие принципы построения алгоритмов, основные алгоритмические конструкции;
- эволюцию языков программирования, их классификацию, понятие системы программирования;
- основные элементы языка, структуру программы, операторы и операции, управляющие структуры, структуры данных, файлы, классы памяти;
- подпрограммы, составление библиотек подпрограмм;
- объектно-ориентированную модель программирования, основные принципы объектно-ориентированного программирования на примере алгоритмического языка: понятие классов и объектов, их свойств и методов, инкапсуляция и полиморфизма, наследования и переопределения.

1.5 Общая трудоемкость дисциплины «Основы алгоритмизации и программирования» составляет 190 ч. максимальной учебной нагрузки обучающегося, в том числе:

обязательной аудиторной учебной нагрузки обучающегося – 152 часа; самостоятельной работы обучающегося – 30 часов.

Программа предусматривает изучение материала на лекциях и лабораторных. Предусмотрена самостоятельная работа обучающихся по темам и разделам. Программа предусматривает использование в образовательном процессе активных и интерактивных форм проведения занятий в сочетании с внеаудиторной работой. Проверка знаний осуществляется фронтально, индивидуально.

1.6 Объем дисциплины и виды учебной деятельности

Вид учебной работы	Объем часов
Максимальная учебная нагрузка (всего)	**190**
Обязательная аудиторная учебная нагрузка (всего)	**152**
в том числе:	
- лекции	66
- лабораторные занятия	86
Самостоятельная работа обучающегося (всего)	**30**
Консультации	2
Промежуточная аттестация: дифференцированный зачет, экзамен	6

2 ТЕМАТИЧЕСКИЙ ПЛАН И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

Наименование разделов и тем	Содержание учебного материала, лабораторные работы, самостоятельная работа обучающегося	Объем в часах
Раздел 1.	**Введение в программирование**	**16**
Тема 1.1. Языки программирования	**Содержание учебного материала**	4
	1. Развитие языков программирования.	
	2. Обзор языков программирования. Области применения языков программирования. Стандарты языков программирования. Среда проектирования. Компиляторы и интерпретаторы.	
	3. Жизненный цикл программы. Программа. Программный продукт и его характеристики.	
	4. Основные этапы решения задач на компьютере.	
	В том числе лабораторных работ	2
	Самостоятельная работа обучающихся	2
Тема 1.2. Типы данных	**Содержание учебного материала**	
	1. Типы данных. Простые типы данных. Производные типы данных. Структурированные типы данных.	4
	В том числе лабораторных работ	2
	Самостоятельная работа обучающихся	2
Раздел 2.	**Содержание учебного материала**	**46**
Тема 2.1. Операторы языка программирования	1. Операции и выражения. Правила формирования и вычисления выражений. Структура программы. Ввод и вывод данных. Оператор присваивания. Составной оператор.	16
	2. Условный оператор. Оператор выбора.	
	3. Цикл с постусловием. Цикл с предусловием. Цикл	

	с параметром. Вложенные циклы.	
	4. Массивы. Двумерные массивы. Строки. Стандартные процедуры и функции для работы со строками.	
	5. Структурированный тип данных – множество. Операции над множествами.	
	6. Комбинированный тип данных – запись. Файлы последовательного доступа. Файлы прямого доступа	
	В том числе лабораторных работ	24
	Самостоятельная работа обучающихся	6
Раздел 3.	**Содержание учебного материала**	**36**
Тема 3.1. Процедуры и функции	1. Общие сведения о подпрограммах. Определение и вызов подпрограмм. Область видимости и время жизни переменной. Механизм передачи параметров. Организация функций.	6
	2. Рекурсия. Программирование рекурсивных алгоритмов.	
	В том числе лабораторных работ	8
	Самостоятельная работа обучающихся	2
Тема 3.2. Структуризация в программировании	**Содержание учебного материала**	
	1. Основы структурного программирования. Методы структурного программирования.	2
	В том числе лабораторных работ	4
	Самостоятельная работа обучающихся	2
Тема 3.3. Модульное программирование	**Содержание учебного материала**	
	1. Модульное программирование. Понятие модуля. Структура модуля. Компиляция и компоновка программы.	4
	2. Стандартные модули.	
	В том числе лабораторных работ	6
	Самостоятельная работа обучающихся	2
Раздел 4	**Основные конструкции языков программирования**	**20**
Тема 4.1 Указатели.	**Содержание учебного материала**	
	1. Указатели. Описание указателей. Основные понятия и применение динамически распределяемой памяти. Создание и удаление динамических переменных.	8
	2. Структуры данных на основе указателей.	
	3. Задача о стеке.	
	В том числе лабораторных работ	10
	Самостоятельная работа обучающихся	2
Раздел 5	**Содержание учебного материала**	**64**
Тема 5.1 Основные принципы объектно-ориентированного программирования (ООП)	1. История развития ООП. Базовые понятия ООП: объект, его свойства и методы, класс, интерфейс.	4
	2. Основные принципы ООП: инкапсуляция, наследование, полиморфизм.	
	3. Классы объектов. Компоненты и их свойства.	
	4. Событийно-управляемая модель программирования. Компонентно-ориентированный подход.	
	В том числе лабораторных работ	6

	Самостоятельная работа обучающихся	2
Тема 5.2 Интегрированная среда разработчика.	**Содержание учебного материала**	
	1. Требования к аппаратным и программным средствам интегрированной среды разработчика.	4
	2. Интерфейс среды разработчика: характеристика, основные окна, инструменты, объекты. Форма и размещение на ней управляющих элементов.	
	3. Панель компонентов и их свойства. Окно кода проекта.	
	4. Состав и характеристика проекта. Выполнение проекта. Настройка среды и параметров проекта.	
	5. Панель компонентов и их свойства. Окно кода проекта. Состав и характеристика проекта. Выполнение проекта. Настройка среды и параметров проекта.	
	6. Настройка среды и параметров проекта.	
	В том числе лабораторных работ	4
	Самостоятельная работа обучающихся	2
Тема 5.3. Визуальное событийно-управляемое программирование	**Содержание учебного материала**	
	1. Основные компоненты (элементы управления) интегрированной среды разработки, их состав и назначение.	4
	2. Дополнительные элементы управления. Свойства компонентов. Виды свойств. Синтаксис определения свойств. Назначения свойств и их влияние на результат. Управление объектом через свойства.	
	3. События компонентов (элементов управления), их сущность и назначение. Создание процедур на основе событий.	
	В том числе лабораторных работ	4
	Самостоятельная работа обучающихся	2
Тема 5.4 Разработка оконного приложения	**Содержание учебного материала**	
	1. Разработка функционального интерфейса приложения. Создание интерфейса приложения.	2
	2. Разработка функциональной схемы работы приложения.	
	3. Разработка игрового приложения.	
	В том числе лабораторных работ	2
	Самостоятельная работа обучающихся	2
Тема 5.5 Этапы разработки приложений	**Содержание учебного материала**	
	1.Разработка приложения.	4
	2. Проектирование объектно-ориентированного приложения.	
	3. Создание интерфейса пользователя.	
	4. Тестирование, отладка приложения.	
	В том числе лабораторных работ	8
	Самостоятельная работа обучающихся	2
Тема 5.6 Иерархия классов.	**Содержание учебного материала**	
	1. Классы ООП: виды, назначение, свойства, методы, события.	4

	2. Перегрузка методов.	
	3. Тестирование и отладка приложения.	
	4. Решение задач	
	В том числе лабораторных работ	6
	Самостоятельная работа обучающихся	2
Примерная тематика лабораторных работ: Знакомство со средой программирования. Составление программ линейной структуры. Составление программ разветвляющейся структуры. Составление программ циклической структуры Обработка одномерных массивов. Обработка двумерных массивов. Работа со строками. Работа с данными типа множество. Файлы последовательного доступа. Типизированные файлы. Нетипизированные файлы. Организация процедур. Организация функций. Применение рекурсивных функций. Программирование модуля. Создание библиотеки подпрограмм. Использование указателей для организации связанных списков. Изучение интегрированной среды разработчика. Создание проекта с использованием компонентов для работы с текстом. Создание проекта с использованием компонентов ввода и отображения чисел, дат и времени. События компонентов (элементов управления), их сущность и назначение. Создание процедур на основе событий. Создание проекта с использованием кнопочных компонентов. Создание проекта с использованием компонентов стандартных диалогов и системы меню. Разработка функциональной схемы работы приложения. Разработка оконного приложения с несколькими формами. Разработка игрового приложения. Создание процедур обработки событий. Компиляция и запуск приложения. Разработка интерфейса приложения. Тестирование, отладка приложения. Классы ООП: виды, назначение, свойства, методы, события. Объявления класса. Создание наследованного класса. Программирование приложений. Перегрузка методов.		
Консультации		**2**
Промежуточная аттестация: дифференцированный зачет, экзамен		**2+4**
Всего:		**190**

3 УСЛОВИЯ РЕАЛИЗАЦИИ УЧЕБНОЙ ДИСЦИПЛИНЫ

3.1 Требования к минимальному материально-техническому обеспечению

Реализация учебной дисциплины требует наличия компьютерного класса – учебная аудитория для проведения всех видов учебных занятий, групповых и индивидуальных консультаций, текущего контроля, промежуточной аттестации и самостоятельной работы.

Комплект учебной мебели, компьютерные столы, аудиторная доска, компьютеры с установленным лицензионным программным обеспечением, мультимедийный проектор, экспозиционный экран, 11 персональных компьютеров.

Используемое программное обеспечение: Microsoft®WINEDUperDVC AllLng Upgrade/SoftwareAssurancePack Academic OLV 1License LevelE Platform 1Year; Microsoft®OfficeProPlusEducation AllLng License/SoftwareAssurancePack Academic OLV 1License LevelE Platform 1Year; Dr.Web Security Suite; Java Runtime Environment; Calculate Linux.

3.2 Информационное обеспечение обучения

Литература

1. Огнева, М. В. Программирование на языке C++: практический курс : учебное пособие для среднего профессионального образования / М. В. Огнева, Е. В. Кудрина. — Москва : Издательство Юрайт, 2021. — 335 с. — (Профессиональное образование). — ISBN 978-5-534-05780-5. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/473118

2. Трофимов, В. В. Основы алгоритмизации и программирования : учебник для среднего профессионального образования / В. В. Трофимов, Т. А. Павловская ; под редакцией В. В. Трофимова. — Москва : Издательство Юрайт, 2021. — 137 с. — (Профессиональное образование). — ISBN 978-5-534-07321-8. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/473347

3. Черпаков, И. В. Основы программирования : учебник и практикум для среднего профессионального образования / И. В. Черпаков. — Москва : Издательство Юрайт, 2021. — 219 с. — (Профессиональное образование). — ISBN 978-5-9916-9984-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/470969

4. Чернышев, С. А. Основы программирования на Python : учебное пособие для среднего профессионального образования / С. А. Чернышев. — Москва : Издательство Юрайт, 2021. — 286 с. — (Профессиональное образование). — ISBN 978-5-534-15160-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/487638

Базы данных и информационно-справочные системы

1. Алгоритмы и методы программирования – Режим доступа : http://algolist.manual.ru
2. Дистанционная подготовка по информатике – Режим доступа : http://informatics.mccme.ru
3. Сайт Государственного научно-исследовательского институт информационных технологий и телекоммуникаций. – Режим доступа: Informika | ВКонтакте (vk.com)
4. Федеральный портал «Российское образование» – Режим доступа: http://www.edu.ru
5. Шень, А. Программирование: теоремы и задачи, 2-е изд. – М.: МЦНМО, 2004, 296 с. – Режим доступа : http://www.mccme.ru/free-books/shen/shen-progbook.pdf

Электронно-библиотечные ресурсы

1. ЭБС «Юрайт». – Режим доступа: https://urait.ru
2. Полпред (обзор СМИ). – Режим доступа: https://polpred.com/news

4 КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Контроль и оценка результатов освоения учебной дисциплины осуществляется преподавателем в процессе проведения лекционных занятий, лабораторных занятий, опроса, а также выполнения обучающимися лабораторных работ и разноуровневых задач и заданий.

Результаты обучения (освоенные умения, усвоенные знания)	**Формы и методы контроля результатов обучения**
Умения: – разрабатывать алгоритмы для конкретных задач; – использовать программы для графического отображения алгоритмов; – определять сложность работы алгоритмов; – работать в среде программирования; – реализовывать построенные алгоритмы в виде программ на конкретном языке программирования; – оформлять код программы в соответствии со стандартом кодирования; – выполнять проверку, отладку кода программы; **Знания**: – понятие алгоритмизации, свойства алгоритмов, общие принципы построения алгоритмов, основные алгоритмические конструкции; – эволюцию языков программирования, их классификацию, понятие системы программирования; – основные элементы языка, структуру программы, операторы и операции, управляющие структуры, структуры данных, файлы, классы памяти; – подпрограммы, составление библиотек подпрограмм; – объектно-ориентированную модель программирования, основные принципы объектно-ориентированного программирования на примере алгоритмического языка: понятие классов и объектов, их свойств и методов, инкапсуляция и полиморфизма, наследования и переопределения.	Опрос Лабораторная работа Разноуровневые задачи и задания Интерпретация результатов наблюдений за деятельностью обучающегося в процессе выполнения лабораторных работ Защита (в форме собеседования) лабораторных работ по выполнению разноуровневых задач и заданий

5 ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ПО ДИСЦИПЛИНЕ

Формируемая компетенция	Показатели освоения компетенций
ОК-1. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.	**Знать:** • актуальный профессиональный и социальный контекст, в котором приходится работать и жить; • основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; • алгоритмы выполнения работ в профессиональной и смежных областях; • методы работы в профессиональной и смежных сферах; • структуру плана для решения задач; • порядок оценки результатов решения задач профессиональной деятельности **Уметь:** • распознавать задачу и/или проблему в профессиональном и/или социальном контексте; • анализировать задачу и/или проблему и выделять её составные части; • определять этапы решения задачи; выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы; • составить план действия; определить необходимые ресурсы; • владеть актуальными методами работы в профессиональной и смежных сферах; • реализовать составленный план; • оценивать результат и последствия своих действий (самостоятельно или с помощью наставника)

Задание 1. Расположите в правильном порядке жизненные циклы программного обеспечения.

a. Разработка
b. Анализ требований
c. Техническая поддержка
d. Проектирование
e. Тестирование

Ответ: b d a e c

Задание 2. Соотнесите парадигму программирование и её описание.

a. Структурное программирование	1. Программист описывает, что должна делать программа, а не как это делать. Основной акцент на желаемом результате.
b. Декларативное программирование	2. Основное внимание уделяется функциям. Программа строится с использованием функций

	высшего порядка, функциональных композиций и избегания изменяемого состояния.
c. Функциональное программирование	3. Основной принцип – разбивать программу на небольшие, легко управляемые блоки (подпрограммы), что облегчает понимание и сопровождение кода.
d. Императивное программирование	4. Программа представляет собой набор логических утверждений, и система логического программирования заботится о том, как достичь целевого состояния на основе этих утверждений.
e. Объектно-ориентированное программирование	5. В этой парадигме программа представляет собой последовательность инструкций, которые изменяют состояние программы. Основной фокус на том, как выполнять определенные действия.
f. Логическое программирование	6. Программа рассматривается как набор объектов, которые взаимодействуют между собой. Каждый объект содержит данные и методы для их обработки.

Ответ: a – 3; b – 1; c – 2; d – 5; e – 6; f – 4.

Задание 3. _____ – специальная программа, преобразующая программный код с того или иного языка программирования в машинный код.

Ответ: Транслятор

Задание 4. Дайте определение термину «Алгоритм».

Ответ:

это точная последовательность команд, предназначенная исполнителю, в результате выполнения которой он должен решить поставленную задачу.	*3 балла* (В формулировке сказано о точности выполняемых действий, о том, кому предназначен алгоритм и о конечном результате)
Последовательность действий для получения результата	*1 балл* (определение дано не в полном объёме)

Задание 5. Опишите виды комментирования программного кода в языке Python.

Ответ:
Началом однострочных комментариев является значок хештега (#). Возможно написание заметки возле оператора кода. Используется при написании дополнительных указаний, не превышающих 70−72 символов.

```
print("Hello, World!")  # Написан текст «Привет, мир!»
```

Второй способ – многострочный текст выделяется тройными кавычками. Размещая пояснение после функции/метода, язык программирования воспримет комментарий, как часть функции/метода.

```
'''
Функция для вывода на экран «Привет, мир!»
'''
def print_hi():
    print("Hello, World!")
```

Задание 6. _____ – документ, описывающий соглашение о том, как писать код на языке Python. Документ создан на основе рекомендаций создателя: код читается намного больше раз, чем пишется. Если весь код будет написан в едином стиле, то любой сможет легко его прочесть.

Ответ: PEP8

Задание 7. Переменная в программировании это – …

a. Это шаблон или описание для создания объектов.
b. Команда для объявления функций в языках программирования.
c. Специальный тип данных для хранения текстовой информации.
d. Однозначный идентификатор, используемый для обозначения конкретного места в памяти, в котором хранится значение.

Ответ: d

Задание 8. Оператор = в программировании имеет название:

a. Оператор сравнения.
b. Оператор присваивания.
c. Оператор записи/перезаписи
d. Оператор инкремента.

Ответ: b

Задание 9. Соотнесите систему контроля версий и её описание.

a. Локальные СКВ	1. Суть их работы состоит в выгрузке клиентам не только версий файлов с последними изменениями, а всего репозитория. Различают центральный и локальные репозитории.
b. Централизованные СКВ	2. Хранит и все зарегистрированные в ней файлы, и любые вносимые в них изменения. Специально к текстовым файлам применяется алгоритм дельта-компрессии. Это означает, что система сохраняет последнюю версию и все предшествующие ей изменения.
c. Распределенные СКВ	3. Имеют собственный центральный сервер, где накапливаются все файлы проекта. Система контролирует и их сохранность, и выдачу их копий определенному ряду пользователей.
d. Copy-paste	4. Локальное хранений файлов по шаблону. Напри-

мер, filename_{version}, возможно с добавлением времени создания или изменения.

Ответ: a – 2; b – 3, c – 1, d – 4.

Формируемая компетенция	Показатели освоения компетенций
ОК-2. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности.	**Знать:** • номенклатура информационных источников, применяемых в профессиональной деятельности; • приемы структурирования информации; • формат оформления результатов поиска информации **Уметь:** • определять задачи для поиска информации; • определять необходимые источники информации; • планировать процесс поиска; • структурировать получаемую информацию; • выделять наиболее значимое в перечне информации; • оценивать практическую значимость результатов поиска; • оформлять результаты поиска.

Задание 1. _____ – это набор инструкций, описаний и примеров, которые помогают разработчикам освоить язык программирования или решить конкретные проблемы в процессе разработки. Является самым актуальным и достоверным источником информации для изучения языка программирования.

Ответ: документация языка программирования

Задание 2. Дайте определение термину «Рефакторинг»

Ответ: Рефакторинг – это переработка исходного кода программы, чтобы он стал более простым и понятным.

Задание 3. По стандарту PEP8 в одном файле можно одновременно использовать для отступов и табуляцию и пробелы

a. верно
b. неверно

Ответ: b

Задание 4. В каком документе фиксируются требования к разрабатываемой программе?

a. Документация языка программирования
b. Инструкция по эксплуатации
c. Техническое задание
d. Справочной документации

Ответ: c

Задание 5. _____ – это инструмент, используемый разработчиками программного обеспечения для управления изменениями в исходном коде и других файловых ресурсах. Она позволяет отслеживать историю изменений, возвращаться к предыдущим версиям, сливать изменения из разных источников и сотрудничать с другими разработчиками.

Ответ: Система контроля версий.

Задание 6. Распределить уровни квалификаций разработчика и его обязанности в команде.

a. Junior разработчик	1. всё делает сам: пишет код, занимается архитектурой, обучает младших коллег. Решает сложные задачи и отвечает за результаты.
b. Middle разработчик	2. руководитель команды разработчиков. Не пишет код. Занимается распределением нагрузки на команду, следит за ходом проекта и берёт на себя ответственность за проект в целом.
c. Senior разработчик	3. работает над простыми задачами под руководством опытных коллег
d. Teamlid	4. может решать сложные задачи или писать код самостоятельно, старшие коллеги проверяют только результаты;

Ответ: a – 3; b – 4; c – 1; d – 2.

Задание 7. Соотнесите термин Git и его определение.

a. Repository	1. запрос на слияние форка репозитория с основным репозиторием.
b. Fork	2. каталог файловой системы, в котором находятся: файлы конфигурации, файлы журналов операций, выполняемых над репозиторием, индекс расположения файлов и хранилище, содержащее сами контролируемые файлы.
c. Branch	3. слияние изменений из какой-либо ветки репозитория с любой веткой этого же репозитория.
d. Pull Request	4. копия репозитория.
e. Push	5. фиксация изменений или запись изменений в репозиторий.
f. Commit	6. параллельная версия репозитория.
g. Merge	7. отправка всех неотправленных коммитов на удалённый сервер репозитория.

Ответ: a – 2; b – 4; c – 6; d – 1; e – 7; f – 5; g – 3.

Задание 8. Основные принципы методологии разработки ПО «Agile».

Ответ:
Agile – подход к управлению проектами и разработке программного обеспечения, который помогает быстрее создавать качественные продукты и правильно развивать их. Такой результат становится возможным благодаря гибкости рабочих процессов и эффективному взаимодействию всех заинтересованных лиц: клиентов, заказчиков и команды проекта.

- Люди и взаимодействия важнее процессов и инструментов.
- Работающий продукт важнее точной и подробной документации.
- Сотрудничество с заказчиком важнее условий договора.
- Готовность к изменениям важнее следования изначальному плану.

Формируемая компетенция	Показатели освоения компетенций
ОК-9. Использовать информационные технологии в профессиональной деятельности.	**Знать:** • современные средства и устройства информатизации; • порядок их применения и программное обеспечение в профессиональной деятельности **Уметь:** • применять средства информационных технологий для решения профессиональных задач; • использовать современное программное обеспечение

Задание 1. _____ – это программа, в которой разработчики пишут, проверяют, тестируют и запускают код, а также ведут большие проекты. Она включает в себя сразу несколько инструментов: редактор для написания кода, сервисы для его проверки и запуска, расширения для решения дополнительных задач разработки.

Ответ: IDE / Integrated Development Environment / интегрированная среда разработки

Задание 2. Соотнесите термин с его определением.

a. Компилятор	1. программа, которая выполняет код, написанный на языке программирования. Она не переводит его в машинные коды целиком, а построчно принимает команды и сразу выполняет их.
b. Интерпретатор	2. читает всю программу целиком, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется. Результат работы – бинарный исполняемый файл.

Ответ: a – 2; b – 1;

Задание 3. Что такое «Отладчик (Debugger)»?

Ответ:

программа, предназначенная для анализа поведения другой программы, обеспечивающая ее трассировку (отслеживание и распечатку выполняемых программой ко-	*3 балла*

манд, изменений переменных или данных о других событиях, связанных с выполнением программы), останов в указанных точках или, при выполнении указанных условий.	
Программа для обнаружения ошибок в коде.	*1 балл* (определение дано не в полном объёме)

Задание 4. Соотнесите язык программирования и его парадигму.

a. Haskell	1. Функциональное программирование
b. Prolog	2. Объектно-ориентированное программирование
c. C++	3. Структурное программирование
d. Python	4. Логическое программирование
e. Pascal	5. Мультипарадигменное программирвоание

Ответ: a – 1; b – 4; c – 2; d – 5; e – 3.

Задание 5. Дайте определение термину «Функция».

Ответ:
Функция – это именованный блок кода, к которому можно обратиться из лю-бого места программы. У функции есть имя и список входных парамет-ров, а также воз-вращаемое значение.

Задание 6. Рекурсия в программировании – это …

a. Процесс, при котором программа выполняет одну и ту же функцию или процедуру внутри самой себя.
b. Тип данных, представляющий набор элементов с одинаковыми значениями.
c. Алгоритм, использующий математические операции для решения сложных задач.
d. Метод создания объектов в объектно-ориентированных языках программирования.

Ответ: a

Задание 7.
_____ – это любые данные, которыми можно характеризовать объект класса.
_____ – это блок кода, состоящий из ряда инструкций, который можно вызывать по его имени.

Ответ: Свойство класса, метод класса

Задание 8. Соотнесите принцип ООП с его описанием.

a. Абстракция	1. подразумевает возможность нескольких реализаций одной идеи
b. Полиморфизм	2. способность одного класса расширять понятие другого, и главный механизм повторного использования кода в ООП
c. Наследование	3. блокирует доступ к деталям сложной концепции, подразумевает возможность рассмотреть объект с общей точки зре-

ния

d. Инкапсуляция	4. выделение основных, наиболее значимых характеристик объекта и игнорирование второстепенных.

Ответ: a – 4; b – 1; c – 2; d – 3.

Задание 9. _____ – это процесс изменения кода, призванный упростить его обслуживание, понимание и расширение, при этом не изменяя его поведение.

Ответ: Рефакторинг

Задание 10. _____ – это процесс преобразования части кода в другую функционально эквивалентную часть для улучшения одной или более характеристик кода.

Ответ: Оптимизация

Формируемая компетенция	**Показатели освоения компетенций**
ПК-1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.	**Знать:** • Основные этапы разработки программного обеспечения. • Основные принципы технологии структурного и объектно-ориентированного программирования. • Дополнительно для квалификаций "Программист" и "Технический писатель": актуальная нормативно-правовая база в области документирования алгоритмов. **Уметь:** • Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием. • Оформлять документацию на программные средства. • Дополнительно для квалификаций "Программист" и "Технический писатель": оценка сложности алгоритма. **Практический опыт:** • Разрабатывать алгоритм решения поставленной задачи и реализовывать его средствами автоматизированного проектирования.

Задание 1. Какого сложность алгоритма «Сортировка пузырьком»?

a. $O(n^2)$
b. $O(n)$
c. $O(n!)$
d. $O(\log(n))$
e. $O(1)$

Ответ: a

Задание 2. _____ – это специальная нотация, используемая для описания асимптотической сложности; то есть, скорости роста времени выполнения алгоритма с увеличением размера входных данных.

Ответ: Big O

Задание 3. Соотнесите правила для именования и кому они принадлежат:

a. Классы	1. Нотация under_score: Слова в имени написаны с маленькой буквы, через символ _
b. Переменные	2. Нотация UNDER_SCORE: Слова в имени написаны с большой буквы, через символ _
c. Константы	3. Нотация CamelCase: Каждое слово в имени начинается с большой буквы
d. Функции/методы	4. Нотация under_score(): Слова в имени написаны с маленькой буквы, через символ _

Ответ: a – 3, b – 1, c – 2, d – 4.

Задание 4. _____ – это набор функций и инструментов, написанные другими людьми ранее, которые помогают в написании кода и расширяют возможности языка программирования.

Ответ: Библиотека

Задание 5. Реализуйте следующие классы:

- Родитель Animal принимающий на вход следующие параметры:
  - имя
  - возраст
  - необязательный параметр голос, равный по умолчанию None.
  - необязательный параметр голод, равный по умолчанию 0.

Также реализуйте метод eat, принимающий на вход «еда». В теле функции реализуйте уменьшения свойства голод в соответствии с параметром «еда», но уточните что свойство голод не должно опускаться ниже 0.

- Потомок Dog принимающий на вход следующие параметры:
  - Теже параметры что и родитель
  - Порода

Реализовать метод голос, возвращающий свойство голос и метод информация, печатающий всю информацию об объекте класса.

Ответ:

```
class Animal:
    def __init__(self, name: str, age: int, voice=None, health=0):
        self.name = name
        self.age = age
        self.voice = voice
        self.hunger = health

    def eat(self, food: int):
```

```
        self.hunger = max(sel.hunger - food, 0)

class Dog(Animal):
    def __init__(self, name: str, age: int, voice: str, breed: str, health=0):
        super().__init__(name, age, voice, health)
        self.breed = breed

    def speak(self) -> str:
        return self.voice

    def print_info(self):
        print(f'Порода - {self.breed}')
        print(f'Имя - {self.name}')
        print(f'Возраст - {self.age}')
        print(f'Голод - {self.hunger}')
```

Задание 6. Напишите программу для нахождения минимального числа в диапазоне [-100; 100], введённого с клавиатуры, числа вводятся до тех пор, пока человек не напишет 404. В случае если такого числа не нашлось, вывести 404.

Ответ:

```
n_max = 404
while (n := int(input())) != 404:
    if -100 <= n <= 100:
        n_max = min(n, n_max)
print(n_max)
```

Задание 7. Соотнесите вид цикла и ситуацию, в которой он применим.

a. while	1. Цикл используется, когда заранее неизвестно число итераций цикла
b. for	2. Цикл используется, когда заранее известно число итераций цикла

Ответ: a – 1, b – 2.

Задание 8. Итерация – это …

a. Ошибка в программе.
b. Процесс повторения блока кода или выполнения цикла.
c. Программа, выполняющая запуск кода построчно
d. Тип данных в языке программирования.
e. Форма представления алгоритма.

Ответ: b

Задание 9. def func(*args, **kwargs): в какой тип данных будут собраны аргументы в переменной **kwargs

a. string
b. tuple
c. dictionary
d. list
e. set

Ответ: c

Задание 10. def func(*args, **kwargs): в какой тип данных будут собраны аргументы в переменной *args

a. string
b. tuple
c. dictionary
d. list
e. set

Ответ: b

Задание 11. Выберете правильный пример использования списочных выражений

a. [i if i % 2 == 0 else 0 for _ in range(int(input()))]
b. [i for _ in range(int(input())) if i % 2 == 0]
c. [i for _ in range(int(input())) if i % 2 == 0 else 0]
d. [i for i in range(int(input())) if i % 2 == 0]

Ответ: d

Задание 12. Исправьте ошибку в коде, чтобы умножение переменных осуществлялось корректно.

```
a = input()
b = input()
c = a * b
```

Ответ:

```
a = int(input())
b = int(input())
c = a * b
```

ИЛИ

```
a = float(input())
b = float(input())
c = a * b
```

Задание 13. Установите строгую типизацию данных в классе, напишите описание класса «Класс животное».

```
class Animal:
    def __init__(self, name, age, health, voice):
        self.name = name
        self.age = age
        self.health = health
        self.voice = voice
```

Ответ:

```
class Animal:
    """Класс животное"""
    def __init__(self, name: str, age: int, health: int, voice: str):
        self.name = name
        self.age = age
        self.health = health
        self.voice = voice
```

Формируемая компетенция	Показатели освоения компетенций
ПК-1.2. Разрабатывать программные модули в соответствии с техническим заданием.	**Знать:** • Основные этапы разработки программного обеспечения. • Основные принципы технологии структурного и объектно-ориентированного программирования. • Дополнительно для квалификаций "Программист": знание API современных мобильных операционных систем. **Уметь:** • Создавать программу по разработанному алгоритму как отдельный модуль. • Оформлять документацию на программные средства. • Дополнительно для квалификаций "Программист": осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ. • Дополнительно для квалификации "Специалист по тестированию в области информационных технологий": осуществлять разработку модулей для различных видов тестирования. **Практический опыт:** • Разрабатывать код программного продукта на основе готовой спецификации на уровне модуля. • Дополнительно для квалификаций "Программист": разрабатывать мобильные приложения.

Задание 1. Написать алгоритм сортировки «Пузырьком».

Ответ:

```
a = [7, 0, 2, -5, -1, 9, 1]
for i in range(len(a)-1):
    for j in range(len(a)-i-1):
        if a[j] > a[j+1]:
            a[j], a[j+1] = a[j+1], a[j]
print(a)
```

Задание 2. _____ – Это интерфейс, предоставляемый программным обеспечением для взаимодействия с другими программами или компонентами.

Ответ: API

Задание 3. _____ – Это шаблон или описание для создания объектов. Он определяет свойства и методы, которые могут быть использованы объектами этого шаблона.

Ответ: Класс

Задание 4. Написать рекурсивную функцию для нахождения факториала числа.
Ответ:

```
def fac(n):
    if n == 1:
        return 1
    return fac(n - 1) * n
```

Задание 5. Что будет выведено на экран в результате выполнения программы

```
for i in range(1, 7, 2):
    print(i, end=' ')
```

a. 1 2 3 4 5 6 7 1 2 3 4 5 6 7
b. 1 3 5
c. 1 3 5 7
d. 3 5 7
e. 3 5
f. 1 2 3 4 5 6 7
g. 1 2 3 4 5 6

Ответ: b

Задание 6. Какой из magic методов отвечает за конструктор класса?

a. __init__
b. __int__
c. __call__
d. __new__
e. __repr__

Ответ: a

Задание 7. Расположите в правильном порядке приоритет выполнения операторов

a. and
b. not
c. or
d. =
e. >, <, >=, <=, ==, !=
f. *, //, /, %
g. +, -

Ответ: f, g, b, a, c, e, d

Задание 8. Соотнесите структуру данных и функцию для её инициализации

a. Список	1. dict()
b. Кортеж	2. list()
c. Множество	3. tuple()
d. Словарь	4. set()

Ответ: a – 2; b – 3; c – 4; d – 3.

Задание 9. Аналогом какой структуры данных в python является файл *.json ?

a. Кортеж
b. Множество
c. Словарь
d. Список

Ответ: c

Задание 10. Соотнесите конструкцию и её название.

a. for элемент in последовательность:	1. Объявление функции
b. while условие:	2. Условная конструкция
c. if условие:	3. Цикл пока
d. def имя(аргументы):	4. Цикл для перебора

Ответ: a – 4, b – 3, c – 2, d – 4.

Задание 11. Как вывести на экран имя переменной и её значение? Например: a=5.

a. `print('{a=}', {a})`
b. `print(f'a=')`
c. `print(f'{a=}')`
d. `print(f'{a}')`

Ответ: c

Формируемая компетенция	Показатели освоения компетенций
ПК-1.3. Выполнять отладку программных модулей с использованием специализированных программных средств.	**Знать:** • Основные принципы отладки и тестирования программных продуктов. • Инструментарий отладки программных продуктов. **Уметь:** • Выполнять отладку и тестирование программы на уровне модуля. • Оформлять документацию на программные средства. • Дополнительно для квалификаций "Программист" и "Специалист по тестированию в области информационных технологий": Применять инструментальные средства отладки программного обеспечения. **Практический опыт:** • Использовать инструментальные средства на этапе отладки программного продукта. • Проводить тестирование программного модуля по определенному сценарию.

Задание 1. Соотнесите вид отладки программы с её описанием

a. Шаг с заходом (step into)	1. Команда выполняет очередную инструкцию. Когда происходит вход в вызовы функций и выполнять их строка за строкой, происходит выполнение всей функции, не останавливаясь, и вернет управ-

	ление после ее выполнения. Есть возможность пропустить функции, если разработчик уверен, что они уже исправлены, или не заинтересован в их отладке в данный момент.
b. Шаг с обходом (step over)	2. Выполняет не следующую строку кода, а весь оставшийся код функции, исполняемой в настоящее время. После возврата из функции он возвращает управление разработчику. Эта команда полезна, когда специалист случайно вошел в функцию, которую не нужно отлаживать.
c. Шаг с выходом (step out)	3. Команда выполняет очередную инструкцию, а потом приостанавливает процесс, чтобы с помощью отладчика было можно проверить состояние программы. Если в выполняемом операторе есть вызов функции, то происходит переход в начало вызываемой функции, где она приостанавливается.

Ответ: a – 3, b – 1, c – 2.

Задание 2. _____ – набор связанных функций отладчика, позволяющих поэтапно выполнять код.

Ответ: Пошаговое выполнение

Задание 3. Отладчик помогает найти синтаксические ошибки в программном коде.

a. Верно
b. Неверно

Ответ: b

Задание 4. Напишите программу для сложения двух чисел. Используя, обработчик ошибок проверьте корректность входных данных.

Ответ:

```
try:
    a = int(input())
    b = int(input())
    print(a + b)
except ValueError:
    print("Ошибка ввода числа")
```

Задание 5. Напишите программу для деления двух чисел. Используя, обработчик ошибок проверьте корректность второго числа (не 0).

Ответ:

```
try:
    a = int(input())
    b = int(input())
    print(a / b)
```

```
except ZeroDivisionError:
    print("b не может быть нулем!")
```

Задание 6. Конструкция _____ – необходима для обработки ошибок, возникающих во время выполнения программы.

Ответ: Обработчик исключений

Задание 7. Какой тип ошибки показан на изображении:

```
fol i in range(5):
    ^
```

a. Логическая
b. Исключение
c. Синтаксическая
d. Компоновки

Ответ: c

Задание 8. Какой тип ошибки показан на изображении:

```
for i in range("5"):
          ^^^^^^^^^^
```

a. Логическая
b. Исключение
c. Синтаксическая
d. Компоновки

Ответ: b

Задание 9. Что нужно применить если: программа работает некорректно, но сообщений об ошибках нет?

a. Рефакторинг
b. Оптимизацию
c. Отладку
d. Компиляцию
e. Рекурсию

Ответ: c

Задание 10. Что такое трассировка (tracing) в отладке?

a. Метод анализа производительности программы
b. пошаговое исполнение программы с всех изменений данных
c. Инструмент для автоматической генерации тестовых случаев
d. Тип статического анализа кода.

Ответ: b

Задание 11. Оптимизируйте код используя списочные выражения. Ответ должен быть в одну строку кода.

```
arr = []
n = input()
n = int(n)
for i in range(n):
    s = input()
```

```
    s = int(s)
    arr.append(s)
print(arr)
```

Ответ: `print([int(input()) for _ in range(int(input()))])`

Задание 12. В чём отличие *Refactoring* от *code review*?

Ответ:

Code review:

Проверка уже написанного кода: обычно в команде этим занимается самый опытный разработчик.

Refactoring:

процесс изменения кода, призванный упростить его обслуживание, понимание и расширение, при этом не изменяя его поведение.

Задание 13. Что можно улучшить в процессе рефакторинга?

a. Тесты
b. Дизайн
c. Идею
d. Код

Ответ: d

Задание 14. Можно ли данную функцию оптимизировать и/или применить рефакторинг?

```
def ok(value):
    result = None
    if value == 1:
        result = True
    if value == 0:
        result = False
    return result
```

a. Можно оптимизировать
b. Можно применить рефакторинг
c. Можно проделать и то и то
d. Код хорошо написан

Ответ: c

Задание 15. _____ – это определение соответствия, разрабатываемого ПО ожиданиям и потребностям пользователя, его требованиям к системе.

a. Верификация
b. Авторизация
c. Валидация
d. Идентификация
e. Аутентификация

Ответ: c

Задание 16. К какому виду системы контроля версий относится GitHub?

a. Локальные СКВ
b. Централизованные СКВ

c. Распределенные СКВ
d. Copy-paste

Ответ: с

Формируемая компетенция	Показатели освоения компетенций
ПК-1.4. Выполнять тестирование программных модулей.	**Знать:** • Основные виды и принципы тестирования программных продуктов. • Дополнительно для квалификации "Специалист по тестированию в области информационных технологий": Методы организации работы при проведении функционального тестирования. **Уметь:** • Выполнять отладку и тестирование программы на уровне модуля. • Оформлять документацию на программные средства. • Дополнительно для квалификации "Специалист по тестированию в области информационных технологий": выполнять тестирование в соответствие с функциональными требованиями. • Выполнять оценку тестового покрытия. **Практический опыт:** • Проводить тестирование программного модуля по определенному сценарию. • Использовать инструментальные средства на этапе тестирования программного продукта. • Дополнительно для квалификации "Специалист по тестированию в области информационных технологий": проводить тестирование в соответствие с функциональными требованиями.

Задание 1. _____ – это программа, которая проверяет работу небольшой части кода.

Ответ: Юнит-тест / unit test / или модульный тест

Задание 2. Что будет выведено на экран 6 итерации цикла?

```
for i in range(10):
    if i == 3:
        break
    print(i)
```

Ответ: Цикл не дойдёт до 6 итерации, на 4 произойдёт выход из цикла.

Задание 3. Какое значение будет хранится в переменной C в конце выполнения программы.

```
c = 1
b = 0
while b < 100:
    if b % 10 == 0:
        c += 2
    b += 8
print(c)
```

Ответ: 7

Задание 4. Соотнесите методологию разработки ПО с её описанием.

a. TDD – Test Driven Development	1. ваши типы данных и сигнатуры типов являются спецификацией программы. Типы также служат формой документации, которая гарантированно обновляется. Типы представляют из себя небольшие контрольные точки, благодаря которым, мы получаем множество мини-тестов по всему нашему приложению.
b. TDD – Type Driven Development	2. разработка реального, работающего программного обеспечения систематически, в поставленные сроки. делает основной упор на коротких итерациях, каждая из которых служит для проработки определенной части функциональности системы. Одна итерация длится две недели. методология насчитывает пять процессов.
c. BDD – Behaviour Driven Development	3. методология разработки ПО, которая основывается на повторении коротких циклов разработки: изначально пишется тест, покрывающий желаемое изменение, затем пишется программный код, который реализует желаемое поведение системы и позволит пройти написанный тест. Затем проводится рефакторинг написанного кода с постоянной проверкой прохождения тестов.
d. FDD – Features Driven Development	4. предполагает описание тестировщиком или аналитиком пользовательских сценариев на естественном языке — если можно так выразиться, на языке бизнеса.

Ответ: a – 3; b – 1; c – 4, d – 2.

Задание 5. Соотнесите тип тестирования ПО с его описанием.

a. модульное тестирование	1. подход к тестированию программного обеспечения используется программистом для тестирования отдельно взятой части кода программы. Это помогает разработчикам узнать, правильно ли работает каждый блок кода в изоляции от остальных.
b. интеграционное тестирование	2. проверяет, соответствует ли программа заявленным характеристикам
c. системное тестирование	3. проверка связей между готовыми элементами, а также сочетаемости программы с окружающей средой (оборудованием и ОС)
d. Валидационное (приемочное)	4. Процесс оценки программного обес-

тестирование	печения с целью определить – удовлетворяет ли оно определенным бизнес-требованиям. Гарантирует, что продукт соответствует потребностям клиента.

Ответ: a – 1; b – 3, c – 2, d – 4.

Задание 6. Соотнесите вид тестирования ПО с его описанием.

a. статическое	1. проверка кода и документации с запуском ПО. Тестировщики оценивают, долго ли грузятся страницы, сколько оперативной памяти нужно для нормальной работы приложения
b. динамическое	2. когда тестировщик работает с интерфейсной частью продукта и видит его код;
c. функциональное	3. исследует совместимость и производительность компонентов приложения
d. нефункциональное	4. когда тестировщик работает только с интерфейсной частью продукта и не видит его код
e. тестирование «черного ящика»	5. проверка кода и документации без запуска приложения или программы
f. тестирование «белого ящика»	6. проверяет, справляется ли приложение с возложенными на него функциями и задачами

Ответ: a – 5; b – 1, c – 6, d – 3, e – 4, f – 2.

Задание 7. Какая встроенная функция в Python позволяет протестировать функцию и в случае ложного возвращённого результата, позволяет вывести в консоль строку с текстом.

a. test
b. assert
c. unittest
d. pytest
e. nose

Ответ: b

Задание 8. Какой модуль для тестирования кода имеется в стандартной библиотеке Python?

a. test
b. unittest
c. pytest
d. nose
e. unitpy

Ответ: b

Задание 9. _____ – это поиск (локализация), анализ и устранение ошибок в программном обеспечении, которые были найдены во время тестирования.

a. Рефакторинг
b. Оптимизация
c. Отладка
d. Тестирование
e. Декомпозиция

Ответ: c

Задание 10. master branch – это …

a. ветка, в которой программа в любой момент считается рабочей и хорошо протестированной. В идеале программу из этой ветки всегда можно передать заказчику
b. это ветка со всеми законченными в настоящий момент изменениями, без нестабильных изменений и изменений «в работе»
c. нужная для исправления критического бага и обратного вливания в исходную ветку
d. ветка, в которой разработчик ведет разработку одной конкретной задачи или функциональности

Ответ: a

Задание 11. Develop branch – это …

a. ветка, в которой программа в любой момент считается рабочей и хорошо протестированной. В идеале программу из этой ветки всегда можно передать заказчику
b. это ветка со всеми законченными в настоящий момент изменениями, без нестабильных изменений и изменений «в работе»
c. нужная для исправления критического бага и обратного вливания в исходную ветку
d. ветка, в которой разработчик ведет разработку одной конкретной задачи или функциональности

Ответ: b

Задание 12. Выстроите последовательность действий при работе с Git.

a. Создать ветку dev
b. Решить конфликт слияния
c. Запушить коммит
d. Переключится на dev ветку
e. Создать коммит
f. Произвести слияние веток
g. Создать репозиторий

Ответ: g, a, d, e, c, f, d

Задание 13. _____ – проверка соответствия реальных и ожидаемых результатов поведения программы, проводимая на конечном наборе тестов, выбранном определённым образом.

Ответ: Тестирование программного обеспечения

Задание 14. _____ – это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части.

Ответ: Тестовый сценарий

Задание 15. _____ – это документ, который описывает что должно быть протестировано. Чек-лист может быть абсолютно разного уровня детализации.

a. Тестовый сценарий
b. Чек-лист
c. Тест план
d. Тест-дизайн

Ответ: b

Задание 16. _____ – это документ, который описывает весь объем работ по тестированию, начиная с описания объекта, стратегии, расписания, критериев начала и окончания тестирования, до необходимого в процессе работы оборудования, специальных знаний, а также оценки рисков.

a. Тестовый сценарий
b. Чек-лист
c. Тест план
d. Тест-дизайн

Ответ: c

Задание 17. _____ – это этап тестирования ПО, на котором проектируются и создаются тестовые случаи (тест-кейсы).

a. Тестовый сценарий
b. Чек-лист
c. Тест план
d. Тест-дизайн

Ответ: d

Составитель: Алутина Е.Ф., кандидат физико-математических наук, доцент

6 ЛИСТ ИЗМЕНЕНИЙ И ДОПОЛНЕНИЙ

Утверждение изменений и дополнений в РПД для реализации в 2023/2024 уч. г.

РПД обсуждена и одобрена для реализации в 2023/2024 уч. г. на заседании кафедры информатики и методики преподавания информатики (протокол №9 от 26 июня 2023 г.).