Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Гриб Владислав Валерьевич
Должность: Ректор
Дата подписания: 23.09.2026 19:10:44
Уникальный программный ключ:
637517d24e103c3db032acf37e839d98ec1c5bb2f5eb89c29abfcd7f43985447

Образовательное частное учреждение высшего образования

«МОСКОВСКИЙ УНИВЕРСИТЕТ ИМЕНИ А.С. ГРИБОЕДОВА»

(ИМПЭ им. А.С. Грибоедова)

МЕЖДУНАРОДНЫЙ ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И БИЗНЕС-ИНФОРМАТИКИ

УТВЕРЖДАЮ
Директор Международного института информационных технологий и бизнес-информатики
_____ Г.В. Новикова
«17» декабря 2025г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по учебной дисциплине:
WEB-ПРОГРАММИРОВАНИЕ

Направление подготовки
09.03.03 Прикладная информатика
(уровень бакалавриат)

Направленность (профиль):
«Анализ данных»

Форма обучения: **очная, заочная**

Москва

Фонд оценочных средств для дисциплины «Web-программирование». Направление подготовки/специальность 09.03.03 Прикладная информатика, направленность (профиль/специализация): Анализ Данных/ И.В. Торицын – М.: ИМПЭ им. А.С. Грибоедова.

Фонд оценочных средств является неотъемлемой частью рабочей программы дисциплины.

Разработчик: И.В. Торицын

Заведующий кафедрой: _____/ Н. Н. Загускин

1. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

Настоящий Фонд оценочных средств (ФОС) по дисциплине «Web-программирование» является неотъемлемым приложением к рабочей программе дисциплины (РПД) «Web-программирование». На данный ФОС распространяются все реквизиты утверждения, представленные в РПД по данной дисциплине.

2. ПЕРЕЧЕНЬ ОЦЕНОЧНЫХ СРЕДСТВ

Для определения качества освоения обучающимися учебного материала по дисциплине используются следующие оценочные средства:

№ п/п	Оценочное средство	Краткая характеристика оценочного средства	Представление оценочного средства в ФОС
1	Тестирование	Вид контроля, позволяющий оценить изученный теоретический материал.	Вопросы для проведения тестирования
2	Практические задания	Вид контроля, позволяющий оценить умение обучающегося применять осваиваемую компетенцию в практических ситуациях и при решении производственных задач	Задания к практическому занятию
3	Контрольная работа	Вид контроля, позволяющий определить результат освоения компетенций по дисциплине в рамках рассматриваемой темы, оцениваемый с помощью соответствующих индикаторов достижения компетенций	Задания контрольной работы
4	Самостоятельная работа	Вид контроля, позволяющий оценить проработку теоретического материала, изучение рекомендуемой литературы, выполнение практико-ориентированных заданий (заполнение таблиц, проведение сравнительного анализа, составление схем и др.), решение практических задач, создание презентаций, написание рефератов, подборку нормативного и иного материала и выполнение других заданий	Задания самостоятельной работы
5	Курсовая работа	Вид контроля, позволяющий выявить степень владения базовыми знаниями, умениями и навыками, необходимыми для обучения, и определить уровень владения новым материалом	Индивидуальные задания (темы) для курсовой работы
6	Зачет/Зачет с оценкой/Экзамен	Вид контроля, позволяющий выявить степень овладения знаниями, умениями и навыками, необходимыми для дальнейшего освоения образовательной программы подготовки	Вопросы для подготовки к зачету/зачету с оценкой/экзамену

3. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ ДИСЦИПЛИНЕ

3.1. Сопроводительная информация.

Разработчик	И.В. Торицын
Кафедра	Прикладной информатики и

	информационных технологий
Наименование дисциплины	Web-программирование
Факультет / институт	Институт международной экономики, лидерства и менеджмента
Направление подготовки / специальность	09.03.03 Прикладная информатика
Количество вопросов в оценочных заданиях (диапазон)	От 10 до 20
Общее время тестирования (мин)	90
Общее количество вопросов/заданий в ФОС	20
Размещенность на сайте Университета примерного перечня вопросов, заданий ФОС – для подготовки обучающихся к прохождению оценки (да / нет)	Да

3.1. Модели контролируемых компетенций:

Код компетенции	Формулировка компетенции	Индикаторы достижения компетенции
ПК-7	Способен осуществлять проектирование программных интерфейсов	**ИПК-7.1** Знать методы и средства проектирования программных интерфейсов **ИПК-7.2** Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов

4. СОДЕРЖАНИЕ ОЦЕНОЧНЫХ СРЕДСТВ ТЕКУЩЕГО КОНТРОЛЯ

4.1. ТИПОВЫЕ ТЕСТОВЫЕ ЗАДАНИЯ.

Тесты содержат набор вопросов, в полном объеме охватывающие изученный теоретический материал по указанной теме. Выполнение тестов позволяет определить результат освоения компетенций по дисциплине в рамках рассматриваемой темы, оцениваемый с помощью соответствующих индикаторов достижения компетенций. Индивидуальный тестовый сеанс для каждого обучающегося формируется по специальному алгоритму, обеспечивающему заданную тематическую структуру и пропорциональное наличие вопросов разного типа и сложности.

При формировании тестов необходимо использовать задания следующих типов:

Тип задания 1. Задания закрытого типа на установление соответствия.

Тип задания 2. Задания закрытого типа на установление последовательности.

Тип задания 3. Задания комбинированного типа, предполагающие выбор одного правильного ответа из предложенных

Тип задания 4. Задания комбинированного типа, предполагающие выбор нескольких ответов из предложенных

Тип задания 5. Задания открытого типа с развернутым ответом.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
Тема 1.1 Основы PHP. Управляющие конструкции	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 1.1** Какие из следующих утверждений о типах данных и операторах в PHP являются верными? Варианты ответов: 1. Оператор === сравнивает значения с приведением типов. 2. Результатом выражения (int) '12abc' будет целое число 12. 3. Константу, объявленную с помощью define(), можно переопределить в той же области видимости. 4. Выражение $a = 0; $b = '0'; при проверке if ($a == $b) вернёт true. Ответ: 2, 4. Пояснение: Правильный (2): При приведении строки, начинающейся с цифр, к целому типу (int), PHP берёт числовую часть с начала строки. Правильный (4): Оператор == выполняет сравнение с приведением типов. Строка '0' и число 0 при таком сравнении равны. **Задание 1.2** Какие из следующих фрагментов кода приведут к выводу на экран всех элементов ассоциативного массива $data = ['name' => 'Ivan', 'age' => 25];? Варианты ответов: 1. php for ($i = 0; $i < count($data); $i++) { echo $data[$i]; } 2. php foreach ($data as $value) { echo $value; } 3. php foreach ($data as $key => $value) { echo "$key: $value"; } 4. php do { echo current($data); } while (next($data)); Ответ: 2, 3. Пояснение: Правильный (2): Цикл foreach корректно перебирает значения ассоциативного массива. Правильный (3): Цикл foreach корректно перебирает ключи и значения ассоциативного массива.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			Задание 1.3 Установите правильную последовательность выполнения скрипта PHP при запросе через веб-сервер (например, Apache). Шаги: 1. Интерпретатор PHP выполняет код, встречая управляющие конструкции (условия, циклы) и манипулируя данными. 2. Веб-сервер получает HTTP-запрос и определяет, что запрошенный файл имеет расширение .php. 3. Результат выполнения (обычно HTML) отправляется веб-сервером обратно клиенту в HTTP-ответе. 4. Веб-сервер передаёт файл и данные запроса интерпретатору PHP. 5. Интерпретатор PHP парсит файл, начиная с первого символа, и выходит из режима парсера при закрывающем теге ?>. Ответ: 2, 4, 5, 1, 3 Пояснение: Интерпретатор PHP не работает напрямую с HTTP-запросами — он получает уже подготовленные данные от веб-сервера и возвращает ему готовый результат для отправки клиенту. **Задание 1.4** Расположите этапы обработки следующей конструкции switch в порядке их выполнения интерпретатором PHP. php switch ($value) { case 10: echo 'Десять'; break; case 20: echo 'Двадцать'; break;

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			default: echo 'Другое'; } Этапы: 1. Сравнение $value со значением второго case (20) на строгое равенство (===). 2. Выполнение кода в секции default, если ни один case не совпал. 3. Выполнение кода в секции case 10: до оператора break. 4. Сравнение $value со значением первого case (10) на строгое равенство (===). Ответ: 4312 Пояснение: Оператор switch проверяет case-ы последовательно сверху вниз. При совпадении выполняется соответствующий блок кода до первого встреченного break (или до конца switch). Блок default выполняется только если не сработал ни один case. **Задание 1.5** Установите соответствие между управляющей конструкцией и её наиболее типичным применением. Список 1 (Конструкция): A. if...elseif...else B. switch C. for D. foreach Список 2 (Применение): 1. Перебор элементов массива без необходимости управления счётчиком. 2. Выбор одного блока кода для выполнения из множества вариантов на основе одного выражения. 3. Выполнение блока кода определённое, заранее известное число раз. 4. Выполнение разных блоков кода в зависимости от сложных логических условий. Правильное соответствие: A4, B2, C3, D1.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме

			Задание 1.6
			Установите соответствие между фрагментом кода и результатом его выполнения (значением переменной $result).
			
			Список 1 (Код):
			A. $result = 1;
			while ($result < 10) {
			$result *= 2;
			}
			
			B. $result = '';
			for ($i = 0; $i < 3; $i++) {
			$result .= $i;
			}
			
			C. $arr = [5, 10, 15];
			$result = 0;
			foreach ($arr as $v) {
			if ($v == 10) continue;
			$result += $v;
			}
			
			D. $result = 7 >= 7 ? 'Да' : 'Нет';
			
			Список 2 (Результат):
			1. '012'
			2. 20
			3. 16
			4. 'Да'
			
			Ответ: A3, B1, C2, D4.

Тема 1.2 Пользовательские	ПК- 7 Способен осуществлять	ИПК-7.1 Знать методы и средства	**Задание 2.1.** Установите правильную последовательность этапов, которые выполняет

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме

функции	проектирование программных интерфейсов	проектирования программных интерфейсов	интерпретатор PHP при вызове пользовательской функции.
			
			Этапы:
			1. Выполнение кода функции: вычисление выражений, работа с аргументами, выполнение управляющих конструкций.
			2. Возврат управления (и значения, если есть return) в точку вызова функции.
			3. Объявление функции (её код) считывается и сохраняется в памяти при первой встрече в процессе парсинга скрипта, но не выполняется.
			4. Передача управления в тело функции: создание локальной области видимости, инициализация аргументов переданными значениями.
			5. Встреча вызова функции (functionName(...)) в исполняемом коде.
			
			Правильная последовательность: 35412
			
			Пояснение: PHP работает в два прохода: сначала парсит весь скрипт (объявления), затем выполняет код последовательно. Функция не выполняется в момент объявления, а лишь «подготавливается» к будущим вызовам.
			
			Задание 2.2.
			Расположите шаги в правильном порядке для получения результата «Сумма: 30» из данного фрагмента кода.
			
			php
			function calculateSum($numbers) {
			$total = 0;
			foreach ($numbers as $num) {
			$total += $num;
			}
			return $total;
			}
			
			$data = [10, 20];
			// ... дальнейшие действия ...
			
			Шаги:

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			1. Вызов функции calculateSum с аргументом $data. 2. Объявление функции calculateSum. 3. Присвоение возвращённого значения переменной, например, $result. 4. Вывод на экран строки "Сумма: " и значения переменной $result. 5. Создание массива $data со значениями [10, 20]. Правильная последовательность: 25134 Пояснение: PHP работает в два прохода: сначала парсит весь скрипт (объявления), затем выполняет код последовательно. Функция не выполняется в момент объявления, а лишь «подготавливается» к будущим вызовам. **Задание 2.3** Какое ключевое слово используется для передачи аргумента в функцию PHP таким образом, чтобы изменения его значения внутри функции отражались на оригинальной переменной, переданной при вызове? Варианты ответов: 1. ref 2. & (амперсанд) в объявлении параметра 3. pointer 4. global Ответ: 2. Пояснение: Амперсанд &, поставленный перед именем параметра в объявлении функции (function foo(&$bar)), указывает, что аргумент должен передаваться по ссылке. Это единственный корректный встроенный механизм языка для такого поведения. **Задание 2.4** Что может вернуть функция в PHP?

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			Варианты ответов: 1. Только скалярные значения (целое число, строку и т.д.). 2. Ровно одно значение любого типа или ничего (NULL). 3.Неограниченное количество значений через запятую. 4& Только массивы, если нужно вернуть несколько значений. Ответ: 2. Пояснение: Оператор return возвращает ровно одно значение (любого типа: массив, объект, ресурс и т.д.) или NULL, если return отсутствует или вызван без аргумента. **Задание 2.5** Как называется механизм в PHP, который позволяет функции принимать переменное количество аргументов, и как называется встроенная функция для получения этих аргументов внутри неё? Правильный ответ: Variadic function (или "функция с переменным числом аргументов"), для получения используется функция func_get_args() (или оператор ... (spread operator) в объявлении). Пояснение: Современный (PHP 5.6+): Использование оператора ... (spread/ellipsis operator) в списке параметров (function sum(...$numbers)), который собирает все переданные аргументы в массив $numbers. **Задание 2.6** Какое ключевое слово, добавленное перед переменной внутри функции, заставляет её сохранять своё значение между последовательными вызовами этой же функции? Правильный ответ: static Пояснение: Статическая переменная существует только в локальной области видимости функции, но не теряет своего значения, когда выполнение программы выходит из этой области. Инициализация (static $count = 0;) происходит только при первом вызове функции.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
Тема 1.3 Встроенные функции	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 3.1.** Установите правильную последовательность приоритета (от высшего к низшему) источников данных для суперглобального массива $_REQUEST при настройках PHP по умолчанию (request_order = "GP" в php.ini). Источники данных: 1. Данные из файлов, загруженных через $_FILES (метод multipart/form-data). 2. Данные, переданные через куки ($_COOKIE). 3. Данные, переданные методом GET ($_GET). 4. Данные, переданные методом POST ($_POST). Правильная последовательность: 432. Пояснение: При стандартной настройке request_order = "GP" порядок таков: сначала обрабатываются POST-данные (G), затем GET-данные (P). Куки (C) по умолчанию не включаются в $_REQUEST. Данные из $_FILES в $_REQUEST никогда не попадают. Правильный порядок для стандартных настроек: 4 (POST), 3 (GET). **Задание 3.2.** Расположите действия в правильном порядке для безопасной обработки данных формы, отправленной методом POST. Действия: 1. Валидация данных: проверка на соответствие ожидаемым типам, форматам и длине (например, filter_var() для email, preg_match() для шаблонов). 2. Выполнение основного действия с очищенными данными (сохранение в БД, отправка письма и т.д.). 3. Получение сырых данных из суперглобального массива $_POST. 4. Санитизация (очистка) данных: экранирование специальных символов для конкретного контекста (например, htmlspecialchars() для вывода в HTML, mysqli_real_escape_string() или подготовленные выражения для SQL). Правильная последовательность: 3142. Пояснение: Правильный порядок — это «защита в глубину»: Получаем сырые данные

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			(3) — изначальный ввод пользователя. Валидируем (1) — проверяем, что данные соответствуют ожиданиям (формат, тип, допустимые значения). Если нет — отклоняем. Санитизируем (4) — преобразуем данные для безопасного использования в конкретном контексте (SQL, HTML, и т.д.). Выполняем действие (2) — работаем с уже проверенными и очищенными данными. Валидация проверяет «что» ввели (корректность), санитизация — «как» использовать (безопасность). Сначала убеждаемся, что данные корректны, затем готовим их для конкретной операции. **Задание 3.3** Установите соответствие между встроенной функцией/константой и её категорией или назначением. Список 1 (Функция/Константа): A. __DIR__ B. phpinfo() C. isset() D. empty() Список 2 (Категория/Назначение): 1. Псевдоконстанта, возвращающая директорию текущего файла. 2. Функция для проверки существования и не-NULL значения переменной. 3. Функция для получения подробной информации о конфигурации PHP. 4. Функция для проверки, является ли переменная "пустой" (0, '', null, false, [] и т.д.). Ответ: A1, B3, C2, D4. **Задание 3.4** Установите соответствие между суперглобальным массивом и типом данных, который он обычно содержит. Список 1 (Суперглобальный массив): A. $_ENV B. $_FILES

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			C. $_SERVER D. $_COOKIE Список 2 (Тип данных): 1. Информация о загружаемых на сервер файлах (имя, тип, временный путь). 2. Переменные окружения операционной системы и веб-сервера. 3. HTTP-куки, отправленные клиентом. 4. Информация о сервере и среде выполнения (заголовки, пути, скрипты). Ответ: A2, B1, C4, D3. **Задание 3.5** Какая встроенная константа PHP содержит номер текущей строки в файле, в которой она была использована? Варианты ответов: 1. __LINE__ 2. __FILE__ 3. __FUNCTION__ 4. PHP_VERSION Ответ: 1. Пояснение: __LINE__ — это "магическая" псевдоконстанта, которая в момент выполнения содержит номер текущей строки в файле. **Задание 3.5** Какой оператор или функция НЕ входит в стандартный набор встроенных возможностей PHP для подключения содержимого другого файла в текущий скрипт? Варианты ответов: 1. require 2. include_once 3. import 3. require_once

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			Ответ: 3. Пояснение: require, include_once, require_once — это все корректные языковые конструкции PHP для включения файлов.
Тема 1.4 Работа со строками	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 4.1** Установите соответствие между строковой функцией и её описанием. Список 1 (Функция): A. explode() B. implode() (или join()) C. str_replace() D. strpos() Список 2 (Описание): 1. Заменяет все вхождения поисковой строки на строку замены. 2. Разбивает строку на массив по заданному разделителю. 3. Возвращает позицию первого вхождения подстроки в строке. 4. Объединяет элементы массива в строку, вставляя между ними разделитель. Ответ: A2, B4, C1, D3. **Задание 4.2** Установите соответствие между синтаксисом задания строки и его описанием. Список 1 (Синтаксис): A. 'Текст $var текст' B. "Текст $var текст" C. <<<TEXT ... TEXT D. <<<'TEXT' ... TEXT Список 2 (Описание): 1. Интерполирует переменные, обрабатывает escape-последовательности. 2. Не интерполирует переменные, обрабатывает только \\ и \'.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			3. Интерполирует переменные, не требует экранирования кавычек внутри, удобен для многострочного текста. 4. Не интерполирует переменные, не требует экранирования кавычек внутри, удобен для многострочного текста. Ответ: A2, B1, C3, D4. **Задание 4.3** Какой будет результат выполнения следующего кода? $str = "Hello\nWorld"; echo strlen($str); (Предполагаем, что строка содержит символ перевода строки \n). Варианты ответов: 1. 10 2. 11 3. 12 4. 2 Ответ: 2. Пояснение: Функция strlen() возвращает длину строки в байтах, а не в символах. Строка "Hello\nWorld" состоит из: H(1) + e(1) + l(1) + l(1) + o(1) + \n(1 байт) + W(1) + o(1) + r(1) + l(1) + d(1) = 11 байт. **Задание 4.4** Какой функцией следует воспользоваться, чтобы найти последнее вхождение подстроки внутри строки, учитывая многобайтовую кодировку UTF-8? Варианты ответов: 1. strrpos() 2. mb_strrpos($str, $find, 0, 'UTF-8')

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			3. strpos() 4. mb_stripos($str, $find, 0, 'UTF-8') Ответ: 2. Пояснение: Для поиска последнего вхождения используется функция strrpos() (с двумя 'r'). Для работы с UTF-8 необходим её аналог из расширения mbstring — mb_strrpos(), где четвёртым аргументом указывается кодировка. **Задание 4.5** Как называется синтаксис для задания строк в PHP, который начинается с <<< и идентификатора в одинарных кавычках, и внутри которого переменные НЕ интерполируются, а поведение аналогично строке в одинарных кавычках? Ответ: Nowdoc (или Nowdoc-синтаксис). Пояснение: Nowdoc (<<<'EOT' ... EOT;) — это способ задания строк, который полезен для вставки больших блоков текста без необходимости экранировать кавычки и без подстановки переменных. Это делает его идеальным для шаблонов, SQL-запросов (хотя для SQL предпочтительнее подготовленные выражения) или любого литерального текста. **Задание 4.6** Как называется и какую функцию выполняет расширение PHP, которое необходимо активировать для корректной работы функций mb_* (таких как mb_strlen, mb_substr) с многобайтовыми кодировками, например, UTF-8? Ответ: Расширение mbstring (Multibyte String). Пояснение: Без активированного расширения mbstring функции mb_* не будут доступны, а стандартные строковые функции (strlen, substr и т.д.) будут работать некорректно с не-ASCII символами в UTF-8, так как оперируют байтами, а не символами.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
Тема 2.1 Протокол HTTP. Обработка форм	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 5.1.** Какие из следующих утверждений о методах HTTP и передаче данных являются верными? Варианты ответов: 1. Метод GET следует использовать для передачи конфиденциальных данных (паролей, номеров карт), так как данные не отображаются в адресной строке. 2. Данные, отправленные методом POST, доступны в PHP через суперглобальный массив $_POST. 3. Метод GET имеет ограничение на длину передаваемых данных, которое зависит от браузера и сервера. 4. Для получения сырых данных тела запроса (например, JSON) можно использовать file_get_contents('php://input'). Ответ: 2, 3, 4. Пояснение: Правильный (2): Это базовый принцип. Данные форм с method="POST" попадают в $_POST. Правильный (3): Ограничение существует, так как данные метода GET передаются в URL, длина которого ограничена (часто 2048-4096 символов). Для POST такого строгого ограничения обычно нет. Правильный (4): Поток php://input позволяет читать необработанное тело HTTP-запроса, что необходимо для работы с JSON, XML или данными, не пришедшими из HTML-формы. **Задание 5.2.** Какие из следующих действий являются обязательными или критически важными этапами безопасной обработки данных, полученных из HTML-формы? Варианты ответов: 1. Всегда проверять существование ключа в массиве $_POST с помощью isset() или empty() перед его использованием. 2. Фильтровать и валидировать данные: проверять тип, длину, формат (email, число) с помощью filter_var() или регулярных выражений.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			3. Использовать данные напрямую в SQL-запросах для удобства и читаемости кода. 4. Экранировать специальные символы при выводе данных в HTML с помощью htmlspecialchars() для предотвращения XSS. Ответ: 1, 2, 4. Пояснение: Правильный (1): Попытка обратиться к несуществующему ключу массива вызовет предупреждение (E_NOTICE). Проверка isset() — это базовый уровень защиты. Правильный (2): Валидация гарантирует, что данные соответствуют ожиданиям бизнес-логики (например, возраст — число, email — корректный формат). Правильный (4): Это основная мера защиты от межсайтового скриптинга (XSS) при выводе любых пользовательских данных на страницу. **Задание 5.3** Какой код состояния HTTP должен вернуть сервер, чтобы браузер понял, что запрашиваемый ресурс не был найден? Варианты ответов: 1. 200 OK 2. 302 Found 3. 404 Not Found 4. 500 Internal Server Error Ответ: 3. Пояснение: Код 404 Not Found — стандартный ответ, когда сервер не может найти запрашиваемый ресурс (страницу, файл). **Задание 5.4** Какая функция в PHP является наиболее безопасным и рекомендуемым способом для хэширования паролей перед их сохранением в базу данных? Варианты ответов:

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			1. md5($password) 2. sha1($password) 3. password_hash($password, PASSWORD_DEFAULT) 4. crypt($password, $salt) Ответ: 3. Пояснение: Функция password_hash() использует современный, стойкий алгоритм (по умолчанию bcrypt), автоматически генерирует криптографическую соль и предназначена специально для безопасного хранения паролей. **Задание 5.5** Как называется техника/уязвимость, при которой злоумышленник может внедрить и выполнить произвольный JavaScript-код на странице вашего сайта, используя некорректно обработанные данные, введённые другим пользователем? И какова основная встроенная функция PHP для защиты от неё при выводе данных в HTML? Ответ: Межсайтовый скриптинг (XSS — Cross-Site Scripting). Функция защиты — htmlspecialchars(). Пояснение: XSS — одна из самых распространённых веб-уязвимостей. Функция htmlspecialchars() преобразует специальные символы HTML (<, >, ", ', &) в их HTML-сущности, предотвращая их интерпретацию браузером как кода. **Задание 5.6** Какой метод HTTP является идемпотентным и безопасным (в терминологии RFC), не должен изменять состояние сервера и обычно используется только для получения данных? Ответ: GET. Пояснение: Согласно стандарту HTTP/1.1, методы GET и HEAD определяются как "безопасные" (safe), meaning they are intended only for information retrieval and should not change the state of the server. GET также является идемпотентным (многократный

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			повтор одного и того же запроса даёт тот же результат). Это фундаментальное свойство, отличающее его от POST, PUT, DELETE.
Тема 2.2 Работа с файловой системой	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 6.1.** Установите правильную последовательность безопасной обработки загружаемого на сервер изображения (аватара). Этапы обработки: 1. Проверить код ошибки загрузки в $_FILES['avatar']['error'] (должен быть UPLOAD_ERR_OK). 2. Сгенерировать уникальное имя файла (например, md5(uniqid()) . ".jpg") и определить безопасный путь для сохранения вне корневой веб-директории. 3. Проверить реальный MIME-тип файла с помощью finfo_file($finfo, $tmpPath) на соответствие image/jpeg, image/png и т.д. 4. Переместить временный файл из $_FILES['avatar']['tmp_name'] в итоговое место с помощью move_uploaded_file(). 5. Проверить наличие файла в массиве $_FILES и что загрузка вообще происходила методом POST. 6. Изменить размер изображения или создать миниатюру с помощью библиотеки GD или Imagick. Ответ: 513246. Пояснение: Проверки идут от общих к частным: существование запроса → техническая успешность → проверка содержимого → безопасное сохранение → дополнительная обработка. Тип файла проверяется после успешной загрузки, но до сохранения в постоянное место. **Задание 6.2** Расположите этапы в правильном порядке для рекурсивного удаления директории со всем её содержимым с помощью встроенных функций PHP. Этапы: 1. Открыть директорию с помощью opendir(). 2. Для каждого элемента проверить: если это файл — удалить unlink(), если это

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			директория — рекурсивно вызвать эту же процедуру. 3. Убедиться, что путь ведёт к существующей директории и это не символьная ссылка на опасный путь (базовая проверка безопасности). 4. Закрыть дескриптор директории с помощью closedir(). 5. Прочитать содержимое директории в цикле с помощью readdir(), пропуская . и ... 6. После очистки содержимого удалить саму пустую директорию с помощью rmdir(). Ответ: 315246. Пояснение: Сначала полностью очищаем все вложенные элементы (рекурсивно), и только потом удаляем саму родительскую директорию, когда она уже пуста. Безопасность проверяется в самом начале. **Задание 6.3** Установите соответствие между функцией PHP и её основным назначением при работе с файлами и директориями. Список 1 (Функция): A. scandir() B. is_uploaded_file() C. realpath() D. basename() Список 2 (Назначение): 1. Возвращает массив файлов и папок в указанной директории. 2. Проверяет, был ли файл загружен через HTTP POST (используется для безопасности перед move_uploaded_file). 3. Возвращает абсолютный канонизированный путь, убирая символы .., . и символические ссылки. 4. Возвращает последний компонент имени из указанного пути (имя файла). Ответ: A1, B2, C3, D4. **Задание 6.4**

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			Установите соответствие между режимом функции fopen() и его описанием. Список 1 (Режим): A. 'r' B. 'w' C. 'a' D. 'x' Список 2 (Описание): 1. Открывает файл только для записи; помещает указатель в конец файла. Если файл не существует, пытается его создать. 2. Открывает файл только для записи; обрезает файл до нулевой длины. Если файл не существует, пытается его создать. 3. Открывает файл только для чтения; помещает указатель в начало файла. 4. Создаёт и открывает файл только для записи; возвращает FALSE, если файл уже существует (защита от перезаписи). Ответ: A3, B2, C1, D4. **Задание 6.5** Какая функция является единственным безопасным способом переместить файл, загруженный через HTTP POST, из временной директории в постоянное место хранения на сервере? Варианты ответов: 1. copy($_FILES['file']['tmp_name'], $targetPath) 2. rename($_FILES['file']['tmp_name'], $targetPath) 3. move_uploaded_file($_FILES['file']['tmp_name'], $targetPath) 4. file_put_contents($targetPath, file_get_contents($_FILES['file']['tmp_name'])) Ответ: 3. Пояснение: move_uploaded_file() выполняет дополнительную критически важную проверку: она убеждается, что указанный файл ($tmp_name) действительно был

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			загружен методом HTTP POST в текущем скрипте. Это предотвращает возможность скрипта манипулировать произвольными системными файлами (например, /etc/passwd), если злоумышленник подделает значение $_FILES. **Задание 6.6** Какой результат вернёт функция file_exists() для пути, который является символьной ссылкой (symlink) на несуществующий файл? Варианты ответов: 1. Всегда TRUE, так как ссылка существует. 2. Всегда FALSE, так как файл, на который она ссылается, не существует. 3. Зависит от операционной системы и прав доступа. 4. Вызовет ошибку E_WARNING. Правильный ответ: 2. Пояснение: Функция file_exists() проверяет существование файла или директории, на который указывает путь. Если путь является "битой" (dangling) символьной ссылкой (указывает на несуществующий целевой файл), функция вернёт FALSE. Для проверки существования самой ссылки как файла нужно использовать is_link().
Тема 2.3 Cookie. Сессии	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 7.1** Установите соответствие между понятием/механизмом и его описанием. Список 1 (Понятие): A. Cookie (куки) B. Сессия (Session) C. Session ID D. $_SESSION Список 2 (Описание): 1. Уникальный идентификатор, связывающий данные на сервере с конкретным браузером пользователя. 2. Суперглобальный массив PHP для хранения данных, которые должны сохраняться в течение всей сессии пользователя.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			3. Механизм хранения состояния пользователя, при котором данные хранятся на сервере, а клиенту передаётся только идентификатор. 4. Небольшой фрагмент данных, отправляемый сервером браузеру и хранимый на стороне клиента; отправляется обратно с каждым запросом. Ответ: A4, B3, C1, D2. **Задание 7.2** Установите соответствие между параметром функции setcookie() и его назначением. Список 1 (Параметр): A. expires (или time) B. path C. secure D. httponly Список 2 (Назначение): 1. Определяет, в течение какого времени (Unix timestamp) cookie будет действителен. 0 или время в прошлом удаляет cookie. 2. Указывает путь на сервере, для которого cookie будет доступен. '/' означает весь сайт. 3. Если true, cookie будет передаваться только по защищённому соединению HTTPS. 4. Если true, cookie будет недоступен для скриптового языка (например, JavaScript), что помогает снизить риск XSS-атак. Ответ: A-1, B-2, C-3, D-4. **Задание 7.3** Каким образом клиент (браузер) обычно передаёт идентификатор сессии (session ID) серверу при каждом последующем запросе после начала сессии? Варианты ответов: 1. Через специальный заголовок HTTP X-Session-ID. 2. Через суперглобальный массив $_SERVER['HTTP_SESSION'].

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			3. Через параметр в теле POST-запроса. 4. Через cookie (обычно с именем PHPSESSID). Ответ: 4. Пояснение: Это основной и рекомендуемый механизм. При первом вызове session_start() сервер отправляет браузеру заголовок Set-Cookie. Браузер сохраняет этот cookie и автоматически включает его в заголовок Cookie всех последующих запросов к тому же домену. PHP автоматически извлекает оттуда session ID. **Задание 7.4** Почему хранение конфиденциальных данных (например, логинов, паролей, баланса) непосредственно в cookie является небезопасным? Варианты ответов: 1. Потому что cookie передаются только по HTTP и никогда по HTTPS. 2. Потому что cookie имеют ограниченный размер (4 КБ). 3. Потому что cookie хранятся на стороне клиента и пользователь может их легко просмотреть, изменить или подделать. 4. Потому что cookie автоматически шифруются PHP перед отправкой клиенту, что замедляет работу. Ответ: 3. Пояснение: Основная причина — отсутствие контроля над данными на стороне клиента. Пользователь или вредоносный скрипт (XSS) может прочитать, изменить (document.cookie) или подделать cookie. Поэтому в cookie следует хранить только нечувствительные данные (идентификаторы, токены) или данные, подписанные криптографической подписью для проверки их целостности на сервере. **Задание 7.5** Какая распространённая атака на веб-приложения возможна, если злоумышленнику удастся завладеть идентификатором сессии (session ID) легитимного пользователя, и

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			как называется основной метод защиты от неё (не считая использования HTTPS)? Ответ: Сессионный угон (Session Hijacking). Пояснение: Если атакующий узнает session ID (например, через перехват трафика, XSS), он может подставить его в свои запросы и выдать себя за жертву. Функция session_regenerate_id(true) создаёт новый ID и удаляет старую сессию, делая украденный ID бесполезным. Этот приём обязателен после аутентификации и при повышении уровня привилегий. **Задание 7.6** Как называется директива конфигурации PHP (php.ini), которая, если установлена в значение 1, предотвращает передачу идентификатора сессии через URL (в качестве параметра ?PHPSESSID=...), оставляя в качестве единственного способа передачу через cookie? Ответ: session.use_only_cookies. Пояснение: Установка session.use_only_cookies = 1 (рекомендуемое значение) отключает альтернативные, менее безопасные способы передачи session ID (через GET/POST параметры). Это предотвращает фиксацию сессии (session fixation) через URL и попадание session ID в лог-файлы, историю браузера и реферальные заголовки.
Тема 2.4 Использование MySQL в PHP	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 8.1** Какие из следующих утверждений о расширении MySQLi (MySQL Improved) являются верными? Варианты ответов: 1. MySQLi поддерживает как процедурный, так и объектно-ориентированный стиль программирования. 2. Функция mysqli_connect() возвращает объект соединения, который необходимо использовать во всех последующих вызовах. 3. Для выполнения запроса SELECT и получения результата всегда необходимо использовать комбинацию mysqli_query() и mysqli_store_result(). 4. Подготовленные выражения (prepared statements) в MySQLi позволяют безопасно

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			подставлять пользовательские данные в SQL-запрос, предотвращая SQL-инъекции. Ответ: 1, 4. Пояснение: Правильный (1): Это одно из ключевых преимуществ MySQLi. Можно использовать mysqli_ функции (процедурный стиль) или методы класса mysqli (ООП-стиль). Правильный (4): Это основное назначение подготовленных выражений. Параметры привязываются к запросу отдельно от его текста, что исключает интерпретацию данных как части SQL-команды. **Задание 8.2** Какие из следующих действий являются обязательными или настоятельно рекомендуемыми для безопасного исполнения SQL-запросов с пользовательскими данными? Варианты ответов: 1. Проверять пользовательский ввод с помощью preg_match() на отсутствие SQL-ключевых слов (SELECT, UNION, DROP). 2. Использовать подготовленные выражения ($stmt->prepare(), $stmt->bind_param(), $stmt->execute()). 3. Экранировать специальные символы в пользовательских строках с помощью mysqli_real_escape_string() перед подстановкой в SQL-строку. 4. Указывать кодировку соединения с базой данных как UTF-8 с помощью mysqli_set_charset($link, 'utf8mb4'). Ответ: 2, 4. Пояснение: Правильный (2): Подготовленные выражения — это единственный по-настоящему надёжный и рекомендуемый способ предотвращения SQL-инъекций в современных приложениях. Правильный (4): Установка корректной кодировки (предпочтительно utf8mb4 для полной поддержки Unicode) критически важна для предотвращения проблем с

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			данными и является частью безопасной настройки соединения. Без неё даже mysqli_real_escape_string() может работать некорректно. **Задание 8.3** Какая функция в MySQLi является единственным безопасным и корректным способом экранирования специальных символов в строке перед её использованием в SQL-запросе, если подготовленные выражения по какой-то причине не могут быть применены (например, для имени таблицы)? Варианты ответов: 1. addslashes() 2. htmlspecialchars() 3. mysqli_real_escape_string() 4. str_replace("'", "\'", $string) Ответ: 3. Пояснение: mysqli_real_escape_string() учитывает текущую кодировку соединения с БД и экранирует символы, которые имеют специальное значение в SQL, делая строку безопасной для использования внутри SQL-строк в кавычках. **Задание 8.4** Что возвращает метод $mysqli->query() при успешном выполнении запроса INSERT? Варианты ответов: 1. Количество вставленных строк. 2. Идентификатор (id) последней вставленной записи (AUTO_INCREMENT). 3. Объект результата mysqli_result. 4. Булево значение TRUE. Ответ: 4. Пояснение: Для запросов INSERT, UPDATE, DELETE, CREATE TABLE и т.д. (т.н. non-query) метод query() возвращает TRUE в случае успеха. Количество затронутых

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			строк можно получить через $mysqli->affected_rows. ID последней вставленной записи — через $mysqli->insert_id. **Задание 8.5** Как называется наиболее опасный класс уязвимостей веб-приложений, возникающий из-за некорректной обработки пользовательского ввода перед его включением в SQL-запрос, и какой единственный метод в MySQLi считается надёжной защитой от него? Ответ: SQL-инъекция (SQL Injection). Метод защиты — использование подготовленных выражений (Prepared Statements) с привязкой параметров. Пояснение: SQL-инъекция позволяет злоумышленнику выполнить произвольный SQL-код в базе данных. Подготовленные выражения разделяют команду (шаблон запроса) и данные, что исключает возможность данных влиять на структуру команды. **Задание 8.6** Какой метод объекта mysqli_result следует использовать для получения количества строк, возвращённых последним запросом SELECT, при использовании буферизованных запросов (по умолчанию)? Ответ: $result->num_rows (или mysqli_num_rows($result) в процедурном стиле). Пояснение: Свойство num_rows объекта результата содержит число строк в наборе результатов для буферизованных запросов. Для небуферизованных результатов или для получения количества строк, затронутых запросами INSERT, UPDATE, DELETE, используется $mysqli->affected_rows.
Тема 3.1 Объектно-ориентированное программирование	ПК- 7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов	**Задание 9.1** Какие из следующих утверждений о классах и объектах в PHP являются верными? Варианты ответов: 1. Класс — это чертёж (шаблон), а объект — его конкретная реализация в памяти во время выполнения программы. 2. Свойства класса должны всегда объявляться с модификатором доступа (public,

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			protected, private), иначе возникнет ошибка. 3. Метод __construct() вызывается автоматически при создании нового объекта оператором new. 4. Ключевое слово $this внутри метода класса ссылается на сам класс, а не на конкретный объект. Ответ: 1, 3. Пояснение: Правильный (1): Это классическое определение сути ООП. Класс описывает структуру и поведение, объект — конкретный экземпляр с уникальным состоянием. Правильный (3): __construct() — это "магический" метод-конструктор. Его вызов — часть процесса инстанцирования (создания) объекта. **Задание 9.2** Какие из следующих утверждений о наследовании и полиморфизме в PHP верны? Варианты ответов: 1. Класс-потомок (дочерний класс) наследует все public и protected методы и свойства родительского класса. 2. Чтобы переопределить (override) метод родительского класса в дочернем, необходимо использовать аннотацию @override. 3. Ключевое слово final, применённое к методу, запрещает его переопределение в дочерних классах. 4. Абстрактный класс может быть инстанцирован напрямую с помощью оператора new. Ответ: 1, 3. Пояснение: Правильный (1): Это базовый принцип наследования. private члены родительского класса не наследуются и недоступны в дочернем. Правильный (3): Модификатор final предотвращает переопределение метода в классах-наследниках. Если final применён к самому классу, то от него нельзя наследоваться.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			Задание 9.3 Установите правильную последовательность вызова "магических" методов при полном цикле создания, клонирования и уничтожения объекта. Последовательность вызова: 1. Вызов деструктора __destruct() оригинального объекта при завершении скрипта или unset. 2. Вызов конструктора __construct() для создания нового объекта. 3. Вызов метода __clone() для клонированного объекта (после операции clone $obj). Ответ: 231 Пояснение: конструктор вызывается при new. __clone() вызывается только для нового объекта-клона. Деструкторы вызываются автоматически в конце скрипта или при unset(). __toString() вызывается эпизодически при необходимости. Последовательность зависит от конкретного сценария, но логическая цепочка: создание → использование → (опционально клонирование) → уничтожение. **Задание 9.4** Расположите этапы в правильном порядке для разрешения (resolution) вызова метода $child->someMethod() в иерархии наследования, где ChildClass наследует от ParentClass. Этапы: 1. Интерпретатор PHP проверяет, существует ли метод someMethod() в классе ChildClass. 2. Если метод найден в ChildClass и он не private, выполняется он. 3. Если метод не найден в ChildClass, интерпретатор ищет его в родительском классе ParentClass. 4. Если метод не найден ни в одном из родителей в иерархии, возникает фатальная ошибка Fatal error: Uncaught Error: Call to undefined method. 5. Если метод найден в ParentClass и он private, возникает ошибка доступа. Ответ: 12354.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			Пояснение: PHP сначала проверяет класс, в контексте которого вызван метод (ChildClass), затем поднимается по иерархии наследования. Даже если метод существует в родителе, но объявлен как private, он не будет унаследован и недоступен из дочернего класса. **Задание 9.5** Как называется специальный метод в классе PHP, который автоматически вызывается при попытке выполнить операцию clone над объектом этого класса, и для чего он чаще всего используется? Ответ: __clone(). Пояснение: По умолчанию clone создает поверхностную (shallow) копию: все свойства копируются со своими значениями. Если свойство является ссылкой на другой объект, то копируется сама ссылка, и оба объекта ($original и $clone) начинают ссылаться на один и тот же вложенный объект. Метод __clone() позволяет переопределить это поведение и создать новые копии вложенных объектов. **Задание 9.6** Как называются методы в PHP, имена которых начинаются с двух символов подчёркивания (например, __construct, __get, __toString), которые предоставляют возможность перехватывать и переопределять различные операции над объектами? Ответ: "Магические" методы (Magic Methods). Пояснение: Эти методы не вызываются напрямую, а автоматически вызываются интерпретатором PHP в определённых ситуациях (при создании объекта, обращении к свойству, сериализации и т.д.). Они являются основой для реализации перегруженного поведения объектов в PHP.
Тема 3.2 Классы и интерфейсы	ПК- 7 Способен осуществлять проектирование	ИПК-7.1 Знать методы и средства проектирования	**Задание 10.1** Какие из следующих утверждений о константах и статических членах класса в PHP являются верными?

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
	программных интерфейсов	программных интерфейсов	Варианты ответов: 1. Константы класса можно объявить с помощью ключевого слова const внутри класса. Они не могут быть изменены после объявления. 2. К статическому свойству можно получить доступ только через объект класса ($obj->staticProperty). 3. Статические методы могут использовать псевдопеременную $this для доступа к обычным (нестатическим) свойствам объекта. 4. Статическое свойство, в отличие от константы, может менять своё значение в ходе выполнения программы. Ответ: 1, 4. Пояснение: Правильный (1): Константы класса (const NAME = value;) действительно неизменяемы и связаны с классом, а не с объектом. Правильный (4): Статические свойства (static $property;) принадлежат самому классу и могут быть изменены, например, через self::$property = newValue;. **Задание 10.2** Какие из следующих утверждений об абстрактных классах и интерфейсах в PHP верны? Варианты ответов: 1. Абстрактный класс может содержать как абстрактные (без реализации), так и обычные (с реализацией) методы. 2. Интерфейс может содержать объявления свойств с модификаторами доступа. 3.Класс может реализовывать (implements) несколько интерфейсов одновременно. 4. Интерфейс может содержать реализацию методов, если они объявлены как default. Ответ: 1, 3. Пояснение: Правильный (1): Это ключевое отличие абстрактного класса от интерфейса (в PHP до

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			8.0). Абстрактный класс может частично реализовывать функциональность. Правильный (3): Это одно из главных преимуществ интерфейсов перед одиночным наследованием классов. Класс может реализовать множество интерфейсов. **Задание 10.3** Установите правильную последовательность разрешения при обращении к статическому свойству self::$count в иерархии наследования, где Child наследует от Parent. Пункты обращения: 1. PHP ищет свойство $count в классе Parent. 2. PHP ищет свойство $count в классе Child. 3. Пользовательский код вызывает Child::incrementCount(). 4. Если свойство найдено в классе Parent, используется его значение. 5. Внутри метода Child::incrementCount() есть строка self::$count++. Правильная последовательность: 35214 Пояснение: ключевое слово self ссылается на класс, в котором оно написано (Child). Однако, если Child не переопределяет статическое свойство $count, PHP будет искать его в родительском классе (Parent) в рамках одного и того же объявления self::$count. **Задание 10.4** Расположите этапы в правильном порядке при создании экземпляра класса, который реализует интерфейс и наследует абстрактный класс. Этапы: 1. Интерпретатор проверяет, что класс предоставляет конкретную реализацию для всех абстрактных методов родительского абстрактного класса. 2. Разработчик объявляет класс с помощью ключевого слова class MyClass extends AbstractClass implements MyInterface. 3. Разработчик создаёт экземпляр: $obj = new MyClass();. 4. Интерпретатор проверяет, что класс предоставляет реализацию для всех методов, объявленных в интерфейсе MyInterface.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			5, Если все проверки пройдены, вызывается конструктор __construct() (если есть) и создаётся объект. Правильная последовательность: 21435. Пояснение: Проверки абстрактных методов и интерфейсов происходят на этапе компиляции/интерпретации, ещё до попытки создания объекта. Если проверки не пройдены — фатальная ошибка возникнет сразу, а не в момент вызова new. **Задание 10.5** Установите соответствие между ключевым словом/концепцией и его описанием. Список 1 (Ключевое слово/Концепция): A. final B. abstract C. static D. interface Список 2 (Описание): 1. Применённый к классу, запрещает наследование от него. Применённый к методу, запрещает его переопределение в дочерних классах. 2. Применённый к методу, указывает, что метод не имеет реализации в данном классе и должен быть реализован в дочернем. Применённый к классу, запрещает создание его экземпляров. 3. Указывает, что член класса (свойство или метод) принадлежит самому классу, а не его экземплярам. Обращение к нему происходит через ::. 4. Определяет контракт (набор методов), который должен быть реализован классом. Позволяет реализовать полиморфизм без иерархии наследования. Ответ: A1, B2, C3, D4. **Задание 10.6** Установите соответствие между магическим методом и его описанием или примером вызова.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Тест по теме
			Список 1 (Магический метод): A. __call($name, $arguments) B. __toString() C. __debugInfo() D. __isset($name) Список 2 (Описание): 1. Автоматически вызывается при попытке выполнить echo или print на объекте. 2. Вызывается при вызове функций var_dump() или print_r() на объекте, позволяет контролировать выводимые данные. 3. Вызывается при вызове недоступного (несуществующего или private) метода в контексте объекта. 4. Вызывается при использовании функции isset() или empty() на недоступном (несуществующем или private) свойстве. Ответ: A3, B1, C2, D4.

Критерии оценивания тестового задания:

Оценка	Критерии оценивания
отлично	от 90 до 100 % правильно выполненных заданий
хорошо	от 70 до 89 % правильно выполненных заданий
удовлетворительно	от 50 до 69 % правильно выполненных заданий
неудовлетворительно	менее 50 % правильно выполненных заданий

4.2 ТИПОВЫЕ ПРАКТИЧЕСКИЕ ЗАДАНИЯ

Практические задания должны отражать умение обучающегося применять осваиваемую компетенцию в практических ситуациях и при решении производственных задач.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию
Раздел №1 Тема 1.1 Основы PHP. Управляющие конструкции Тема 1.2 Пользовательские функции Тема 1.3 Встроенные функции Тема 1.4 Работа со строками	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	**Задание №1.** Создайте php-скрипт, который формирует веб-страницу с таблицей умножения. Ответ: <?php declare(strict_types=1); ?> <!DOCTYPE html> <html lang="ru"> <head> <meta charset="UTF-8"> <title>Таблица умножения</title> <style> body { font-family: Arial, sans-serif; margin: 20px; } table { border-collapse: collapse; margin: 20px auto; } th, td { border: 1px solid #333; padding: 8px 12px; text-align: center; } th { background-color: #f2f2f2; font-weight: bold; } tr:nth-child(even) { background-color: #f9f9f9; } tr:hover { background-color: #e8f4f8; } .header-cell { background-color: #4CAF50; color: white; } </style> </head> <body> Таблица умножения (от 1 до 10) <table>

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			<tr>
			<th class="header-cell">×</th>
			<?php for ($i = 1; $i <= 10; $i++): ?>
			<th class="header-cell"><?= $i ?></th>
			<?php endfor; ?>
			</tr>
			
			<?php for ($i = 1; $i <= 10; $i++): ?>
			<tr>
			<th class="header-cell"><?= $i ?></th>
			<?php for ($j = 1; $j <= 10; $j++): ?>
			<td><?= $i * $j ?></td>
			<?php endfor; ?>
			</tr>
			<?php endfor; ?>
			</table>
			 Всего операций: 100
			</body>
			</html>
			
			Задание №2.
			Напишите функцию распознавания простого числа. Функция в качестве аргумента должна получать число и возвращать TRUE, если число простое, и FALSE в противном случае.
			
			Ответ:
			<?php
			declare(strict_types=1);
			
			/**
			* Проверяет, является ли число простым
			*
			* @param int $number Число для проверки
			* @return bool TRUE если число простое, FALSE в противном случае
			*/
			function isPrime(int $number): bool {
			// Отрицательные числа, 0 и 1 не являются простыми
			if ($number <= 1) {
			return false;

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			}
			
			// 2 и 3 - простые числа
			if ($number <= 3) {
			return true;
			}
			
			// Все четные числа (кроме 2) и числа, делящиеся на 3, не простые
			if ($number % 2 === 0 || $number % 3 === 0) {
			return false;
			}
			
			// Проверяем делители в форме 6k ± 1 до квадратного корня из числа
			$i = 5;
			while ($i * $i <= $number) {
			if ($number % $i === 0 || $number % ($i + 2) === 0) {
			return false;
			}
			$i += 6;
			}
			
			return true;
			}
			
			// Тестирование функции
			?>
			<!DOCTYPE html>
			<html lang="ru">
			<head>
			<meta charset="UTF-8">
			<title>Проверка простых чисел</title>
			<style>
			body { font-family: Arial, sans-serif; margin: 20px; }
			.prime { color: green; font-weight: bold; }
			.not-prime { color: red; }
			</style>
			</head>
			<body>
			 Проверка чисел на простоту
			
			<?php

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			$testNumbers = [-5, 0, 1, 2, 3, 4, 17, 25, 29, 97, 100, 113];
			
			foreach ($testNumbers as $num) {
			$isPrime = isPrime($num);
			$class = $isPrime ? 'prime' : 'not-prime';
			$result = $isPrime ? 'простое' : 'не простое';
			echo "<li class='$class'>$num — $result ";
			}
			?>
			
			 Простые числа от 1 до 100:
			<?php
			$primes = [];
			for ($i = 1; $i <= 100; $i++) {
			if (isPrime($i)) {
			$primes[] = $i;
			}
			}
			echo implode(', ', $primes);
			?>
			</body>
			</html>

Раздел №2
Тема 2.1 Протокол HTTP. Обработка форм
Тема 2.2 Работа с файловой системой
Тема 2.3 Cookie. Сессии
Тема 2.4 Использование MySQL в PHP
	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	**Задание №3.**
			Создайте форму ввода данных о пользователе (ФИО, e-mail, телефон). Напишите скрипт, который проверяет правильность заполнения полей формы (отсутствие пустых полей). Если форма заполнена неверно, то скрипт должен выводить сообщение с указанием ошибки и давать возможность скорректировать ввод.
			
			Ответ:
			<?php
			declare(strict_types=1);
			
			// Обработка отправки формы
			$errors = [];
			$fields = [
			'fio' => '',

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

```
    'email' => '',
    'phone' => ''
];

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Получение и очистка данных
    $fields['fio'] = trim($_POST['fio'] ?? '');
    $fields['email'] = trim($_POST['email'] ?? '');
    $fields['phone'] = trim($_POST['phone'] ?? '');

    // Валидация
    if (empty($fields['fio'])) {
        $errors[] = 'Поле "ФИО" обязательно для заполнения.';
    } elseif (strlen($fields['fio']) < 5) {
        $errors[] = 'ФИО должно содержать не менее 5 символов.';
    }

    if (empty($fields['email'])) {
        $errors[] = 'Поле "E-mail" обязательно для заполнения.';
    } elseif (!filter_var($fields['email'], FILTER_VALIDATE_EMAIL)) {
        $errors[] = 'Введите корректный e-mail адрес.';
    }

    if (empty($fields['phone'])) {
        $errors[] = 'Поле "Телефон" обязательно для заполнения.';
    } elseif (!preg_match('/^[\d\s\-\+\(\)]{10,15}$/', $fields['phone'])) {
        $errors[] = 'Введите корректный номер телефона (10-15 цифр, могут быть пробелы, +, -, скобки).';
    }

    // Если ошибок нет - выводим успешное сообщение
    if (empty($errors)) {
        echo '<div style="color: green; padding: 15px; border: 2px solid green; margin: 20px;">';
        echo '<h3>Данные успешно отправлены!</h3>';
        echo '<p><strong>ФИО:</strong> ' . htmlspecialchars($fields['fio']) . '</p>';
        echo '<p><strong>E-mail:</strong> ' . htmlspecialchars($fields['email']) . '</p>';
        echo '<p><strong>Телефон:</strong> ' . htmlspecialchars($fields['phone']) . '</p>';
        echo '<p><a href="' . htmlspecialchars($_SERVER['PHP_SELF']) . '">Вернуться к форме</a></p>';
        echo '</div>';
        exit;
    }
}
```

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию
			?> <!DOCTYPE html> <html lang="ru"> <head> <meta charset="UTF-8"> <title>Форма ввода данных</title> <style> body { font-family: Arial, sans-serif; max-width: 600px; margin: 20px auto; padding: 20px; } .error { color: red; background-color: #FFE6E6; padding: 10px; border: 1px solid red; margin: 10px 0; } .success { color: green; padding: 10px; } form { background-color: #f9f9f9; padding: 20px; border-radius: 5px; } label { display: block; margin: 10px 0 5px; font-weight: bold; } input { width: 100%; padding: 8px; margin-bottom: 15px; box-sizing: border-box; } button { background-color: #4CAF50; color: white; padding: 10px 20px; border: none; cursor: pointer; } button:hover { background-color: #45a049; } </style> </head> <body> Форма ввода данных о пользователе <?php if (!empty($errors)): ?> Обнаружены ошибки: <?php foreach ($errors as $error): ?> <?= htmlspecialchars($error) ?> <?php endforeach; ?> <?php endif; ?> <form method="POST" action="<?= htmlspecialchars($_SERVER['PHP_SELF']) ?>"> <label for="fio">ФИО:</label> <input type="text" id="fio" name="fio" value="<?= htmlspecialchars($fields['fio']) ?>" placeholder="Иванов Иван Иванович" required> <label for="email">E-mail:</label> <input type="email" id="email" name="email"

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию
			value="<?= htmlspecialchars($fields['email']) ?>" placeholder="example@mail.ru" required> <label for="phone">Телефон:</label> <input type="tel" id="phone" name="phone" value="<?= htmlspecialchars($fields['phone']) ?>" placeholder="+7 (999) 123-45-67" required> <button type="submit">Отправить</button> <button type="reset">Очистить</button> </form> Требования к полям: - ФИО: не менее 5 символов - E-mail: корректный email адрес - Телефон: 10-15 цифр, могут быть пробелы, +, -, скобки </body> </html> **Задание №4.** Элементы массива заданы в файле. Произвести чтение из файла и вывод элементов массива. Ответ: Файл data.txt: Яблоки Бананы Апельсины Груши Виноград Персики Файл read_array.php:

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			<?php
			declare(strict_types=1);
			
			/**
			* Читает массив из файла
			*
			* @param string $filename Имя файла
			* @return array Массив строк из файла
			*/
			function readArrayFromFile(string $filename): array {
			if (!file_exists($filename)) {
			throw new RuntimeException("Файл '$filename' не найден.");
			}
			
			$content = file_get_contents($filename);
			if ($content === false) {
			throw new RuntimeException("Не удалось прочитать файл '$filename'.");
			}
			
			// Разбиваем на строки, убираем пустые строки
			$lines = explode(PHP_EOL, $content);
			$lines = array_map('trim', $lines);
			$lines = array_filter($lines, function($line) {
			return $line !== '';
			});
			
			return array_values($lines); // Переиндексируем массив
			}
			?>
			<!DOCTYPE html>
			<html lang="ru">
			<head>
			<meta charset="UTF-8">
			<title>Чтение массива из файла</title>
			<style>
			body { font-family: Arial, sans-serif; margin: 20px; }
			table { border-collapse: collapse; margin: 20px 0; }
			th, td { border: 1px solid #333; padding: 10px; }
			th { background-color: #f2f2f2; }
			tr:nth-child(even) { background-color: #f9f9f9; }
			</style>

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			</head>
			<body>
			 Чтение массива из файла data.txt
			
			<?php
			try {
			$filename = 'data.txt';
			$array = readArrayFromFile($filename);
			
			echo ' Файл успешно прочитан. Количество элементов: ' . count($array) . ' ';
			
			if (!empty($array)) {
			echo '<table>';
			echo '<tr><th>№</th><th>Элемент массива</th><th>Длина строки</th></tr>';
			
			foreach ($array as $index => $value) {
			$number = $index + 1;
			echo "<tr><td>$number</td><td>" . htmlspecialchars($value) . "</td><td>" . strlen($value) . "</td></tr>";
			}
			
			echo '</table>';
			
			echo ' Дополнительная информация: ';
			echo ' Первый элемент: ' . htmlspecialchars(reset($array)) . ' ';
			echo ' Последний элемент: ' . htmlspecialchars(end($array)) . ' ';
			echo ' Все элементы через запятую: ' . htmlspecialchars(implode(', ', $array)) . ' ';
			} else {
			echo ' Файл пуст или содержит только пустые строки. ';
			}
			
			} catch (RuntimeException $e) {
			echo ' Ошибка: ' . htmlspecialchars($e->getMessage()) . ' ';
			}
			?>
			
			 <a href="<?= htmlspecialchars($_SERVER['PHP_SELF']) ?>">Обновить страницу
			</body>
			</html>

Раздел №3 Тема 3.1	ПК-7 Способен осуществлять	ИПК-7.2 Уметь применять методы и	**Задание №5**

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию
Объектно-ориентированное программирование Тема 3.2 Классы и интерфейсы	проектирование программных интерфейсов	средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	Создайте иерархию классов для системы учёта транспортных средств. Требования: 1. Создайте абстрактный класс Vehicle (Транспортное средство) со следующими характеристиками: a. Защищённые свойства: $brand (марка), $model (модель), $year (год выпуска) b. Конструктор, инициализирующий эти свойства c. Абстрактный метод getInfo(), возвращающий информацию о транспортном средстве d. Абстрактный метод calculateTax(), рассчитывающий налог (5% от года выпуска × 100) 2. Создайте интерфейс FuelConsumable (Расходует топливо) с методами: a. getFuelType() - возвращает тип топлива b. calculateFuelConsumption($distance) - рассчитывает расход топлива на расстояние 3. Создайте интерфейс CargoTransport (Перевозит грузы) с методом: a. getMaxCargoWeight() - возвращает максимальный вес груза 4. Создайте класс Car (Автомобиль), который: a. Наследует от Vehicle b. Реализует интерфейс FuelConsumable c. Имеет дополнительное свойство $fuelType (тип топлива) d. Реализует все необходимые методы 5. Создайте класс Truck (Грузовик), который: a. Наследует от Vehicle b. Реализует оба интерфейса: FuelConsumable и CargoTransport c. Имеет дополнительные свойства: $fuelType и $maxCargoWeight

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			d. Реализует все необходимые методы
			
			В основном скрипте создайте массив транспортных средств (минимум 3 объекта: 2 автомобиля и 1 грузовик). Выведите информацию о каждом транспортном средстве, включая расчёт налога и, где применимо, расход топлива и грузоподъёмность.
			
			Ответ:
			`<?php`
			`declare(strict_types=1);`
			
			`// 1. Абстрактный класс Vehicle`
			`abstract class Vehicle`
			`{`
			`protected string $brand;`
			`protected string $model;`
			`protected int $year;`
			
			`public function __construct(string $brand, string $model, int $year)`
			`{`
			`$this->brand = $brand;`
			`$this->model = $model;`
			`$this->year = $year;`
			`}`
			
			`abstract public function getInfo(): string;`
			
			`abstract public function calculateTax(): float;`
			
			`protected function getBaseInfo(): string`
			`{`
			`return "{$this->brand} {$this->model} ({$this->year} г.)";`
			`}`
			`}`
			
			`// 2. Интерфейс FuelConsumable`
			`interface FuelConsumable`
			`{`

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			public function getFuelType(): string;
			public function calculateFuelConsumption(float $distance): float;
			}
			
			// 3. Интерфейс CargoTransport
			interface CargoTransport
			{
			public function getMaxCargoWeight(): float;
			}
			
			// 4. Класс Car
			class Car extends Vehicle implements FuelConsumable
			{
			private string $fuelType;
			private const FUEL_CONSUMPTION_RATE = 0.08; // 8 л/100 км
			
			public function __construct(string $brand, string $model, int $year, string $fuelType)
			{
			parent::__construct($brand, $model, $year);
			$this->fuelType = $fuelType;
			}
			
			public function getInfo(): string
			{
			return "Автомобиль: " . $this->getBaseInfo() .
			", тип топлива: {$this->fuelType}";
			}
			
			public function calculateTax(): float
			{
			return 0.05 * $this->year * 100;
			}
			
			public function getFuelType(): string
			{
			return $this->fuelType;
			}
			
			public function calculateFuelConsumption(float $distance): float
			{
			return ($distance / 100) * self::FUEL_CONSUMPTION_RATE;

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			}
			}
			
			// 5. Класс Truck
			class Truck extends Vehicle implements FuelConsumable, CargoTransport
			{
			private string $fuelType;
			private float $maxCargoWeight;
			private const FUEL_CONSUMPTION_RATE = 0.25; // 25 л/100 км
			
			public function __construct(
			string $brand,
			string $model,
			int $year,
			string $fuelType,
			float $maxCargoWeight
			) {
			parent::__construct($brand, $model, $year);
			$this->fuelType = $fuelType;
			$this->maxCargoWeight = $maxCargoWeight;
			}
			
			public function getInfo(): string
			{
			return "Грузовик: " . $this->getBaseInfo() .
			", тип топлива: {$this->fuelType}, " .
			"грузоподъёмность: {$this->maxCargoWeight} т";
			}
			
			public function calculateTax(): float
			{
			// Для грузовиков налог в 1.5 раза выше
			return 1.5 * (0.05 * $this->year * 100);
			}
			
			public function getFuelType(): string
			{
			return $this->fuelType;
			}
			
			public function calculateFuelConsumption(float $distance): float

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			{
			return ($distance / 100) * self::FUEL_CONSUMPTION_RATE;
			}
			
			public function getMaxCargoWeight(): float
			{
			return $this->maxCargoWeight;
			}
			}
			
			// 6. Основной скрипт
			?>
			<!DOCTYPE html>
			<html lang="ru">
			<head>
			<meta charset="UTF-8">
			<title>Система учёта транспортных средств</title>
			<style>
			body { font-family: Arial, sans-serif; margin: 20px; }
			.vehicle { border: 1px solid #ccc; padding: 15px; margin: 10px 0; border-radius: 5px; }
			.car { background-color: #f0f8ff; }
			.truck { background-color: #fff0f5; }
			h3 { color: #333; margin-top: 0; }
			.info { margin: 5px 0; }
			.total { background-color: #e6ffe6; padding: 15px; margin-top: 20px; }
			</style>
			</head>
			<body>
			 Система учёта транспортных средств
			
			<?php
			// Создание массива транспортных средств
			$vehicles = [
			new Car('Toyota', 'Camry', 2020, 'Бензин'),
			new Car('Tesla', 'Model 3', 2022, 'Электричество'),
			new Truck('MAN', 'TGX', 2019, 'Дизель', 18.5),
			new Truck('КАМАЗ', '54901', 2021, 'Дизель', 15.0)
			];
			
			$totalTax = 0;
			$distance = 500; // расстояние в км для расчёта расхода

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			foreach ($vehicles as $vehicle) {
			$class = $vehicle instanceof Truck ? 'truck' : 'car';
			echo " ";
			echo " " . $vehicle->getInfo() . " ";
			
			$tax = $vehicle->calculateTax();
			$totalTax += $tax;
			echo " Налог: " . number_format($tax, 2) . " руб. ";
			
			// Если транспортное средство расходует топливо
			if ($vehicle instanceof FuelConsumable) {
			$consumption = $vehicle->calculateFuelConsumption($distance);
			echo " Расход топлива на {$distance} км: " .
			number_format($consumption, 2) . " л ";
			echo " Тип топлива: " . $vehicle->getFuelType() . " ";
			}
			
			// Если транспортное средство перевозит грузы
			if ($vehicle instanceof CargoTransport) {
			echo " Макс. грузоподъёмность: " .
			$vehicle->getMaxCargoWeight() . " т ";
			}
			
			// Проверка типа объекта
			echo " Тип объекта: " . get_class($vehicle) . " ";
			
			echo " ";
			}
			
			// Итоговая информация
			echo " ";
			echo " Итоговая информация: ";
			echo " Всего транспортных средств: " . count($vehicles) . " ";
			echo " Общая сумма налога: " . number_format($totalTax, 2) . "
			руб. ";
			echo " Количество автомобилей: " .
			count(array_filter($vehicles, fn($v) => $v instanceof Car && !$v instanceof Truck)) . " ";
			echo " Количество грузовиков: " .
			count(array_filter($vehicles, fn($v) => $v instanceof Truck)) . " ";
			echo " ";

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию
			?> </body> </html> **Задание №6** Разработайте систему обработки платежей для интернет-магазина. Требования: 1. Создайте трейт Loggable с методами: a. log(string $message) - записывает сообщение в лог с временной меткой b. Статическое свойство $logFile с именем файла лога 2. Создайте абстрактный класс PaymentMethod (Способ оплаты) со следующими характеристиками: a. Защищённые свойства: $paymentId (идентификатор платежа), $amount (сумма), $currency (валюта) b. Конструктор, инициализирующий эти свойства c. Абстрактный метод processPayment(), возвращающий результат обработки d. Абстрактный метод getPaymentDetails(), возвращающий детали платежа e. Используйте трейт Loggable 3. Создайте интерфейс Refundable (Возвратный) с методом: a. processRefund(float $amount) - обрабатывает возврат средств 4. Создайте интерфейс SecurePayment (Безопасный платеж) с методом: a. validateSecurity() - выполняет проверку безопасности 5. Создайте класс CreditCardPayment (Оплата картой), который: a. Наследует от PaymentMethod b. Реализует интерфейсы Refundable и SecurePayment c. Имеет дополнительные свойства: $cardNumber

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			(маскированный номер), $cardHolder
			d. Переопределяет все необходимые методы
			6. Создайте класс PayPalPayment (Оплата PayPal), который:
			a. Наследует от PaymentMethod
			b. Реализует интерфейс Refundable
			c. Имеет дополнительные свойства: $email (email пользователя)
			d. Переопределяет все необходимые методы
			7. Создайте финальный класс PaymentProcessor (Обработчик платежей) со статическими методами:
			a. process(PaymentMethod $payment) - обрабатывает переданный платеж
			b. getTotalProcessed() - возвращает общее количество обработанных платежей
			В основном скрипте создайте несколько платежей разных типов, обработайте их и выведите результаты, включая логи.
			
			Ответ:
			<?php
			declare(strict_types=1);
			
			// 1. Трейт Loggable
			trait Loggable
			{
			protected static string $logFile = 'payment_log.txt';
			
			protected function log(string $message): void
			{
			$timestamp = date('Y-m-d H:i:s');
			$logEntry = "[$timestamp] " . get_class($this) . ": $message" . PHP_EOL;
			
			// В реальном приложении лучше использовать Monolog или аналоги
			file_put_contents(self::$logFile, $logEntry, FILE_APPEND);
			}

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			public static function getLogContent(): string
			{
			return file_exists(self::$logFile) ?
			file_get_contents(self::$logFile) : 'Лог пуст';
			}
			
			public static function clearLog(): void
			{
			if (file_exists(self::$logFile)) {
			file_put_contents(self::$logFile, '');
			}
			}
			}
			
			// 2. Абстрактный класс PaymentMethod
			abstract class PaymentMethod
			{
			use Loggable;
			
			protected string $paymentId;
			protected float $amount;
			protected string $currency;
			protected string $status;
			
			public function __construct(float $amount, string $currency = 'RUB')
			{
			$this->paymentId = uniqid('pay_', true);
			$this->amount = $amount;
			$this->currency = $currency;
			$this->status = 'pending';
			
			$this->log("Создан новый платёж ID: {$this->paymentId}, сумма: {$amount} {$currency}");
			}
			
			abstract public function processPayment(): bool;
			
			abstract public function getPaymentDetails(): string;
			
			public function getPaymentId(): string
			{
			return $this->paymentId;

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			}
			
			public function getAmount(): float
			{
			return $this->amount;
			}
			
			public function getStatus(): string
			{
			return $this->status;
			}
			
			protected function setStatus(string $status): void
			{
			$this->status = $status;
			$this->log("Статус платежа {$this->paymentId} изменён на: {$status}");
			}
			}
			
			// 3. Интерфейс Refundable
			interface Refundable
			{
			public function processRefund(float $amount): bool;
			}
			
			// 4. Интерфейс SecurePayment
			interface SecurePayment
			{
			public function validateSecurity(): bool;
			}
			
			// 5. Класс CreditCardPayment
			class CreditCardPayment extends PaymentMethod implements Refundable, SecurePayment
			{
			private string $cardNumber;
			private string $cardHolder;
			private const MAX_REFUND_PERCENT = 0.8; // Максимум 80% возврата
			
			public function __construct(
			float $amount,
			string $cardNumber,

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			string $cardHolder,
			string $currency = 'RUB'
			) {
			parent::__construct($amount, $currency);
			$this->cardNumber = substr($cardNumber, -4); // Сохраняем только последние 4 цифры
			$this->cardHolder = $cardHolder;
			}
			
			public function processPayment(): bool
			{
			$this->log("Начало обработки платежа по карте: ****{$this->cardNumber}");
			
			// Имитация проверки безопасности
			if (!$this->validateSecurity()) {
			$this->setStatus('security_failed');
			$this->log("Платёж отклонён: проверка безопасности не пройдена");
			return false;
			}
			
			// Имитация обработки платежа (в реальности здесь был бы API банка)
			sleep(1); // Имитация задержки
			
			$success = rand(0, 100) > 10; // 90% успешных платежей
			
			if ($success) {
			$this->setStatus('completed');
			$this->log("Платёж успешно обработан");
			return true;
			} else {
			$this->setStatus('failed');
			$this->log("Платёж не прошёл по техническим причинам");
			return false;
			}
			}
			
			public function getPaymentDetails(): string
			{
			return "Оплата картой | Держатель: {$this->cardHolder} \| Карта: ****{$this->cardNumber} | " .
			"Сумма: {$this->amount} {$this->currency} | Статус: {$this->status}";
			}

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			public function processRefund(float $amount): bool
			{
			$maxRefund = $this->amount * self::MAX_REFUND_PERCENT;
			
			if ($amount > $maxRefund) {
			$this->log("Запрошен возврат {$amount}, но максимально возможный: {$maxRefund}");
			return false;
			}
			
			// Имитация возврата средств
			$this->log("Возврат средств на карту ****{$this->cardNumber}: {$amount} {$this->currency}");
			$this->setStatus('refunded');
			return true;
			}
			
			public function validateSecurity(): bool
			{
			// Простая имитация проверки безопасности
			$isValid = strlen($this->cardHolder) > 3 && $this->amount > 0;
			$this->log("Проверка безопасности: " . ($isValid ? 'Пройдена' : 'Не пройдена'));
			return $isValid;
			}
			}
			
			// 6. Класс PayPalPayment
			class PayPalPayment extends PaymentMethod implements Refundable
			{
			private string $email;
			
			public function __construct(float $amount, string $email, string $currency = 'USD')
			{
			parent::__construct($amount, $currency);
			$this->email = $email;
			}
			
			public function processPayment(): bool
			{
			$this->log("Начало обработки PayPal платежа для: {$this->email}");
			
			// Имитация обработки через PayPal API
			sleep(2); // PayPal обычно медленнее

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			$success = rand(0, 100) > 5; // 95% успешных платежей
			
			if ($success) {
			$this->setStatus('completed');
			$this->log("PayPal платёж успешно обработан");
			return true;
			} else {
			$this->setStatus('failed');
			$this->log("PayPal платёж отклонён");
			return false;
			}
			}
			
			public function getPaymentDetails(): string
			{
			return "PayPal оплата | Email: {$this->email} | " .
			"Сумма: {$this->amount} {$this->currency} | Статус: {$this->status}";
			}
			
			public function processRefund(float $amount): bool
			{
			// PayPal позволяет полный возврат
			if ($amount <= $this->amount) {
			$this->log("Возврат через PayPal на {$this->email}: {$amount} {$this->currency}");
			$this->setStatus('refunded_partial');
			
			if ($amount == $this->amount) {
			$this->setStatus('refunded_full');
			}
			
			return true;
			}
			
			$this->log("Запрошен возврат {$amount}, но сумма платежа только {$this->amount}");
			return false;
			}
			}
			
			// 7. Финальный класс PaymentProcessor
			final class PaymentProcessor

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			{
			private static int $totalProcessed = 0;
			private static array $processedPayments = [];
			
			public static function process(PaymentMethod $payment): array
			{
			$startTime = microtime(true);
			
			$result = [
			'payment_id' => $payment->getPaymentId(),
			'payment_type' => get_class($payment),
			'amount' => $payment->getAmount(),
			'status_before' => $payment->getStatus()
			];
			
			$success = $payment->processPayment();
			
			$result['success'] = $success;
			$result['status_after'] = $payment->getStatus();
			$result['processing_time'] = round(microtime(true) - $startTime, 3);
			$result['details'] = $payment->getPaymentDetails();
			
			self::$totalProcessed++;
			self::$processedPayments[] = $result;
			
			return $result;
			}
			
			public static function getTotalProcessed(): int
			{
			return self::$totalProcessed;
			}
			
			public static function getProcessedPayments(): array
			{
			return self::$processedPayments;
			}
			
			public static function getStatistics(): array
			{
			$stats = [

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

```
            'total' => self::$totalProcessed,
            'successful' => 0,
            'failed' => 0,
            'total_amount' => 0
        ];

        foreach (self::$processedPayments as $payment) {
            if ($payment['success']) {
                $stats['successful']++;
                $stats['total_amount'] += $payment['amount'];
            } else {
                $stats['failed']++;
            }
        }

        return $stats;
    }
}

// 8. Основной скрипт

// Очистка лога перед запуском
Loggable::clearLog();

?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Система обработки платежей</title>
    <style>
        body { font-family: Arial, sans-serif; margin: 20px; }
        .payment { border: 1px solid #ccc; padding: 15px; margin: 10px 0; border-radius: 5px; }
        .credit-card { background-color: #f0fff0; }
        .paypal { background-color: #f0f8ff; }
        .success { border-left: 5px solid #4CAF50; }
        .failed { border-left: 5px solid #f44336; }
        .refunded { border-left: 5px solid #ff9800; }
        h3 { color: #333; margin-top: 0; }
        .info { margin: 5px 0; }
        .stats { background-color: #e6f7ff; padding: 15px; margin: 20px 0; }
```

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

```
        .log { background-color: #f9f9f9; padding: 15px; font-family: monospace; white-space: pre-wrap; }
        .interface-badge {
            display: inline-block;
            background-color: #6c757d;
            color: white;
            padding: 2px 8px;
            border-radius: 12px;
            font-size: 0.8em;
            margin-right: 5px;
        }
    </style>
</head>
<body>
<h2>Система обработки платежей</h2>

<?php
// Создание платежей
$payments = [
    new CreditCardPayment(5000.00, '4111111111111111', 'Иванов И.И.', 'RUB'),
    new CreditCardPayment(125.50, '5555555555554444', 'Петрова А.С.', 'USD'),
    new PayPalPayment(89.99, 'customer1@example.com', 'USD'),
    new PayPalPayment(7500.00, 'company@example.com', 'RUB'),
    new CreditCardPayment(300.00, '378282246310005', 'Сидоров А.П.', 'EUR')
];

// Обработка платежей
echo "<h3>Обработка платежей:</h3>";
foreach ($payments as $payment) {
    $result = PaymentProcessor::process($payment);

    $classes = [];
    $classes[] = strpos(get_class($payment), 'CreditCard') !== false ? 'credit-card' : 'paypal';
    $classes[] = $result['success'] ? 'success' : 'failed';
    if (strpos($result['status_after'], 'refund') !== false) {
        $classes[] = 'refunded';
    }

    echo "<div class='payment " . implode(' ', $classes) . "'>";
    echo "<h3>" . $result['details'] . "</h3>";

    echo "<div class='info'><strong>ID платежа:</strong> " . $result['payment_id'] . "</div>";
```

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			echo " Тип: " . $result['payment_type'] . " ";
			echo " Время обработки: " . $result['processing_time'] . "
			сек ";
			
			// Показываем реализованные интерфейсы
			$interfaces = class_implements($payment);
			if (!empty($interfaces)) {
			echo " Интерфейсы: ";
			foreach ($interfaces as $interface) {
			echo "$interface";
			}
			echo " ";
			}
			
			echo " ";
			}
			
			// Вывод статистики
			$stats = PaymentProcessor::getStatistics();
			echo " ";
			echo " Статистика обработки: ";
			echo " Всего обработано платежей: " . $stats['total'] . " ";
			echo " Успешных: " . $stats['successful'] . " ";
			echo " Неудачных: " . $stats['failed'] . " ";
			echo " Общая сумма успешных платежей: " .
			number_format($stats['total_amount'], 2) . " (в разных валютах) ";
			echo " ";
			
			// Демонстрация возврата средств
			echo " Демонстрация возврата средств: ";
			$firstPayment = $payments[0];
			if ($firstPayment instanceof Refundable) {
			$refundAmount = 2500.00;
			$refundResult = $firstPayment->processRefund($refundAmount);
			
			echo " ";
			echo " Возврат средств по платежу: " . $firstPayment->getPaymentId() . " ";
			echo " Запрошенная сумма возврата: {$refundAmount} ";
			echo " Результат: " .
			($refundResult ? 'Успешно' : 'Не удалось') . " ";
			echo " Новый статус: " . $firstPayment->getStatus() . " ";

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Задания к практическому занятию

			echo " ";
			}
			
			// Вывод лога
			echo " Лог выполнения: ";
			echo " " . htmlspecialchars(Loggable::getLogContent()) . " ";
			
			// Информация о классах
			echo " ";
			echo " Информация о системе: ";
			echo " Абстрактный класс: PaymentMethod ";
			echo " Интерфейсы: Refundable, SecurePayment ";
			echo " Трейт: Loggable ";
			echo " Финальный класс: PaymentProcessor ";
			echo " Конкретные классы: CreditCardPayment, PayPalPayment ";
			echo " ";
			?>
			</body>
			</html>

Критерии оценивания практических занятий:

Оценка	Критерии оценивания
отлично	Выставляется, если обучающийся умеет увязывать теорию с практикой (решает задачи, формулирует выводы, умеет пояснить полученные результаты), владеет понятийным аппаратом, полно и глубоко овладел материалом по заданной теме, обосновывает свои суждения и даёт правильные ответы на вопросы преподавателя
хорошо	Выставляется, если обучающийся умеет увязывать теорию с практикой (решает задачи и формулирует выводы, умеет пояснить полученные результаты), владеет понятийным аппаратом, полно и глубоко овладел материалом по заданной теме, но содержание ответов имеют некоторые неточности и требуют уточнения и комментария со стороны преподавателя
удовлетворительно	Выставляется, если обучающийся знает и понимает материал по заданной теме, но изложение неполное, непоследовательное, допускаются неточности в определении понятий, студент не может обосновать свои ответы на уточняющие вопросы преподавателя
неудовлетворительно	Выставляется, если обучающийся допускает ошибки в определении понятий, искажающие их смысл, беспорядочно и неуверенно излагает материал. Делает ошибки в ответах на уточняющие вопросы преподавателя

4.3. ТИПОВЫЕ ЗАДАНИЯ ДЛЯ КОНТРОЛЬНЫХ РАБОТ

Контрольные работы содержат несколько практических заданий по индивидуальным вариантам, в полном объеме охватывающих изученный материал по указанной теме. Выполнение контрольных работ позволяет определить результат освоения компетенций по дисциплине в рамках рассматриваемой темы, оцениваемый с помощью соответствующих индикаторов достижения компетенций.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
Тема 1.3 Встроенные функции	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	**Задание 3.1.** Установите правильную последовательность приоритета (от высшего к низшему) источников данных для суперглобального массива $_REQUEST при настройках PHP по умолчанию (request_order = "GP" в php.ini). Источники данных: 1. Данные из файлов, загруженных через $_FILES (метод multipart/form-data). 2. Данные, переданные через куки ($_COOKIE). 3. Данные, переданные методом GET ($_GET). 4. Данные, переданные методом POST ($_POST). Правильная последовательность: 432. Пояснение: При стандартной настройке request_order = "GP" порядок таков: сначала обрабатываются POST-данные (G), затем GET-данные (P). Куки (C) по умолчанию не включаются в $_REQUEST. Данные из $_FILES в $_REQUEST никогда не попадают. Правильный порядок для стандартных настроек: 4 (POST), 3 (GET). **Задание 3.2.** Расположите действия в правильном порядке для безопасной обработки данных формы, отправленной методом POST. Действия: 1. Валидация данных: проверка на соответствие ожидаемым типам, форматам и длине (например, filter_var() для email, preg_match() для шаблонов). 2. Выполнение основного действия с очищенными данными (сохранение в БД, отправка письма и т.д.). 3. Получение сырых данных из суперглобального массива $_POST. 4. Санитизация (очистка) данных: экранирование специальных символов для конкретного контекста (например, htmlspecialchars() для вывода в HTML, mysqli_real_escape_string() или подготовленные выражения для SQL). Правильная последовательность: 3142. Пояснение: Правильный порядок — это «защита в глубину»: Получаем сырые

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			данные (3) — изначальный ввод пользователя. Валидируем (1) — проверяем, что данные соответствуют ожиданиям (формат, тип, допустимые значения). Если нет — отклоняем. Санитизируем (4) — преобразуем данные для безопасного использования в конкретном контексте (SQL, HTML, и т.д.). Выполняем действие (2) — работаем с уже проверенными и очищенными данными. Валидация проверяет «что» ввели (корректность), санитизация — «как» использовать (безопасность). Сначала убеждаемся, что данные корректны, затем готовим их для конкретной операции. **Задание 3.3** Установите соответствие между встроенной функцией/константой и её категорией или назначением. Список 1 (Функция/Константа): A. __DIR__ B. phpinfo() C. isset() D. empty() Список 2 (Категория/Назначение): 1. Псевдоконстанта, возвращающая директорию текущего файла. 2. Функция для проверки существования и не-NULL значения переменной. 3. Функция для получения подробной информации о конфигурации PHP. 4. Функция для проверки, является ли переменная "пустой" (0, '', null, false, [] и т.д.). Ответ: A1, B3, C2, D4. **Задание 3.4** Установите соответствие между суперглобальным массивом и типом данных, который он обычно содержит. Список 1 (Суперглобальный массив): A. $_ENV

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			B. $_FILES C. $_SERVER D. $_COOKIE Список 2 (Тип данных): 1. Информация о загружаемых на сервер файлах (имя, тип, временный путь). 2. Переменные окружения операционной системы и веб-сервера. 3. HTTP-куки, отправленные клиентом. 4. Информация о сервере и среде выполнения (заголовки, пути, скрипты). Ответ: A2, B1, C4, D3. **Задание 3.5** Какая встроенная константа PHP содержит номер текущей строки в файле, в которой она была использована? Варианты ответов: 1. __LINE__ 2. __FILE__ 3. __FUNCTION__ 4. PHP_VERSION Ответ: 1. Пояснение: __LINE__ — это "магическая" псевдоконстанта, которая в момент выполнения содержит номер текущей строки в файле. **Задание 3.5** Какой оператор или функция НЕ входит в стандартный набор встроенных возможностей PHP для подключения содержимого другого файла в текущий скрипт? Варианты ответов: 1. require 2. include_once

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			3. import 3. require_once Ответ: 3. Пояснение: require, include_once, require_once — это все корректные языковые конструкции PHP для включения файлов.
Тема 2.2 Работа с файловой системой	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	**Задание 6.1.** Установите правильную последовательность безопасной обработки загружаемого на сервер изображения (аватара). Этапы обработки: 1. Проверить код ошибки загрузки в $_FILES['avatar']['error'] (должен быть UPLOAD_ERR_OK). 2. Сгенерировать уникальное имя файла (например, md5(uniqid()) . ".jpg") и определить безопасный путь для сохранения вне корневой веб-директории. 3. Проверить реальный MIME-тип файла с помощью finfo_file($finfo, $tmpPath) на соответствие image/jpeg, image/png и т.д. 4. Переместить временный файл из $_FILES['avatar']['tmp_name'] в итоговое место с помощью move_uploaded_file(). 5. Проверить наличие файла в массиве $_FILES и что загрузка вообще происходила методом POST. 6. Изменить размер изображения или создать миниатюру с помощью библиотеки GD или Imagick. Ответ: 513246. Пояснение: Проверки идут от общих к частным: существование запроса → техническая успешность → проверка содержимого → безопасное сохранение → дополнительная обработка. Тип файла проверяется после успешной загрузки, но до сохранения в постоянное место. **Задание 6.2** Расположите этапы в правильном порядке для рекурсивного удаления директории

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			со всем её содержимым с помощью встроенных функций PHP. Этапы: 1. Открыть директорию с помощью opendir(). 2. Для каждого элемента проверить: если это файл — удалить unlink(), если это директория — рекурсивно вызвать эту же процедуру. 3. Убедиться, что путь ведёт к существующей директории и это не символьная ссылка на опасный путь (базовая проверка безопасности). 4. Закрыть дескриптор директории с помощью closedir(). 5. Прочитать содержимое директории в цикле с помощью readdir(), пропуская . и ... 6. После очистки содержимого удалить саму пустую директорию с помощью rmdir(). Ответ: 315246. Пояснение: Сначала полностью очищаем все вложенные элементы (рекурсивно), и только потом удаляем саму родительскую директорию, когда она уже пуста. Безопасность проверяется в самом начале. **Задание 6.3** Установите соответствие между функцией PHP и её основным назначением при работе с файлами и директориями. Список 1 (Функция): A. scandir() B. is_uploaded_file() C. realpath() D. basename() Список 2 (Назначение): 1. Возвращает массив файлов и папок в указанной директории. 2. Проверяет, был ли файл загружен через HTTP POST (используется для безопасности перед move_uploaded_file). 3. Возвращает абсолютный канонизированный путь, убирая символы .., . и

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			символические ссылки. 4. Возвращает последний компонент имени из указанного пути (имя файла). Ответ: A1, B2, C3, D4. **Задание 6.4** Установите соответствие между режимом функции fopen() и его описанием. Список 1 (Режим): A. 'r' B. 'w' C. 'a' D. 'x' Список 2 (Описание): 1. Открывает файл только для записи; помещает указатель в конец файла. Если файл не существует, пытается его создать. 2. Открывает файл только для записи; обрезает файл до нулевой длины. Если файл не существует, пытается его создать. 3. Открывает файл только для чтения; помещает указатель в начало файла. 4. Создаёт и открывает файл только для записи; возвращает FALSE, если файл уже существует (защита от перезаписи). Ответ: A3, B2, C1, D4. **Задание 6.5** Какая функция является единственным безопасным способом переместить файл, загруженный через HTTP POST, из временной директории в постоянное место хранения на сервере? Варианты ответов: 1. copy($_FILES['file']['tmp_name'], $targetPath) 2. rename($_FILES['file']['tmp_name'], $targetPath) 3. move_uploaded_file($_FILES['file']['tmp_name'], $targetPath)

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			4. file_put_contents($targetPath, file_get_contents($_FILES['file']['tmp_name'])) Ответ: 3. Пояснение: move_uploaded_file() выполняет дополнительную критически важную проверку: она убеждается, что указанный файл ($tmp_name) действительно был загружен методом HTTP POST в текущем скрипте. Это предотвращает возможность скрипта манипулировать произвольными системными файлами (например, /etc/passwd), если злоумышленник подделает значение $_FILES. **Задание 6.6** Какой результат вернёт функция file_exists() для пути, который является символьной ссылкой (symlink) на несуществующий файл? Варианты ответов: 1. Всегда TRUE, так как ссылка существует. 2. Всегда FALSE, так как файл, на который она ссылается, не существует. 3. Зависит от операционной системы и прав доступа. 4. Вызовет ошибку E_WARNING. Правильный ответ: 2. Пояснение: Функция file_exists() проверяет существование файла или директории, на который указывает путь. Если путь является "битой" (dangling) символьной ссылкой (указывает на несуществующий целевой файл), функция вернёт FALSE. Для проверки существования самой ссылки как файла нужно использовать is_link().
Тема 3.1 Объектно-ориентированное программирование	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	**Задание 9.1** Какие из следующих утверждений о классах и объектах в PHP являются верными? Варианты ответов: 1. Класс — это чертёж (шаблон), а объект — его конкретная реализация в памяти во время выполнения программы. 2. Свойства класса должны всегда объявляться с модификатором доступа (public, protected, private), иначе возникнет ошибка.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			3. Метод __construct() вызывается автоматически при создании нового объекта оператором new. 4. Ключевое слово $this внутри метода класса ссылается на сам класс, а не на конкретный объект. Ответ: 1, 3. Пояснение: Правильный (1): Это классическое определение сути ООП. Класс описывает структуру и поведение, объект — конкретный экземпляр с уникальным состоянием. Правильный (3): __construct() — это "магический" метод-конструктор. Его вызов — часть процесса инстанцирования (создания) объекта. **Задание 9.2** Какие из следующих утверждений о наследовании и полиморфизме в PHP верны? Варианты ответов: 1. Класс-потомок (дочерний класс) наследует все public и protected методы и свойства родительского класса. 2. Чтобы переопределить (override) метод родительского класса в дочернем, необходимо использовать аннотацию @override. 3. Ключевое слово final, применённое к методу, запрещает его переопределение в дочерних классах. 4. Абстрактный класс может быть инстанцирован напрямую с помощью оператора new. Ответ: 1, 3. Пояснение: Правильный (1): Это базовый принцип наследования. private члены родительского класса не наследуются и недоступны в дочернем. Правильный (3): Модификатор final предотвращает переопределение метода в классах-наследниках. Если final применён к самому классу, то от него нельзя наследоваться.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			Задание 9.3 Установите правильную последовательность вызова "магических" методов при полном цикле создания, клонирования и уничтожения объекта. Последовательность вызова: 1. Вызов деструктора __destruct() оригинального объекта при завершении скрипта или unset. 2. Вызов конструктора __construct() для создания нового объекта. 3. Вызов метода __clone() для клонированного объекта (после операции clone $obj). Ответ: 231 Пояснение: конструктор вызывается при new. __clone() вызывается только для нового объекта-клона. Деструкторы вызываются автоматически в конце скрипта или при unset(). __toString() вызывается эпизодически при необходимости. Последовательность зависит от конкретного сценария, но логическая цепочка: создание → использование → (опционально клонирование) → уничтожение. **Задание 9.4** Расположите этапы в правильном порядке для разрешения (resolution) вызова метода $child->someMethod() в иерархии наследования, где ChildClass наследует от ParentClass. Этапы: 1. Интерпретатор PHP проверяет, существует ли метод someMethod() в классе ChildClass. 2. Если метод найден в ChildClass и он не private, выполняется он. 3. Если метод не найден в ChildClass, интерпретатор ищет его в родительском классе ParentClass. 4. Если метод не найден ни в одном из родителей в иерархии, возникает фатальная ошибка Fatal error: Uncaught Error: Call to undefined method. 5. Если метод найден в ParentClass и он private, возникает ошибка доступа. Ответ: 12354.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовой вариант контрольной работы
			Пояснение: PHP сначала проверяет класс, в контексте которого вызван метод (ChildClass), затем поднимается по иерархии наследования. Даже если метод существует в родителе, но объявлен как private, он не будет унаследован и недоступен из дочернего класса. **Задание 9.5** Как называется специальный метод в классе PHP, который автоматически вызывается при попытке выполнить операцию clone над объектом этого класса, и для чего он чаще всего используется? Ответ: __clone(). Пояснение: По умолчанию clone создает поверхностную (shallow) копию: все свойства копируются со своими значениями. Если свойство является ссылкой на другой объект, то копируется сама ссылка, и оба объекта ($original и $clone) начинают ссылаться на один и тот же вложенный объект. Метод __clone() позволяет переопределить это поведение и создать новые копии вложенных объектов. **Задание 9.6** Как называются методы в PHP, имена которых начинаются с двух символов подчёркивания (например, __construct, __get, __toString), которые предоставляют возможность перехватывать и переопределять различные операции над объектами? Ответ: "Магические" методы (Magic Methods). Пояснение: Эти методы не вызываются напрямую, а автоматически вызываются интерпретатором PHP в определённых ситуациях (при создании объекта, обращении к свойству, сериализации и т.д.). Они являются основой для реализации перегруженного поведения объектов в PHP.

Критерии оценивания контрольной работы:

Оценка	Критерии оценивания
отлично	Выставляется, если обучающийся умеет увязывать теорию с практикой (решает задачи, формулирует выводы, умеет пояснить полученные результаты), владеет понятийным аппаратом, полно и глубоко овладел материалом по заданной теме, обосновывает свои суждения и даёт правильные ответы на вопросы преподавателя`
хорошо	Выставляется, если обучающийся умеет увязывать теорию с практикой (решает задачи и формулирует выводы, умеет пояснить полученные результаты), владеет понятийным аппаратом, полно и глубоко овладел материалом по заданной теме, но содержание ответов имеют некоторые неточности и требуют уточнения и комментария со стороны преподавателя.
удовлетворительно	Выставляется, если обучающийся знает и понимает материал по заданной теме, но изложение неполное, непоследовательное, допускаются неточности в определении понятий, студент не может обосновать свои ответы на уточняющие вопросы преподавателя
неудовлетворительно	Выставляется, если обучающийся допускает ошибки в определении понятий, искажающие их смысл, беспорядочно и неуверенно излагает материал. Делает ошибки в ответах на уточняющие вопросы преподавателя

4.4. ТИПОВЫЕ ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Самостоятельная работа включает в себя проработку теоретического материала, изучение рекомендуемой литературы, выполнение практико-ориентированных заданий (заполнение таблиц, проведение сравнительного анализа, составление схем и др.), решение практических задач, создание презентаций, написание рефератов, подборка нормативного и иного материала и выполнение других заданий.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовое задание для самостоятельной работы
Тема 1.2 Пользовательские функции	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур	Напишите реферат на тему **Вариант 1** “Пользовательские функции. Уточнение типа аргумента» Что может быть отражено: Определение пользовательских функций. Это специально оформленные фрагменты кода, к которым можно обращаться из любого места внутри программы. Пользовательские функции упрощают работу разработчикам, улучшают читабельность кода и сокращают его. Особенности пользовательских функций. Например, в PHP к ним относят доступность параметров по умолчанию, иерархическую область видимости

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовое задание для самостоятельной работы
		данных, баз данных, программных интерфейсов	переменных внутри функции, способность возвращать любой тип и возможность менять переменные, которые переданы в качестве аргумента. Описание типа аргумента функции. В описании функции есть идентификатор, аргументы (входные параметры) и тип аргументов функции. Аргументы, указанные в описании функции, называются формальными, они получают смысл только после задания фактических аргументов при использовании функции. Напишите реферат на тему **Вариант 2** “Пользовательские функции. Области видимости» Что может быть отражено: В реферате по темам «Пользовательские функции» и «Область видимости переменных» можно рассмотреть следующие аспекты: Пользовательские функции позволяют повторно использовать один и тот же блок кода без копирования. Функции делают программу более читаемой и структурированной за счёт разделения логики на небольшие, понятные блоки. Можно рассказать о том, как создавать собственные функции, передавать в них параметры и возвращать значения. Также можно упомянуть о синтаксисе функции, например, о базовом определении функции в Python. Область видимости переменных определяет, где переменные доступны и где недоступны в программе. Под доступностью подразумевается, что можно считывать и изменять переменные. Можно рассказать о том, что область видимости может быть глобальной или локальной. Локальная область видимости, в свою очередь, бывает функциональной или блочной. Глобальная область видимости. Переменные, объявленные вне функций, находятся в глобальной области видимости и могут быть доступны в любой части программы. Однако использование глобальных переменных может привести к негативным последствиям, например, к сложности отладки и

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовое задание для самостоятельной работы
			нежелательным побочным эффектам. Локальная область видимости. Переменные, объявленные внутри функции или блока кода, являются локальными для этой функции или блока и не видны вне его. Это предотвращает конфликты имён и обеспечивает изоляцию данных и логики внутри функций. Важность области видимости. Область видимости позволяет скрывать переменные от внешнего кода, освобождать память, используемую для переменных, когда они выходят из области видимости. Замыкания в JavaScript. Замыкание — это функция, которая «замыкает» (захватывает) переменные из своей внешней области видимости, даже после завершения выполнения этой функции.
Тема 2.1 Протокол HTTP. Обработка форм	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	Напишите реферат на тему **Вариант 1** “Протокол HTTP. Буферизация» Что может быть отражено: Протокол HTTP (HyperText Transfer Protocol) — это набор правил, по которым браузер и сервер обмениваются данными. Некоторые аспекты протокола: Принцип «запрос-ответ». Браузер отправляет запрос на сервер, сообщая, что ему нужно. Сервер ищет нужный ресурс на своих дисках и, если находит, отправляет ответ обратно браузеру. Методы HTTP-запросов. Каждый из них предназначен для выполнения конкретной задачи при взаимодействии с ресурсом на сервере. Например, GET используется для запроса данных с сервера, POST — для отправки данных на сервер (например, из формы). Заголовки HTTP. Это дополнительные служебные параметры, которые передаются вместе с каждым запросом или ответом. Они содержат метаинформацию о содержимом, формате данных, кэшировании, авторизации и многом другом. HTTPS. Это защищённая версия протокола, которая использует шифрование данных с помощью SSL/TLS. HTTPS обеспечивает высокую безопасность,

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовое задание для самостоятельной работы
			защищая информацию от перехвата, подмены и гарантируя её целостность. Буферизация может быть раскрыта через призму технологии HTTP keep-alive. Она позволяет браузеру и серверу использовать одно и то же сетевое соединение для нескольких последовательных HTTP-запросов и ответов, значительно ускоряя загрузку страниц за счёт устранения необходимости переустанавливать соединение каждый раз. Напишите реферат на тему **Вариант 2** “Протокол HTTP. HTTP-аутентификация» Что может быть отражено: HTTP-аутентификация — процесс проверки подлинности пользователя при доступе к ресурсам веб-сервера с использованием протокола HTTP. Этот процесс обеспечивает контроль доступа к ресурсам и защиту данных на веб-сервере. Некоторые методы HTTP-аутентификации, которые можно рассмотреть в реферате: Базовая аутентификация (Basic Authentication). Клиент отправляет имя пользователя и пароль в закодированном виде (Base64) в заголовке «Authorization» с каждым запросом. Однако этот метод не считается безопасным, так как данные передаются в кодированной, а не зашифрованной форме. Аутентификация по токену (Bearer Authentication). Токен Bearer предоставляет доступ к определённому ресурсу или URL и обычно представляет собой зашифрованную строку, которая генерируется сервером в ответ на запрос авторизации. Клиент должен отправить этот токен в заголовке «Authorization» при запросе защищённых ресурсов. tune-it.ru Аутентификация по ключам доступа. Секретные данные — длинные

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовое задание для самостоятельной работы
			уникальные строки, замещающие имя и пароль. При создании ключа можно ограничить время его действия, ключ нельзя передавать третьим лицам или по небезопасным каналам. Также можно рассмотреть, как аутентификация и авторизация совмещены в HTTP-запросах: внутри запроса (параметров, тела или заголовков) передаются сразу как данные аутентификации, так и запрос на выполнение некоторых действий.
Тема 3.2 Классы и интерфейсы	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	Напишите реферат на тему **Вариант 1** “Классы, интерфейсы и уточнение типа (type-hinting)» Что может быть отражено: Классы — это структуры, которые определяют набор свойств и методов. Классы могут реализовывать интерфейсы, что позволяет использовать один интерфейс для разных целей. Например, класс может обрабатывать данные из базы данных и выводить их пользователю, а другой класс может сохранять данные в файл. Интерфейсы — это описание типа, набор компонентов, которые должен иметь тип данных. Интерфейсы определяют, как объекты могут взаимодействовать друг с другом. Например, можно определить интерфейс для класса, который обрабатывает данные из базы данных, и для класса, который выводит их пользователю. Type-hinting (подсказки типов) — это механизм, который помогает выявлять ошибки, улучшает читаемость, понятность и поддерживаемость кода. Подсказки типов есть в некоторых языках программирования, например в Python и PHP.

Тема	Код и формулировка компетенции	Индикаторы достижения компетенции	Типовое задание для самостоятельной работы
			Также можно рассмотреть примеры использования type-hinting, описать, для чего они применяются и в каких ситуациях используются. Напишите реферат на тему **Вариант 2** “Типажи (traits)» Что может быть отражено: Определение типажа. Типаж (от англ. trait) — абстрактный тип данных, который используется как простая концептуальная модель для структурирования объектно-ориентированных программ. Происхождение понятия. Концепция типажей появилась вследствие конфликтов при использовании классов в ООП для наследования. Особенности типажей. Типажи содержат только методы и не допускают совпадения их названий. Применение типажей. Они являются основой языка Rust, встроены в язык Scala, введены в Perl 6 и C# 8.0, реализованы во фреймворке Joose для JavaScript. Роль типажей. Помимо простого абстрагирования, типажи решают множество разнообразных проблем: используются как «маркеры» для типов, для объявления дополнительных методов, заменяют традиционную перегрузку методов и предоставляют простой способ перегрузки операторов. Примеры использования. Можно привести, например, типаж на Rust, описывающий хеширование.

Критерии оценивания самостоятельной работы:

Оценка	Критерии оценивания
отлично	выставляется если работа носит научно-исследовательский характер, проанализирован и сделан сравнительный анализ нескольких литературных источников, приведены примеры
хорошо	выставляется если проанализирован и сделан сравнительный анализ нескольких литературных источников, приведены примеры
удовлетворительно	выставляется если проведен сравнительный анализ научно-методической литературы, приведены примеры
неудовлетворительно	выставляется если работа прошла проверку на антиплагиат и соответствует требованиям оформления

4.5. ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ ДЛЯ КУРСОВЫХ РАБОТ

Курсовые работы не предусмотрены

Курсовая работа позволяет выявить степень владения базовыми знаниями, умениями и навыками, необходимыми для обучения, и определить уровень владения новым материалом.

Примерные индивидуальные задания (темы) для курсовых работ:

Критерии оценивания курсовой работы:

Оценка	Критерии оценивания
отлично	Содержание курсовой работы полностью соответствует заданию, содержащемуся в методических указаниях, и плану. Представлены результаты структурированного и логически последовательного обзора литературных и иных источников по теме исследования. Структура курсовой работы логически и методически выдержана. Верно определены исходные данные для расчетов. Все аналитические расчеты выполнены верно, корректно применены методы экономического анализа, не нарушена методика анализа предмета исследования. Все выводы и предложения убедительно аргументированы. При защите курсовой работы обучающийся правильно и уверенно отвечает на вопросы преподавателя, демонстрирует глубокое знание теоретического материала, способен аргументировать собственные утверждения и выводы
хорошо	Содержание курсовой работы полностью соответствует заданию, содержащемуся в методических указаниях, и плану. Представлены результаты структурированного и логически последовательного обзора литературных и иных источников по теме исследования. Структура курсовой работы логически и методически выдержана. Верно определены исходные данные для расчетов. В расчетах допускаются незначительные (не искажающие общего итога оценки) погрешности/ошибки. Большинство выводов и предложений аргументировано, корректно применены методы экономического анализа, не нарушена методика анализа предмета исследования. Оформление курсовой работы и полученные результаты в целом отвечают требованиям, изложенным в методических указаниях. Имеются одна-две несущественные ошибки в использовании терминов, в построенных диаграммах и схемах, в оформлении таблиц. Наличествует незначительное количество грамматических и/или стилистических ошибок. При защите курсовой работы обучающийся правильно и уверенно отвечает на большинство вопросов преподавателя, демонстрирует хорошее знание теоретического материала, но не всегда способен аргументировать

	собственные утверждения и выводы. При наводящих вопросах преподавателя исправляет ошибки в ответе
удовлетворительно	Содержание курсовой работы полностью соответствует заданию, содержащемуся в методических указаниях, и плану. Результаты обзора литературных и иных источников представлены недостаточно полно, недостаточно логично и последовательно. Верно определены исходные данные для расчетов, но имеются грубые ошибки в расчетах. Аргументация выводов и предложений слабая или отсутствует. Экономические выводы носят констатирующий (описательный) характер. Имеются одно-два существенных отклонений от требований в оформлении курсовой работы. Полученные результаты в целом отвечают требованиям, изложенным в методических указаниях. Имеются одна-две существенных ошибки в использовании терминов, в построенных диаграммах и схемах. Много грамматических и/или стилистических ошибок. При защите курсовой работы обучающийся допускает грубые ошибки при ответах на вопросы преподавателя, демонстрирует слабое знание теоретического материала, в большинстве случаев не способен уверенно аргументировать собственные утверждения и выводы
неудовлетворительно	Содержание курсовой работы не соответствует заданию, содержащемуся в методических указаниях, и плану. Неверно определены исходные данные для расчетов, неверно и не корректно применены методы экономического анализа. Экономические выводы содержат неверную экономическую оценку. Имеются более двух существенных отклонений от требований в оформлении курсовой работы. Большое количество существенных ошибок по сути работы, много грамматических и стилистических ошибок и др. Полученные результаты не отвечают требованиям, изложенным в методических указаниях. При защите курсовой работы обучающийся демонстрирует слабое понимание программного материала, студент не может защитить свои решения, допускает грубые фактические ошибки при ответах на поставленные вопросы или вовсе не отвечает на них. Курсовая работа не представлена преподавателю. Обучающийся не явился на защиту курсовой работы

5. МАТЕРИАЛЫ ДЛЯ ОЦЕНКИ РЕЗУЛЬТАТОВ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Промежуточная аттестация проводится в форме зачета/зачета с оценкой/экзамена.

5.1. Вопросы к зачету с оценкой:

1. Основы синтаксиса PHP. Типы данных PHP.
2. Переменные, константы, выражения.
3. Операторы.
4. Условный оператор и оператор выбора.
5. Циклы.
6. Объявление массивов. Обработка массивов.
7. Описание функции. Аргументы функции. Область видимости переменных. Возврат значений.

8. Встроенные функции. Встроенные константы и псевдоконстанты.
9. Функции подключения файлов.
10. Способы задания строк.
11. Функции для обработки строк.
12. Регулярные выражения.
13. Стандарт HTTP/1.1. Заголовки запроса и ответа. Статус сервера.
14. Доступ к заголовкам запроса.
15. Способы передачи данных на сервер. Обработка запросов с помощью PHP.
16. Проверка передаваемых значений в запросах.
17. Функции для работы с файлами.
18. Загрузка файлов на сервер.
19. Понятие cookie. Параметры cookie. Создание, чтение, удаление cookie.
20. Понятие сессии. Создание, чтение, удаление сессии.
21. Параметры сессии.
22. Объектно-ориентированное программирование в PHP. Классы.
23. Свойства и методы. Клонирование объектов.
24. Конструкторы и деструкторы.
25. Наследование
26. Перегрузка методов. Методы доступа к свойствам и методам.
27. Константы класса.
28. Абстрактные классы и методы
29. Интерфейсы.
30. Статические свойства и методы класса.
31. «Магические методы».
32. Пространства имен. Объявление.
33. Иерархия. Правила доступа.

Критерии оценивания зачета с оценкой/экзамена:

Оценка	**Критерии оценивания**

зачтено	Обучающийся должен: уметь строить ответ в соответствии со структурой излагаемого вопроса; продемонстрировать прочное, достаточно полное усвоение знаний программного материала; продемонстрировать знание основных теоретических понятий; правильно формулировать определения; последовательно, грамотно и логически стройно изложить теоретический материал; продемонстрировать умения самостоятельной работы с литературой; уметь сделать достаточно обоснованные выводы по излагаемому материалу.
не зачтено	Обучающийся демонстрирует: незнание значительной части программного материала; не владение понятийным аппаратом дисциплины; существенные ошибки при изложении учебного материала; неумение строить ответ в соответствии со структурой излагаемого вопроса; неумение делать выводы по излагаемому материалу.

Критерии оценивания зачета с оценкой/экзамена:

Оценка	Критерии оценивания
отлично	Выставляется, если обучающимся правильно и полностью раскрыто содержание материала в пределах программы, чётко и правильно даны определения и раскрыто содержание понятий, точно использованы научные и технические термины, в ответе использованы ранее приобретённые теоретические знания, сделаны необходимые выводы и обобщения
хорошо	Выставляется, если обучающимся раскрыто основное содержание материала в пределах программы, даны определения и раскрыто содержание понятий, в ответе использованы ранее приобретённые теоретические знания, сделаны необходимые выводы и обобщения, но присутствуют незначительные нарушения в последовательности изложения, имеются одна-две неточности в содержании ответа
удовлетворительно	Выставляется, если обучающимся содержание учебного материала изложено фрагментарно, не всегда последовательно, не даны определения, не раскрыто содержание понятий, или они изложены с ошибками, допускаются ошибки и неточности в использовании научной терминологии, отсутствуют выводы и обобщения из предыдущего материала, или возможны ошибки в их изложении
неудовлетворительно	Выставляется, если обучающимся основное содержание учебного материала не раскрыто, не даются ответы на основные вопросы, допускаются грубые ошибки в определении понятий, в использовании терминологии, отсутствуют выводы и обобщения

6. МАТЕРИАЛЫ ДЛЯ ДИАГНОСТИЧЕСКОЙ РАБОТЫ

Задания для диагностической работы должны обеспечивать оценку полностью или частично сформированных компетенций. Каждое задание должно быть привязано к тому или иному индикатору сформированности компетенций.

При формировании заданий для диагностической работы необходимо использовать тестовые задания следующих типов:

Тип задания 1. Задания закрытого типа на установление соответствия.

Тип задания 2. Задания закрытого типа на установление последовательности.

Тип задания 3. Задания комбинированного типа, предполагающие выбор одного правильного ответа из предложенных

Тип задания 4. Задания комбинированного типа, предполагающие выбор нескольких ответов из предложенных

Тип задания 5. Задания открытого типа с развернутым ответом.

№	Тема	Код	Индикатор	Тип	Задание	Уровень освоения

п/п	занятия	компетенции		задания			компетенции
					Вариант 1	Вариант 2	
1	**Тема 1.1** Основы PHP. Управляющие конструкции	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	4.4	**Задание 1** Какие из следующих утверждений о типах данных и операторах в PHP являются верными? Варианты ответов: 1. Оператор === сравнивает значения с приведением типов. 2. Результатом выражения (int) '12abc' будет целое число 12. 3. Константу, объявленную с помощью define(), можно переопределить в той же области видимости. 4. Выражение $a = 0; $b = '0'; при проверке if ($a == $b) вернёт true. Ответ: 2, 4. Пояснение: Правильный (2): При приведении строки, начинающейся с цифр, к целому типу (int), PHP берёт числовую часть с начала строки. Правильный (4): Оператор == выполняет сравнение с приведением типов. Строка '0' и число 0 при таком сравнении равны.	**Задание 1** Какие из следующих фрагментов кода приведут к выводу на экран всех элементов ассоциативного массива $data = ['name' => 'Ivan', 'age' => 25];? Варианты ответов: 1. php for ($i = 0; $i < count($data); $i++) { echo $data[$i]; } 2. php foreach ($data as $value) { echo $value; } 3. php foreach ($data as $key => $value) { echo "$key: $value"; } 4. php do { echo current($data); } while (next($data)); Ответ: 2, 3. Пояснение: Правильный (2): Цикл foreach корректно перебирает значения ассоциативного массива. Правильный (3): Цикл foreach корректно перебирает ключи и значения ассоциативного массива.	**Повышенный (3-5 минут)**
2	**Тема 1.2** Пользовательские	ПК-7 Способен осуществлят	ИПК-7.1 Знать методы и средства	2.2	**Задание 2.** Установите правильную последовательность этапов,	**Задание 2.** Расположите шаги в правильном порядке для получения результата	**Базовый (1-3 минуты)**

	функции	ь проектирование программных интерфейсов	проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов		которые выполняет интерпретатор PHP при вызове пользовательской функции.
					
					Этапы:
					1. Выполнение кода функции: вычисление выражений, работа с аргументами, выполнение управляющих конструкций.
					2. Возврат управления (и значения, если есть return) в точку вызова функции.
					3. Объявление функции (её код) считывается и сохраняется в памяти при первой встрече в процессе парсинга скрипта, но не выполняется.
					4. Передача управления в тело функции: создание локальной области видимости, инициализация аргументов переданными значениями.
					5. Встреча вызова функции (functionName(...)) в исполняемом коде.
					
					Правильная последовательность: 35412
					
					Пояснение: PHP работает в два прохода: сначала парсит весь скрипт (объявления), затем выполняет код последовательно. Функция не выполняется в момент объявления, а лишь «подготавливается» к будущим
						«Сумма: 30» из данного фрагмента кода.
						
						php
						function calculateSum($numbers) {
						$total = 0;
						foreach ($numbers as $num) {
						$total += $num;
						}
						return $total;
						}
						
						$data = [10, 20];
						// ... дальнейшие действия ...
						
						Шаги:
						1. Вызов функции calculateSum с аргументом $data.
						2. Объявление функции calculateSum.
						3. Присвоение возвращённого значения переменной, например, $result.
						4. Вывод на экран строки "Сумма: " и значения переменной $result.
						5. Создание массива $data со значениями [10, 20].
						
						Правильная последовательность: 25134
						
						Пояснение: PHP работает в два прохода: сначала парсит весь скрипт (объявления), затем выполняет код последовательно. Функция не выполняется в момент	

					вызовам.	объявления, а лишь «подготавливается» к будущим вызовам.	
3	**Тема 1.3** Встроенные функции	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов	1.1	**Задание 3.** Установите соответствие между встроенной функцией/константой и её категорией или назначением. Список 1 (Функция/Константа): A. __DIR__ B. phpinfo() C. isset() D. empty() Список 2 (Категория/Назначение): 1. Псевдоконстанта, возвращающая директорию текущего файла. 2. Функция для проверки существования и не-NULL значения переменной. 3. Функция для получения подробной информации о конфигурации PHP. 4. Функция для проверки, является ли переменная "пустой" (0, '', null, false, [] и т.д.). Ответ: A1, B3, C2, D4.	**Задание 3.** Установите соответствие между суперглобальным массивом и типом данных, который он обычно содержит. Список 1 (Суперглобальный массив): A. $_ENV B. $_FILES C. $_SERVER D. $_COOKIE Список 2 (Тип данных): 1. Информация о загружаемых на сервер файлах (имя, тип, временный путь). 2. Переменные окружения операционной системы и веб-сервера. 3. HTTP-куки, отправленные клиентом. 4. Информация о сервере и среде выполнения (заголовки, пути, скрипты). Ответ: A2, B1, C4, D3.	**Базовый (1-3 минуты)**
4	**Тема 1.4** Работа со	ПК-7 Способен осуществлят	ИПК-7.1 Знать методы и средства	3.3	**Задание 4.** Какой будет результат выполнения следующего кода?	**Задание 4.** Какой функцией следует воспользоваться, чтобы найти	**Повышенный (3-5 минут)**

	строками	ь проектирование программных интерфейсов	проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов		$str = "Hello\nWorld"; echo strlen($str); (Предполагаем, что строка содержит символ перевода строки \n). Варианты ответов: 1. 10 2. 11 3. 12 4. 2 Ответ: 2. Пояснение: Функция strlen() возвращает длину строки в байтах, а не в символах. Строка "Hello\nWorld" состоит из: H(1) + e(1) + l(1) + l(1) + o(1) + \n(1 байт) + W(1) + o(1) + r(1) + l(1) + d(1) = 11 байт.	последнее вхождение подстроки внутри строки, учитывая многобайтовую кодировку UTF-8? Варианты ответов: 1. strrpos() 2. mb_strrpos($str, $find, 0, 'UTF-8') 3. strpos() 4. mb_stripos($str, $find, 0, 'UTF-8') Ответ: 2. Пояснение: Для поиска последнего вхождения используется функция strrpos() (с двумя 'r'). Для работы с UTF-8 необходим её аналог из расширения mbstring — mb_strrpos(), где четвёртым аргументом указывается кодировка	
5	**Тема 2.1** Протокол HTTP. Обработка форм	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и	5.5	**Задание 5.** Как называется техника/уязвимость, при которой злоумышленник может внедрить и выполнить произвольный JavaScript-код на странице вашего сайта, используя некорректно обработанные данные, введённые другим пользователем? И какова основная встроенная функция PHP для защиты от неё при выводе	**Задание 5.** Какой метод HTTP является идемпотентным и безопасным (в терминологии RFC), не должен изменять состояние сервера и обычно используется только для получения данных? Ответ: GET. Пояснение: Согласно стандарту	**Высокий (5-10 минут)**

			средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов		данных в HTML? Ответ: Межсайтовый скриптинг (XSS — Cross-Site Scripting). Функция защиты — htmlspecialchars(). Пояснение: XSS — одна из самых распространённых веб-уязвимостей. Функция htmlspecialchars() преобразует специальные символы HTML (<, >, ", ', &) в их HTML-сущности, предотвращая их интерпретацию браузером как кода.	HTTP/1.1, методы GET и HEAD определяются как "безопасные" (safe), meaning they are intended only for information retrieval and should not change the state of the server. GET также является идемпотентным (многократный повтор одного и того же запроса даёт тот же результат). Это фундаментальное свойство, отличающее его от POST, PUT, DELETE.	
6	**Тема 2.2** Работа с файловой системой	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных,	2.2	**Задание 6.** Установите правильную последовательность безопасной обработки загружаемого на сервер изображения (аватара). Этапы обработки: 1. Проверить код ошибки загрузки в $_FILES['avatar']['error'] (должен быть UPLOAD_ERR_OK). 2. Сгенерировать уникальное имя файла (например, md5(uniqid()) . ".jpg") и определить безопасный путь для сохранения вне корневой веб-директории. 3. Проверить реальный MIME-тип файла с помощью finfo_file($finfo, $tmpPath) на соответствие image/jpeg, image/png и т.д.	**Задание 6.** Расположите этапы в правильном порядке для рекурсивного удаления директории со всем её содержимым с помощью встроенных функций PHP. Этапы: 1. Открыть директорию с помощью opendir(). 2. Для каждого элемента проверить: если это файл — удалить unlink(), если это директория — рекурсивно вызвать эту же процедуру. 3. Убедиться, что путь ведёт к существующей директории и это не символьная ссылка на опасный путь (базовая проверка	**Базовый (1-3 минуты)**

			программных интерфейсов		4. Переместить временный файл из $_FILES['avatar']['tmp_name'] в итоговое место с помощью move_uploaded_file(). 5. Проверить наличие файла в массиве $_FILES и что загрузка вообще происходила методом POST. 6. Изменить размер изображения или создать миниатюру с помощью библиотеки GD или Imagick. Ответ: 513246. Пояснение: Проверки идут от общих к частным: существование запроса → техническая успешность → проверка содержимого → безопасное сохранение → дополнительная обработка. Тип файла проверяется после успешной загрузки, но до сохранения в постоянное место.	безопасности). 4. Закрыть дескриптор директории с помощью closedir(). 5. Прочитать содержимое директории в цикле с помощью readdir(), пропуская . и ... 6. После очистки содержимого удалить саму пустую директорию с помощью rmdir(). Ответ: 315246. Пояснение: Сначала полностью очищаем все вложенные элементы (рекурсивно), и только потом удаляем саму родительскую директорию, когда она уже пуста. Безопасность проверяется в самом начале.	
7	**Тема 2.3** Cookie. Сессии	ПК-7 Способен осуществлять проектирование программных	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять	1.1	**Задание 7.** Установите соответствие между понятием/механизмом и его описанием. Список 1 (Понятие): A. Cookie (куки) B. Сессия (Session) C. Session ID D. $_SESSION	**Задание 7.** Установите соответствие между параметром функции setcookie() и его назначением. Список 1 (Параметр): A. expires (или time) B. path C. secure D. httponly	**Базовый (1-3 минуты)**

		интерфейсов	методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов		Список 2 (Описание): 1. Уникальный идентификатор, связывающий данные на сервере с конкретным браузером пользователя. 2. Суперглобальный массив PHP для хранения данных, которые должны сохраняться в течение всей сессии пользователя. 3. Механизм хранения состояния пользователя, при котором данные хранятся на сервере, а клиенту передаётся только идентификатор. 4. Небольшой фрагмент данных, отправляемый сервером браузеру и хранимый на стороне клиента; отправляется обратно с каждым запросом. Ответ: A4, B3, C1, D2.	Список 2 (Назначение): 1. Определяет, в течение какого времени (Unix timestamp) cookie будет действителен. 0 или время в прошлом удаляет cookie. 2. Указывает путь на сервере, для которого cookie будет доступен. '/' означает весь сайт. 3. Если true, cookie будет передаваться только по защищённому соединению HTTPS. 4. Если true, cookie будет недоступен для скриптового языка (например, JavaScript), что помогает снизить риск XSS-атак. Ответ: A-1, B-2, C-3, D-4.	
8	**Тема 2.4** Использование MySQL в PHP	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования	4.4	**Задание 8.** Какие из следующих утверждений о расширении MySQLi (MySQL Improved) являются верными? Варианты ответов: 1. MySQLi поддерживает как процедурный, так и объектно-ориентированный стиль программирования. 2. Функция mysqli_connect() возвращает объект соединения, который необходимо использовать	**Задание 8.** Какие из следующих действий являются обязательными или настоятельно рекомендуемыми для безопасного исполнения SQL-запросов с пользовательскими данными? Варианты ответов: 1. Проверять пользовательский ввод с помощью preg_match() на отсутствие SQL-ключевых слов (SELECT, UNION, DROP).	Высокий (5-10 минут)

			компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов		во всех последующих вызовах. 3. Для выполнения запроса SELECT и получения результата всегда необходимо использовать комбинацию mysqli_query() и mysqli_store_result(). 4. Подготовленные выражения (prepared statements) в MySQLi позволяют безопасно подставлять пользовательские данные в SQL-запрос, предотвращая SQL-инъекции. Ответ: 1, 4. Пояснение: Правильный (1): Это одно из ключевых преимуществ MySQLi. Можно использовать mysqli_ функции (процедурный стиль) или методы класса mysqli (ООП-стиль). Правильный (4): Это основное назначение подготовленных выражений. Параметры привязываются к запросу отдельно от его текста, что исключает интерпретацию данных как части SQL-команды.	2. Использовать подготовленные выражения ($stmt->prepare(), $stmt->bind_param(), $stmt->execute()). 3. Экранировать специальные символы в пользовательских строках с помощью mysqli_real_escape_string() перед подстановкой в SQL-строку. 4. Указывать кодировку соединения с базой данных как UTF-8 с помощью mysqli_set_charset($link, 'utf8mb4'). Ответ: 2, 4. Пояснение: Правильный (2): Подготовленные выражения — это единственный по-настоящему надёжный и рекомендуемый способ предотвращения SQL-инъекций в современных приложениях. Правильный (4): Установка корректной кодировки (предпочтительно utf8mb4 для полной поддержки Unicode) критически важна для предотвращения проблем с данными и является частью безопасной настройки соединения. Без неё даже mysqli_real_escape_string() может работать некорректно.	
9	**Тема 3.1** Объектно-	ПК-7 Способен	ИПК-7.1 Знать методы и	5.5	**Задание 9.** Как называется специальный	**Задание 9.** Как называются методы в PHP,	**Высокий (5-10 минут)**

	ориентированное программирование	осуществлять проектирование программных интерфейсов	средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов		метод в классе PHP, который автоматически вызывается при попытке выполнить операцию clone над объектом этого класса, и для чего он чаще всего используется? Ответ: __clone(). Пояснение: По умолчанию clone создает поверхностную (shallow) копию: все свойства копируются со своими значениями. Если свойство является ссылкой на другой объект, то копируется сама ссылка, и оба объекта ($original и $clone) начинают ссылаться на один и тот же вложенный объект. Метод __clone() позволяет переопределить это поведение и создать новые копии вложенных объектов.	имена которых начинаются с двух символов подчёркивания (например, __construct, __get, __toString), которые предоставляют возможность перехватывать и переопределять различные операции над объектами? Ответ: "Магические" методы (Magic Methods). Пояснение: Эти методы не вызываются напрямую, а автоматически вызываются интерпретатором PHP в определённых ситуациях (при создании объекта, обращении к свойству, сериализации и т.д.). Они являются основой для реализации перегруженного поведения объектов в PHP.	
10	**Тема 3.2** Классы и интерфейсы	ПК-7 Способен осуществлять проектирование программных интерфейсов	ИПК-7.1 Знать методы и средства проектирования программных интерфейсов ИПК-7.2 Уметь применять методы и средства	3.3	**Задание 10.** Какие из следующих утверждений о константах и статических членах класса в PHP являются верными? Варианты ответов: 1. Константы класса можно объявить с помощью ключевого слова const внутри класса. Они не могут быть изменены после объявления. 2. К статическому свойству можно	**Задание 10.** Какие из следующих утверждений об абстрактных классах и интерфейсах в PHP верны? Варианты ответов: 1. Абстрактный класс может содержать как абстрактные (без реализации), так и обычные (с реализацией) методы. 2. Интерфейс может содержать объявления свойств с	**Повышенный (3-5 минут)**

			проектирования компьютерного программного обеспечения, структур данных, баз данных, программных интерфейсов		получить доступ только через объект класса ($obj->staticProperty). 3. Статические методы могут использовать псевдопеременную $this для доступа к обычным (нестатическим) свойствам объекта. 4. Статическое свойство, в отличие от константы, может менять своё значение в ходе выполнения программы. Ответ: 1, 4. Пояснение: Правильный (1): Константы класса (const NAME = value;) действительно неизменяемы и связаны с классом, а не с объектом. Правильный (4): Статические свойства (static $property;) принадлежат самому классу и могут быть изменены, например, через self::$property = newValue;.	модификаторами доступа. 3.Класс может реализовывать (implements) несколько интерфейсов одновременно. 4. Интерфейс может содержать реализацию методов, если они объявлены как default. Ответ: 1, 3. Пояснение: Правильный (1): Это ключевое отличие абстрактного класса от интерфейса (в PHP до 8.0). Абстрактный класс может частично реализовывать функциональность. Правильный (3): Это одно из главных преимуществ интерфейсов перед одиночным наследованием классов. Класс может реализовать множество интерфейсов.	

Критерии оценивания диагностической работы:

Оценка	Критерии оценивания
отлично	от 90 до 100 % правильно выполненных заданий

хорошо	от 70 до 89 % правильно выполненных заданий
удовлетворительно	от 50 до 69 % правильно выполненных заданий
неудовлетворительно	менее 50 % правильно выполненных заданий