**Комплект оценочных материалов по дисциплине**

**«WEB технологии в системах автоматизации»**

**Задания закрытого типа**

**Задания закрытого типа на выбор правильного ответа**

*Выберите один правильный ответ*

1. Какой HTML-тег используется для создания гиперссылки?

А) <link>

Б) <a>

В) <href>

Г) <url>

Правильный ответ: Б

Компетенции (индикаторы) ПК-4

2. Какой HTML-тег используется для создания нумерованного списка?

А) <ul>

Б) <dl>

В) <ol>

Г) <list>

Правильный ответ: В

Компетенции (индикаторы) ПК-4

3. Какое CSS-свойство изменяет внешние отступы элемента?

А) margin

Б) padding

В) border

Г) spacing

Правильный ответ: А

Компетенции (индикаторы) ПК-4

4. Какой из перечисленных вариантов обладает максимальным приоритетом при применении стилей в CSS?

А).item:hover

Б) div#container > ul li

В) ul li.item

Г) !important

Правильный ответ: Г

Компетенции (индикаторы) ПК-4

5. Что выведет код в консоль: console.log(0.1 + 0.2 === 0.3)?

А) false

Б) true

В) Ошибку

Г) undefined

Правильный ответ: А

Компетенции (индикаторы) ПК-4

6. Какой синтаксис является стандартным и рекомендуемым способом добавления обработчика события в современном JavaScript?

А) element.addEvent("click", handler)

Б) element.onClick(handler)

В) element.addEventListener("click", handler)

Г) element.attachEvent("click", handler)

Правильный ответ: В

Компетенции (индикаторы) ПК-4

7. Какой необходимо применить селектор в jQuery для получения коллекции идентификатора «username», расположенного в таблице?

А) $('table #username')

Б) #('table $username')

В) $('#username table')

Г) #('$username table')

Правильный ответ: А

Компетенции (индикаторы) ПК-4

8. Какой SQL-запрос используется для выборки всех записей из таблицы «users», при этом отсортировав их по полю «rDate» в порядке убывания, и выведя только те записи, где статус пользователя равен «blocked»?

А) SELECT \* FROM users WHERE status="blocked" ORDER BY rDate DESC

Б) GET ALL FROM users SORT BY rDate DESC WHERE status="blocked"

В) SHOW users ORDER BY rDate DESC WHERE status="blocked"

Г) SELECT \* FROM users ORDER BY rDate DESC AND status="blocked"

Правильный ответ: А

Компетенции (индикаторы) ПК-4

9. Какое регулярное выражение корректно соответствует любым действительным числам, включая отрицательные, положительные, с плавающей точкой?

А) /-?\d+(\.\d+)?/

Б) /[0-9]+(\.[0-9]+)?/

В) /\d+/

Г) /-?[0-9]?/

Правильный ответ: А

Компетенции (индикаторы) ПК-4

10. Как получить значение переменной сессии в PHP?

А) session.var

Б) $\_SESSION['var']

В) get\_session('var')

Г) session['var']

Правильный ответ: Б

Компетенции (индикаторы) ПК-4

**Задания закрытого типа на установление соответствия**

*Установите правильное соответствие.*

*Каждому элементу левого столбца соответствует только один элемент правого столбца.*

1. Установите соответствие между тегами и их описанием:

|  |  |
| --- | --- |
| Тег | Описание |
| 1) <strong> | А) определяет заголовок документа |
| 2) <select> | Б) создает поле ввода формы |
| 3) <input> | В) создает выпадающий список |
| 4) <title> | Г) определяет важность текста и визуально выделяет его жирным |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Г | В | Б | А |

Компетенции (индикаторы) ПК-4

2. Соотнесите CSS-свойства с их функциями:

|  |  |
| --- | --- |
| CSS-свойство | Назначение |
| 1) margin | А) управляет тенью элемента |
| 2) box-shadow | Б) определяет внутренние отступы элемента |
| 3) display: none | В) определяет внешние отступы элемента |
| 4) padding | Г) скрывает элемент |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| В | А | Г | Б |

Компетенции (индикаторы) ПК-4

3. Соотнесите псевдоэлементы с их назначением:

|  |  |
| --- | --- |
| Псевдоэлемент | Назначение псевдоэлемента |
| 1) ::before | А) добавляет декоративный или информационный контент перед содержимым выбранного элемента |
| 2) ::selection | Б) стилизует выделенный пользователем текст на странице |
| 3) ::first-letter | В) стилизует первую букву текста |
| 4) ::placeholder | Г) стилизует подсказку в поле ввода текста (<input> и др.) |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| А | Б | В | Г |

Компетенции (индикаторы) ПК-4

4. Сопоставьте методы массивов JavaScript с их описанием:

|  |  |
| --- | --- |
| JavaScript метод | Описание метода |
| 1) join() | А) создает новый массив на основе преобразования каждого элемента |
| 2) pop() | Б) удаляет последний элемент массива (и возвращает его) |
| 3) unshift() | В) добавляет элемент(ы) в начало массива |
| 4) map() | Г) превращает массив в строку через разделитель |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Г | Б | В | А |

Компетенции (индикаторы) ПК-4

5. Соотнесите методы HTTP с их назначением:

|  |  |
| --- | --- |
| HTTP метод | Назначение метода |
| 1) GET | А) изменение существующего ресурса |
| 2) POST | Б) удаление ресурса |
| 3) PUT | В) получение данных |
| 4) DELETE | Г) отправка данных на сервер |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| В | Г | А | Б |

Компетенции (индикаторы) ПК-4

6. Соотнесите регулярные выражения с их назначением:

|  |  |
| --- | --- |
| Регулярное выражение | Назначение выражения |
| 1) /\s+/ | А) любые символы слова |
| 2) /^\d+$/ | Б) пробельные символы |
| 3) /[A-Za-z]+/ | В) только цифры |
| 4) /\w+/ | Г) только буквы |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Б | В | Г | А |

Компетенции (индикаторы) ПК-4

7. Соотнесите уровни ошибок PHP:

|  |  |
| --- | --- |
| Название ошибки | Значение ошибки |
| 1) E\_DEPRECATED | А) предупреждение о нарушении в работе кода, при котором выполнение программы может продолжаться |
| 2) E\_ERROR | Б) сигнализирует об использовании устаревших функций и конструкций, которые могут быть удалены в будущих версиях PHP |
| 3) E\_NOTICE | В) фатальная ошибка, после которой дальнейшее выполнение сценария становится невозможным |
| 4) E\_WARNING | Г) незначительная ошибка, не создает серьезных проблем |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Б | В | Г | А |

Компетенции (индикаторы) ПК-4

8. Соотнесите функции для работы с датой в PHP:

|  |  |
| --- | --- |
| Название функции | Назначение функции |
| 1) date() | А) текущее время в секундах |
| 2) time() | Б) создание timestamp из отдельных значений |
| 3) strtotime() | В) форматирование даты |
| 4) mktime() | Г) преобразование строки в timestamp |

Правильный ответ

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| В | А | Г | Б |

Компетенции (индикаторы) ПК-4

**Задания закрытого типа на установление правильной последовательности**

*Установите правильную последовательность.*

*Запишите правильную последовательность букв слева направо.*

1. Установите правильную последовательность этапов обработки HTTP-запроса браузером:

А) DNS-запрос для получения IP-адреса сервера

Б) получение HTTP-ответа (HTML, CSS, JS и т.д.)

В) рендеринг страницы браузером

Г) установка TCP-соединения (и TLS-шифрования, если HTTPS)

Д) отправка HTTP-запроса (например, GET с заголовком Host)

Правильный ответ: А, Г, Д, Б, В

Компетенции (индикаторы) ПК-4

2. Определите правильную последовательность тегов для построения HTML-таблицы:

А) <tbody>

Б) <td>Данные</td>

В) <tr>

Г) <table>

Правильный ответ: Г, А, В, Б

Компетенции (индикаторы) ПК-4

3. Расположите следующие CSS-селекторы и способы применения стилей в порядке увеличения их специфичности (от наименьшего приоритета к наибольшему):

А) div

Б) .item

В) #navbar

Г) style="color: red"

Правильный ответ: А, Б, В, Г

Компетенции (индикаторы) ПК-4

4. Установите правильную последовательность этапов при обработке события в JavaScript:

А) Выполнение кода внутри обработчика

Б) Вызов callback-функции (обработчика события)

В) Назначение обработчика (addEventListener)

Г) Поиск элемента (querySelector / getElementById и т. д.)

Правильный ответ: Г, В, Б, А

Компетенции (индикаторы) ПК-4

5. Установите правильный порядок загрузки файлов на сервер в PHP:

А) проверка загруженного файла

Б) установка настроек в php.ini

В) отправка файла через форму

Г) сохранение файла на сервере

Правильный ответ: Б, В, А, Г

Компетенции (индикаторы) ПК-4

6. Установите правильный порядок возникновения языков и WEB технологий:

А) JavaScript

Б) Vue.js

В) HTML

Г) jQuery

Правильный ответ: В, А, Г, Б

Компетенции (индикаторы) ПК-4

7. Расположите регулярные выражения по минимальному количеству символов, которые могут быть выбраны в нежадном режиме работы, от меньшего к большему:

А) /\d+\d+/

Б) /\d\*\d+/

В) /\d?\d?/

Г) /\d{3,4}/

Правильный ответ: В, Б, А, Г

Компетенции (индикаторы) ПК-4

8. Расположите этапы работы с токенами в Token Based Authentication в правильном порядке:

А) проверка токена на сервере

Б) генерация токена

В) отправка токена с каждым запросом

Г) сохранение токена на клиенте

Правильный ответ: Б, Г, В, А

Компетенции (индикаторы) ПК-4

**Задания открытого типа**

**Задания открытого типа на дополнение**

*Напишите пропущенное слово (словосочетание)*

1. Тег <tr> определяет \_\_\_\_\_\_\_ таблицы.

Правильный ответ: строку

Компетенции (индикаторы) ПК-4

2. Псевдокласс :nth-child(2n) применяет стили к \_\_\_\_\_\_\_\_\_\_ элементам в родительском контейнере.

Правильный ответ: чётным

Компетенции (индикаторы) ПК-4

3. При использовании CSS-свойства «\_\_\_\_\_\_\_\_\_\_\_: all 0.5s;» плавный переход всех анимируемых свойств элемента будет длиться 500 миллисекунд.

Правильный ответ: transition

Компетенции (индикаторы) ПК-4

4. Что будет выведено в консоль при выполнении следующего кода?

const str = "Hello World";

console.log(str.slice(6, 9).toUpperCase());

Правильный ответ: WOR

Компетенции (индикаторы) ПК-4

5. В PHP цикл, предназначенный для перебора ассоциативных массивов, называется \_\_\_\_\_\_\_\_\_\_.

Правильный ответ: foreach

Компетенции (индикаторы) ПК-4

6. В PHP для создания сессии используется функция \_\_\_\_\_\_.

Правильный ответ: session

Компетенции (индикаторы) ПК-4

7. В Yii для работы с базой данных через объекты используется класс \_\_\_\_\_\_.

Правильный ответ: ActiveRecord

Компетенции (индикаторы) ПК-4

8. Вид атаки, при которой злоумышленник внедряет вредоносный SQL-код через входные данные веб-формы, называется SQL-\_\_\_\_\_\_\_\_\_.

Правильный ответ: инъекция

Компетенции (индикаторы) ПК-4

**Задания открытого типа с кратким свободным ответом**

*Напишите пропущенное слово (словосочетание)*

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ — это программный интерфейс для работы с HTML- и XML-документами, который представляет их в виде древовидной структуры узлов (nodes). Позволяет программам и скриптам (например, JavaScript) динамически изменять содержимое, структуру и стили веб-страницы.

Правильный ответ: DOM / DOM-дерево / DOM-модель / объектная модель документа / Дерево DOM / Document Object Model

Компетенции (индикаторы) ПК-4

2. С помощью атрибута src в теге <img> указывается \_\_\_\_\_\_\_\_\_ изображения.

Правильный ответ: путь / источник / URL / адрес / URL-адрес / местоположение / ссылка

Компетенции (индикаторы) ПК-4

3. CSS-свойство \_\_\_\_\_\_\_\_\_\_\_\_\_\_ позволяет задать величину внутренних отступов элемента web-страницы

Правильный ответ: padding / padding-top / padding-right / padding-bottom / padding-left

Компетенции (индикаторы) ПК-4

4. Ключевое слово \_\_\_\_\_\_ используется в JavaScript для объявления переменной.

Правильный ответ: var / let / const

Компетенции (индикаторы) ПК-4

5. Какие основные языки и технологии можно использовать для создания клиентской части реактивных веб-приложения?

Правильный ответ: JavaScript / React / Angular / Vue / Vue.js

Компетенции (индикаторы) ПК-4

6. Какой метод HTTP может быть использован для отправки данных на сервер?

Правильный ответ: POST / PUT / PATCH / DELETE / GET

Компетенции (индикаторы) ПК-4

7. Какую библиотеку можно использовать для получения AJAX данных с сервера?

Правильный ответ: Axios /jQuery / Fetch API / Fetch / Superagent / XHR

Компетенции (индикаторы) ПК-4

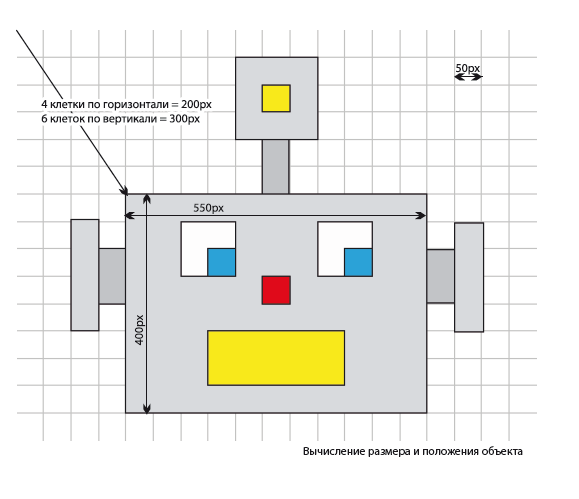
8. Какой протокол может быть применен для обеспечения безопасности передачи данных в WEB приложении?

Правильный ответ: HTTPS / TLS / SSL / Websocket over TLS / SFTP

Компетенции (индикаторы) ПК-4

**Задания открытого типа с развернутым ответом**

1. Разработать веб-страницу, на которой с помощью HTML и CSS воссоздать изображение робота из предоставленной картинки. Все элементы робота должны быть расположены с использованием относительного позиционирования (position: relative).



Время выполнения – 60 мин

Ожидаемый результат:

HTML-структура страницы:

<div class="robot">

    <div class="robot\_\_antenna">

        <div class="robot\_\_antenna-part1">

            <div class="robot\_\_antenna-part2"></div>

        </div>

        <div class="robot\_\_antenna-part3"></div>

    </div>

    <div class="robot\_\_head">

        <div class="robot\_\_ear">

            <div class="robot\_\_ear-part robot\_\_ear-part\_large"></div>

            <div class="robot\_\_ear-part robot\_\_ear-part\_small"></div>

        </div>

        <div class="robot\_\_face">

            <div class="robot\_\_face-eyes">

                <div class="robot\_\_face-eye">

                    <div class="robot\_\_face-eye-pupil"></div>

                </div>

                <div class="robot\_\_face-eye">

                    <div class="robot\_\_face-eye-pupil"></div>

                </div>

            </div>

            <div class="robot\_\_face-nose"> </div>

            <div class="robot\_\_face-mouth"></div>

        </div>

        <div class="robot\_\_ear">

            <div class="robot\_\_ear-part robot\_\_ear-part\_small"></div>

            <div class="robot\_\_ear-part robot\_\_ear-part\_large"></div>

        </div>

    </div>

</div>

CSS для стилизации веб-страницы:

\* {

    box-sizing: border-box;

}

body {

    margin: 0;

}

.robot {

    display: flex;

    flex-direction: column;

    align-items: center;

    margin-top: 50px;

}

.robot \* {

    position: relative;

}

/\* АНТЕННА \*/

.robot\_\_antenna {

    display: flex;

    flex-direction: column;

    align-items: center;

}

.robot\_\_antenna-part1 {

    display: flex;

    align-items: center;

    justify-content: center;

    background-color: lightgrey;

    border: 2px solid #000;

    height: 150px;

    width: 150px;

}

.robot\_\_antenna-part2{

    background-color:yellow  ;

    border: 2px solid #000;

    height: 50px;

    width: 50px;

}

.robot\_\_antenna-part3 {

    border: 2px solid #000;

    border-top: none;

    border-bottom: none;

    background-color:lightgray;

    height: 100px;

    width: 50px;

}

/\* АНТЕННА \*/

.robot\_\_head {

    display: flex ;

}

/\* ЛИЦО \*/

.robot\_\_face {

    display: flex;

    flex-direction: column;

    align-items: center;

    background-color: lightgrey;

    border: 2px solid #000;

    height: 400px;

    width: 550px;

}

.robot\_\_face-eyes {

    display: flex;

    margin-top: 50px;

    justify-content: space-between;

    height: 100px;

    width: 350px;

}

.robot\_\_face-eye {

    display: flex;

    align-items: flex-end;

    justify-content: end;

    background-color: #fff;

    border: 2px solid #000;

    height: 100px;

    width: 100px;

}

.robot\_\_face-eye-pupil {

    background-color: skyblue;

    border: 2px solid #000;

    border-right: none;

    border-bottom: none;

    height: 50px;

    width: 50px;

}

.robot\_\_face-nose {

    background-color: red;

    border: 2px solid #000;

    height: 50px;

    width: 50px;

}

.robot\_\_face-mouth {

    background-color: yellow;

    margin-top: 50px;

    border: 2px solid #000;

    height: 100px;

    width: 250px;

}

/\* ЛИЦО \*/

/\* УШИ \*/

.robot\_\_ear {

    display: flex;

    align-items: center;

    align-self: start;

    margin-top: 50px;

}

.robot\_\_ear-part {

    justify-content: space-between;

    background-color: lightgrey;

    border: 2px solid #000;

}

.robot\_\_ear-part\_small {

    border-right: none;

    border-left: none;

    height: 100px;

    width: 50px;

}

.robot\_\_ear-part\_large {

    height:200px;

    width: 50px;

}

/\* УШИ \*/

Критерии оценивания:

* создана HTML-структура с контейнером для робота и его частями;
* написаны CSS-стили для стилизации и относительного позиционирования частей робота;
* элементы страницы соответствуют указанным пропорциям;
* отсутствие наложений или несоответствий в расположении элементов;
* логичность и структурированность кода.

Компетенции (индикаторы) ПК-4

2. Создайте таймер обратного отсчета, который показывает оставшееся время до заданной даты (например, до Нового года). Таймер должен обновляться каждую секунду.

Время выполнения – 50 мин

Ожидаемый результат:

HTML-структура:

<div class="container">

    <h1>До Нового 2026 года осталось:</h1>

    <div class="timer">

        <div class="timer-block">

            <span id="days">00</span>

            <p>Дней</p>

        </div>

        <div class="timer-block">

            <span id="hours">00</span>

            <p>Часов</p>

        </div>

        <div class="timer-block">

            <span id="minutes">00</span>

            <p>Минут</p>

        </div>

        <div class="timer-block">

            <span id="seconds">00</span>

            <p>Секунд</p>

        </div>

    </div>

</div>

CSS для стилизации элементов:

body {

    font-family: 'Arial', sans-serif;

    display: flex;

    justify-content: center;

    align-items: center;

    height: 100vh;

    margin: 0;

    background: linear-gradient(135deg, #1e3c72, #2a5298);

    color: white;

}

.container {

    text-align: center;

}

h1 {

    margin-bottom: 30px;

    font-size: 2.5rem;

}

.timer {

    display: flex;

    justify-content: center;

    gap: 20px;

}

.timer-block {

    background: rgba(0, 0, 0, 0.3);

    padding: 20px;

    border-radius: 10px;

    min-width: 100px;

}

.timer-block span {

    font-size: 3rem;

    font-weight: bold;

    display: block;

}

.timer-block p {

    margin: 5px 0 0;

    font-size: 1.2rem;

}

JavaScript-код таймера:

// Целевая дата - 1 января 2026 года

const targetDate = new Date('2026-01-01T00:00:00').getTime();

function updateTimer() {

    const now = new Date().getTime();

    const diff = targetDate - now;

    // Если время вышло

    if (diff <= 0) {

        document.querySelector('.timer').innerHTML =

            '<div class="timer-block"><span>С Новым Годом!</span></div>';

        return;

    }

    // Расчет дней, часов, минут и секунд

    const days = Math.floor(diff / (1000 \* 60 \* 60 \* 24));

    const hours = Math.floor((diff % (1000 \* 60 \* 60 \* 24)) / (1000 \* 60 \* 60));

    const minutes = Math.floor((diff % (1000 \* 60 \* 60)) / (1000 \* 60));

    const seconds = Math.floor((diff % (1000 \* 60)) / 1000);

    // Обновление интерфейса

    document.getElementById('days').textContent =

        days < 10 ? `0${days}` : days;

    document.getElementById('hours').textContent =

        hours < 10 ? `0${hours}` : hours;

    document.getElementById('minutes').textContent =

        minutes < 10 ? `0${minutes}` : minutes;

    document.getElementById('seconds').textContent =

        seconds < 10 ? `0${seconds}` : seconds;

}

// Запуск таймера (обновление каждую секунду)

setInterval(updateTimer, 1000);

// Первоначальный вызов, чтобы избежать задержки в 1 секунду

updateTimer();

Критерии оценивания:

* создана HTML-структура с контейнером таймера и блоками для дней, часов, минут, секунд;
* написаны CSS-стили для оформления таймера (по желанию);
* реализован корректный расчет оставшегося времени с обновлением каждую секунду;
* добавлена проверка на дату в прошлом и вывод соответствующего сообщения;
* код логично структурирован, без избыточных элементов, с понятными названиями переменных.

Компетенции (индикаторы) ПК-4

3. Создайте простую форму обратной связи с валидацией на стороне клиента с помощью jQuery.

Время выполнения – 30 мин

Ожидаемый результат:

HTML-структура формы:

<form id="feedback-form">

<div class="form-group">

<label for="name">Имя:</label>

<input type="text" id="name" name="name" required>

</div>

<div class="form-group">

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

</div>

<div class="form-group">

<label for="message">Сообщение:</label>

<textarea id="message" name="message" required></textarea>

</div>

<button type="submit">Отправить</button>

</form>

CSS для стилизации ошибок:

.error {

color: red;

margin-bottom: 5px;

}

input.invalid, textarea.invalid {

border-color: red;

}

JavaScript с использованием jQuery:

$(document).ready(function() {

$('#feedback-form').on('submit', function(e) {

$('.error').remove()

$('input, textarea').removeClass('invalid')

// Получаем значения полей

let name = $('#name').val()

let email = $('#email').val()

let message = $('#message').val()

// Флаг валидности формы

let formIsValid = true

// Валидация имени

if (!name) {

showError('#name', 'введите имя')

formIsValid = false

}

// Валидация email

if (!email || !validateEmail(email)) {

showError('#email', 'введите корректный email')

formIsValid = false

}

// Валидация сообщения

if (!message) {

showError('#message', 'введите сообщение')

formIsValid = false

}

// Если форма валидна - отправляем её

if (formIsValid) {

alert('Форма успешно отправлена!')

} else {

return false

}

});

// Функция показа ошибки

function showError(field, message) {

$(field)

.addClass('invalid')

.after('<div class="error">' + message + '</div>')

}

// Функция проверки email

function validateEmail(email) {

const re = /^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/

return re.test(email)

}

});

Критерии оценивания:

– создана HTML форма;

– прописано событие обработки формы по готовности страницы;

– предусмотрен механизм отображения ошибок ввода;

– программа не отправляет данные на сервер при ошибках ввода.

Компетенции (индикаторы) ПК-4

4. Опишите принцип взаимодействия клиентов с сервером через cookies и сессии.

Время выполнения – 30 мин

Ожидаемый результат:

Принцип взаимодействия клиентов с сервером через cookies и сессии основан на необходимости отслеживания пользователя при посещении веб-сайта. Поскольку HTTP является протоколом без сохранения состояния, сервер не может “запомнить” пользователя при переходе между страницами. Для решения этой проблемы используются два основных механизма: Cookies и session.

Cookies – небольшие фрагменты данных, которые хранятся на компьютере пользователя. Сервер отправляет cookies браузеру в заголовках ответа. Браузер сохраняет их и автоматически отправляет обратно серверу при последующих запросах.

Сессии – механизм идентификации пользователя на сервере. При первом обращении сервер генерирует уникальный идентификатор, чаще всего session\_id. Этот идентификатор передается клиенту, обычно через cookies. Сервер создает файл с данными сессии на своем сервере.

При последующих запросах клиент отправляет идентификатор сессии, обычно через cookies.

Общий процесс взаимодействия выглядит следующим образом. Клиент делает первый запрос на сервер. Сервер генерирует session\_id и отправляет его клиенту через cookies, при этом создает файл сессии на сервере. Клиент сохраняет cookies с session\_id. Потом отправляет этот идентификатор при каждом последующем запросе.

Сервер получает session\_id и использует его для доступа к данным пользователя. Обрабатывает запрос с учетом сохраненной информации.

Критерии оценивания:

– пояснено необходимость использования cookies и сессий;

– пояснено применение cookies;

– пояснено применение сессий;

– дана общая последовательность применения cookies и сессий для успешного управления данными пользователя.

Компетенции (индикаторы) ПК-4

5. Создайте регулярное выражение для проверки корректности email-адреса. Объясните каждый компонент вашего выражения и его назначение.

Время выполнения – 30 мин

Пример решения:

Существует много требований к email адресу, чтобы он соответствовал всем стандартам. Основные требования к электронному адресу:

– структура адреса должна включать имя (перед символом @), сам символ @, доменную часть (после символа @);

– имя должно содержать латинские буквы (a-z), цифры (0-9), специальные символы: точка, дефис, нижнее подчеркивание (\_).

– доменная часть должна содержать: доменное имя и доменную зону.

Если составить регулярное выражение для проверки email по этим требованиям, то получится:

^[a-z0-9\_.-]+@([a-z0-9-]+\.)+[a-z]{2,6}$

В начале регулярного выражения символ «^» – начало строки, который указывает, что проверка должна начинаться с начала строки/

Часто выражения «[a-z0-9\_.-]+» – имя, записанное до знака @. Квадратные скобки указывают из каких элементов может состоять символ повторения. «a-z0-9» – любые буквы и цифры, «\_.-» специальные символы, а символ «+» означает один или более символов, перечисленных ранее.

Далее идет символ «@», который находится в email адресе.

Остальная часть регулярного выражения содержит доменную часть.

Последовательность «([a-z0-9-]+\.)+» формирует имя поддомена которое состоит из чередующихся через точку слов из букв и цифр.

Заключительная часть домена, верхний уровень, описывается выражением «[a-z]{2,6}», где {2,6} – от 2 до 6 символов.

Символ «$» в конце выражения – конец строки, указывает, что проверка должна заканчиваться в конце строки

Критерии оценивания:

– приведены основные требования к email адресам;

– разработано регулярное выражение по заданию;

– описаны все части регулярного выражения.

Компетенции (индикаторы) ПК-4

6. Создайте собственную базовую структуру MVC-приложения. Объясните назначение каждого компонента и их взаимодействие.

Время выполнения – 30 мин

Ожидаемый результат:

Базовая структура MVC-приложения – модель (Model) + представление (View) + контроллер (Controller).

Модель предназначена хранить (управлять) данные приложения, реализовывать бизнес-логику отдельных узлом приложения, определять структуру данных.

Представление необходимо для отображения данных пользователю, создания пользовательского интерфейса.

Контроллер ведет обработку пользовательского ввода и выполняет координацию взаимодействия между Моделью и Представлением. Управляет потоком данных в целом по приложению.

Реализуем простую MVC структуру.

**Файл** Модели:

class Task:

def \_\_init\_\_(self, title, description):

self.title = title

self.description = description

self.completed = False

def mark\_as\_completed(self):

self.completed = True

**Файл** Представления:

<div>

<h2>{{ task.title }}</h2>

<p>{{ task.description }}</p>

<button onclick="completeTask()">Mark as Completed</button>

</div>

**Файл** Контроллера:

class TaskController:

def show\_task(self, task\_id):

task = Task.get(task\_id)

return render\_template('task.html', task=task)

def complete\_task(self, task\_id):

task = Task.get(task\_id)

task.mark\_as\_completed()

task.save()

Сначала польз**ователь отправляет запрос, который поступает контроллеру.** Контроллер получает и обрабатывает входные данные. Контроллер запрашивает необходимые данные из Модели. Модель возвращает необходимые данные. Контроллер определяет дальнейшие действия. Контроллер передает данные в Представление. Представление формирует отображаемый интерфейс. Представление отдает данные контроллеру. Контроллер отдает ответ пользователю.

Такое разделение позволяет вести независимую разработку отдельных компонентов приложения разработчиками разных квалификаций единый программный продукт.

Критерии оценивания:

– дано базовое представление о приложениях с MVC структурой;

– приведены примеры кода по каждой части приложения;

– описана последовательность взаимодействия компонентов.

Компетенции (индикаторы) ПК-4