**Паспорт Образовательной программы**

**«Введение в веб-разработку на языке JavaScript»**

|  |  |
| --- | --- |
| **Версия программы** | **1** |
| **Дата Версии** | **14.10.2020** |

1. **Сведения о Провайдере**

|  |  |  |
| --- | --- | --- |
| 1.1 | Провайдер | **АНО ДПО "СофтЛайн Эдюкейшн"** |
| 1.2 | Логотип образовательной организации |  |
| 1.3 | Провайдер ИНН | **7736228783** |
| 1.4 | Ответственный за программу ФИО | **Максименко Денис Владиславович** |
| 1.5 | Ответственный должность | Руководитель проектного офиса |
| 1.6 | Ответственный Телефон | +7-495-232-0023 доб. 0889 |
| 1.7 | Ответственный Е-mail | Edu2035@softline.com |

1. **Основные Данные**

|  |  |  |
| --- | --- | --- |
| **№** | **Название** | **Описание** |
| 2.1 | Название программы | Введение в веб-разработку на языке JavaScript |
| 2.2 | Ссылка на страницу программы | <https://go.teachbase.ru/course_sessions/vhodnoe-testirovanie-kurs-vvedenie-v-veb-razrabotku-na-yazyke-javascript-11-01/apply> - входное тестирование  <https://go.teachbase.ru/course_sessions/vvedenie-v-veb-razrabotku-na-yazyke-javascript-11-01/apply> - страница программы |
| 2.3 | Формат обучения | Электронное обучение |
|  | Подтверждение от ОО наличия возможности реализации образовательной программы с применением электронного обучения и (или) дистанционных образовательных технологий с возможностью передачи данных в форме элементов цифрового следа | Подтверждаем возможность реализации образовательной программы с применением электронного обучения с возможностью передачи данных в форме элементов цифрового следа |
| 2.4 | Уровень сложности | Базовый |
| 2.5 | Количество академических часов | 72 академических часа |
|  | Практикоориентированный характер образовательной программы: не менее 50 % трудоёмкости учебной деятельности отведено практическим занятиям и (или) выполнению практических заданий в режиме самостоятельной работы | 50 ак. часов отведено практическим занятиям и и выполнению практических заданий в режиме самостоятельной работы, что составляет около 69,4 % трудоёмкости учебной деятельности; |
| 2.6 | Стоимость обучения одного обучающегося по образовательной программе, а также предоставление ссылок на 3 (три) аналогичные образовательные программы иных организаций, осуществляющих обучение, для оценки объективности стоимости или обоснование уникальности представленной образовательной программы в случае отсутствия аналогичных образовательных программ на рынке образовательных услуг | 29900   1. <https://www.luxoft-training.ru/kurs/razrabotka_na_javascript.html>   стоимость: 36 500 руб ( 36 ак часов)   1. <https://netology.ru/programs/web-developer>   Стоимость: 110 000 руб   1. <https://skillfactory.ru/javascript>   Стоимость: 55 000 руб |
| 2.7 | Минимальное количество человек на курсе | 1 |
| 2.8 | Максимальное количество человек на курсе | 1000 |
| 2.9 | Данные о количестве слушателей, ранее успешно прошедших обучение по образовательной программе | отсутствует |
| 2.10 | Формы аттестации | **тестирование** |
|  | Указание на область реализации компетенций цифровой экономики, к которой в большей степени относится образовательная программа, в соответствии с Перечнем областей | Программирование и создание ИТ-продуктов |

1. **Аннотация программы**

Согласно githut.info – статистике самого известного сервиса для хранения исходного кода приложений GitHub, JavaScript – это язык, на котором написан код в самом большом числе репозиториев – свыше 300 000. Вместе с классическими языками C и Java он уже несколько лет входит в десятку самых популярных языков по версии рейтинга TIOBE. Как указывается на stackoverflow.com, уже восьмой год подряд JavaScript сохраняет свое прочное положение как наиболее часто используемый язык программирования (67%). Своей популярностью JavaScript обязан не только простоте и выразительности, обеспечивающим низкий порог вхождения и возможность быстро начать решать реальные профессиональные задачи, но также универсальности и незаменимости – это единственный официальный язык браузера. На нём создаются пользовательские интерфейсы, обрабатываются события и реализуются визуальные эффекты. Также это единственный язык программирования в мире, который можно без специальных преобразований использовать для написания кода и на стороне клиента, и на стороне сервера, чтобы получить полноценное изоморфное приложение. Все это обосновывает актуальность программы повышения квалификации, которая направлена на совершенствование профессиональной компетенции «Способен ставить и решать прикладные задачи по созданию и совершенствованию программного кода с использованием специализированных программных средств на примере инструментов экосистемы JavaScript». Для успешного освоения программы обучающимся необходимо владеть базовыми компетенциями цифровой грамотности, в том числе навыками владения ПК на уровне среднего или продвинутого пользователя; компетенции в области алгоритмизации (знание основных алгоритмических структур, умения выражать их на псевдокоде и в виде блок-схем) и основ программирования. В процессе обучения по программе обучающимся рекомендуется прослушать все видеолекции и видеоролики по практическим занятиям, выполнить по каждой теме практические задания с автоматизированной проверкой, проработать самостоятельно учебную литературу и информационные ресурсы и выполнить задания для самостоятельной работы, после этого необходимо выполнить тесты по каждой теме и для успешного завершения обучения выполнить итоговый тест (70% верных ответов). Обучающийся в результате освоения программы: будут знать синтаксис языка программирования JavaScript и особенности программирования на JavaScript, особенности выбранной среды программирования, стандартные библиотеки языка программирования JavaScript, технологии программирования на языке JavaScript, место JavaScript среди современных интерпретируемых языков и сценарных языков программирования; будут уметь применять JavaScript для написания программного кода, использовать выбранную среду программирования (редактор кода) для создания сценариев, использовать возможности имеющейся программной архитектуры веб-приложения; будут владеть навыками создания программного кода на JavaScript в соответствии с спецификациями; методами совершенствования программного кода с использованием специализированных программных средств на примере линтеров JavaScript; способами определения и манипулирования данными на JavaScript и JSON; навыками размещения программного кода в страницах, созданных при верстке информационных ресурсов (в клиентской части веб-ресурса). В процессе освоения программы обучающиеся научатся создавать модули и слушатели событий, узнают отличия коллбэка от замыкания и статической типизации от динамической, получат опыт транспиляции кода, содержащего новейшие характеристики языка, научатся самостоятельно компилировать проект на TypeScript. В качестве бонуса в курсе предлагается развёртывание простого серверного веб-приложения на JavaScript. Результаты обучения по программе будут востребованы при решении следующих профессиональных задач: создание программного кода в соответствии с спецификациями и его совершенствование с использованием специализированных программных средств на примере линтеров JavaScript; написание программного кода на языках JavaScript и JSON, определение и манипулирование данными; размещение программного кода в клиентской части веб-ресурса.

**Автономная некоммерческая организация**

**дополнительного профессионального образования**

**«СофтЛайн Эдюкейшн»**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ:  Генеральный директор  АНО ДПО «СофтЛайн Эдюкейшн»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Э. Разуваев  «\_14\_» \_\_Октября\_\_\_ 2020\_ г. |

**Дополнительная профессиональная программа повышения квалификации**

**«Введение в веб-разработку на языке JavaScript»**

72 часа

Москва, 2020

**ОБЩАЯ ХАРАКТЕРИСТИКА ПРОГРАММЫ**

**1.Цель программы**

Формирование профессиональных компетенций, необходимых для решения прикладных задач по созданию и совершенствованию программного кода с использованием специализированных программных средств на примере инструментов экосистемы JavaScript

**2.Планируемые результаты обучения:**

2.1.Знание (осведомленность в областях)

2.1.1 синтаксиса языка программирования JavaScript и особенностей программирования на JavaScript;

2.1.2 особенностей выбранной среды программирования;

2.1.3 стандартных библиотек языка программирования JavaScript;

2.1.4 технологии программирования на языке JavaScript;

2.1.5 место JavaScript среди современных интерпретируемых языков и сценарных языков программирования.

2.2. Умение (способность к деятельности)

2.2.1 применять JavaScript для написания программного кода;

2.2.2 использовать выбранную среду программирования (редактор кода) для создания сценариев;

2.2.3 использовать возможности имеющейся программной архитектуры веб-приложения;

2.3.Навыки (использование конкретных инструментов)

2.3.1 создания программного кода на JavaScript в соответствии с спецификациями;

3.3.2. владения методами совершенствования программного кода с использованием специализированных программных средств на примере линтеров JavaScript;

2.3.3. владения способами определения и манипулирования данными на JavaScript и JSON;

2.3.4. размещения программного кода в страницах, созданных при верстке информационных ресурсов (в клиентской части веб-ресурса).

**3. Категория слушателей** (возможно заполнение не всех полей)

* 1. Образование: среднее профессиональное, высшее
  2. Квалификация: уровень квалификации не ниже 4
  3. Наличие опыта профессиональной деятельности: опыт разработки программного обеспечения, электронных образовательных ресурсов и информационных систем
  4. Предварительное освоение иных дисциплин/курсов /модулей:

**4.Учебный план программы «Введение в веб-разработку на языке JavaScript»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Модуль** | **Всего, час** | **Виды учебных занятий** | | |
| **лекции** | **практические занятия** | **самостоятельная работа** |
|  | Модуль 1. Введение в JavaScript | **4** | 2 | 1 | 1 |
|  | Модуль 2. Синтаксис JavaScript. Управление пакетами и зависимостями | **9** | 3 | 5 | 1 |
|  | Модуль 3.  Переменные и типы данных | **7** | 3 | 4 |  |
|  | Модуль 4. Функции в JavaScript | **7** | 2 | 4 | 1 |
|  | Модуль 5. Управляющие конструкции | **5** | 1 | 3 | 1 |
|  | Модуль 6.Строки, бинарные данные и регулярные выражения | **7** | 2 | 4 | 1 |
|  | Модуль 7. Массивы и объекты | **11** | 3 | 8 |  |
|  | Модуль 8. Модули и транспиляция. DOM | **6** | 2 | 3 | 1 |
|  | Модуль 9. Клиентские фреймворки | **7** | 2 | 4 | 1 |
|  | Модуль 10. Внебраузерный JavaScript. Прототипирование серверного приложения | **8** | 2 | 4 | 2 |
| **Итоговая аттестация** | | **1** | **Тест, зачет** | | **1** |
| **ИТОГО** | | **72** | **22** | **40** | **10** |

**5.Календарный план-график реализации образовательной программы**

(дата начала обучения – дата завершения обучения) в текущем календарном году, указания на периодичность набора групп (не менее 1 группы в месяц)

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Наименование учебных модулей** | **Трудоёмкость (час)** | **Сроки обучения** |
| **1** | Модуль 1. Введение в JavaScript | 4 | **01.11** |
| **2** | Модуль 2. Синтаксис JavaScript. Управление пакетами и зависимостями | 9 | **02.11-03.11** |
| **3** | Модуль 3.  Переменные и типы данных | 7 | **04.11-05.11** |
| **4** | Модуль 4. Функции в JavaScript | 7 | **05.11-06.11** |
| **5** | Модуль 5. Управляющие конструкции | 5 | **06.11-07.11** |
| **6** | Модуль 6.Строки, бинарные данные и регулярные выражения | 7 | **07.11-08.11** |
| **7** | Модуль 7. Массивы и объекты | 11 | **08.11-10.11** |
| **8** | Модуль 8. Модули и транспиляция. DOM | 6 | **10.11-11.11** |
| **9** | Модуль 9. Клиентские фреймворки | 7 | **11.11-12.11** |
| **10** | Модуль 10. Внебраузерный JavaScript. Прототипирование серверного приложения | 8 | **13.11-14.11** |
| **11** | Итоговая аттестация | 1 | **15.11** |
| **Всего:** | | **72** | **15 календарных дней** |

**6.Учебно-тематический план программы «Введение в веб-разработку на языке JavaScript»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Модуль / Тема** | **Всего, час** | **Виды учебных занятий** | | | **Формы контроля** |
| **лекции** | **практические занятия** | **самостоятельная работа** |
| 1 | **Модуль 1. Введение в JavaScript** | **4** | 2 | 1 | 1 | - |
| 1.1 | Тема 1. Эволюция JavaScript | **4** | 2 | 1 | 1 | тест |
| 2 | **Модуль 2. Синтаксис JavaScript. Управление пакетами и зависимостями** | **9** | 3 | 5 | 1 | - |
| 2.1 | Тема 2.1. Синтаксис и основные понятия JavaScript | 6 | 2 | 3 | 1 | тест |
| 2.2 | Тема 2.2. Управление пакетами и зависимостями | 3 | 1 | 2 | - | тест |
| 3 | **Модуль 3.**  **Переменные и типы данных** | **7** | 3 | 4 | - | - |
| 3.1 | Тема 3.1. Переменные и области видимости | 2 | 1 | 1 | - | тест |
| 3.2 | Тема 3.2. Примитивные и объектные типы данных | 2 | 1 | 1 | - | тест |
| 3.3 | Тема 3.3. TypeScript и статическая типизация | 3 | 1 | 2 | - | тест |
| 4 | **Модуль 4. Функции в JavaScript** | **7** | 2 | 4 | 1 | - |
| 4.1 | Тема 4.1. Функции как структурный элемент сценария | 3 | 1 | 2 | - | тест |
| 4.2 | Тема 4.2.Функции как тип данных | 4 | 1 | 2 | 1 | тест |
| 5 | **Модуль 5. Управляющие конструкции** | **5** | 1 | 3 | 1 | тест |
| 5.1 | Тема 5.1. Ветвление | 3 | 1 | 1 |  | - |
| 5.2 | Тема 5.2. Циклы | 2 | 1 | 1 | 1 | - |
| **6** | **Модуль 6. Строки, бинарные данные и регулярные выражения** | **7** | 2 | 4 | 1 | - |
| 6.1 | Тема 6.1. Строки и бинарные данные | 4 | 1 | 2 | 1 | тест |
| 6.2 | Тема 6.2. Регулярные выражения | 3 | 1 | 2 |  | тест |
| **7** | **Модуль 7. Массивы и объекты** | **11** | 3 | 8 |  | **-** |
|  | Тема 7.1. Массивы и множества | 3 | 1 | 2 |  | тест |
|  | Тема 7.2. Литеральные объекты. Свойства и методы | 5 | 1 | 4 |  | тест |
|  | Тема 7.3. Прототипы и конструкторы | 3 | 1 | 2 |  | тест |
|  | **Модуль 8. Модули и транспиляция. DOM** | **6** | 2 | 3 | 1 | - |
|  | Тема 8.1. Модули и транспиляция. DOM | 6 | 2 | 3 | 1 | тест |
|  | **Модуль 9. Клиентские фреймворки** | **7** | 2 | 4 | 1 | тест |
|  | Тема 9.1. Клиентские фреймворки: особенности | 2 | 1 | 1 | - | - |
|  | Тема 9.2. Примеры клиентских фреймворков | 5 | 1 | 3 | 1 | - |
|  | **Модуль 10. Внебраузерный JavaScript. Прототипирование серверного приложения** | **8** | 2 | 4 | 2 | тест |
|  | Тема 10.1. Внебраузерный JavaScript | 4 | 1 | 2 | 1 | **-** |
|  | Тема 10.2. Прототипирование серверного приложения | 4 | 1 | 2 | 1 | **-** |
|  | **Итоговая**  **аттестация** | 1 |  |  | 1 | **Итоговый тест** |

**7. Учебная (рабочая) программа повышения квалификации «Введение в веб-разработку на языке JavaScript»**

**Модуль 1. Введение в JavaScript (4 час.)**

Тема 1. Эволюция JavaScript(4 час.)

История JavaScript. Современная экосистема JavaScript. Организация рабочего места для выполнения сценариев на JavaScript. Инструменты, среды и методологии программирования на JavaScript

**Модуль 2. Синтаксис JavaScript. Управление пакетами и зависимостями (9 час.)**

Тема 2.1. Синтаксис и основные понятия JavaScript (6 час.)

Низкоуровневый синтаксис – лексическая структура, кодировки. Выражения и инструкции. Структура программы

Тема 2.2. Управление пакетами и зависимостями (3 час.)

Подключение зависимостей в браузере и вне браузера. Менеджмент зависимостей и подключение библиотек.

Девелоперские зависимости. Система npm и программа yarn

**Модуль 3. Переменные и типы данных (7 час.)**

Тема 3.1. Переменные и области видимости (2 час.)

Понятие переменной, объявления, присваивания и переприсваивания. Область видимости и подъём. Блоки и изоляция

Тема 3.2. Примитивные и объектные типы данных (2 час.)

Примитивные и ссылочные типы данных. Явное и неявное преобразование типов, undefined и null, логические значения

Тема 3.3. TypeScript и статическая типизация (3час.)

Статическая и динамическая типизация в программировании. Аннотации типов. TypeScript. Flow

**Модуль 4. Функции в JavaScript (7 час.)**

Тема 4.1. Функции как структурный элемент сценария (3 час.)

Понятие функции, обычные и стрелочные функции. IIFE. Сигнатура функции и аргументы. Контекст this

Тема 4.2.Функции как тип данных (4 час.)

Сущности первого класса. Лямбды и коллбэки. Синхронные и асинхронные коллбэки

**Модуль 5. Управляющие конструкции (5 час)**

Тема 5.1. Ветвление (3 час.)

Линейная организация кода. Ветвление

Тема 5.2. Циклы (2час.)

Циклы в императивном и функциональном стиле

**Модуль 6. Строки, бинарные данные и регулярные выражения (7 час.)**

Тема 6.1. Строки и бинарные данные (4 час.)

Строки и кодировки. Методы строк (класса String). Цепочки методов

Тема 6.2. Регулярные выражения (3 час.)

Использование регулярных выражений в вебе. Валидация полей формы. Поиск и замена

**Модуль 7. Массивы и объекты (11 час.)**

Тема 7.1. Массивы и множества (3 час.)

Массивы в сравнении со строками. Методы массивов. Цепочки методов

Тема 7.2. Литеральные объекты. Свойства и методы (5 час.)

Ключи и значения. Формат JSON. Иммутабельность, энумерабельность, итерабельность. Контекст this

Тема 7.3. Прототипы и конструкторы (3 час.)

Понятия прототипа, класса и конструктора. Цепочка прототипов. Мутация прототипов

**Модуль 8. Модули и транспиляция. DOM (6 час.)**

Тема 8.1. Модули и транспиляция. DOM (6 час.)

Модули в современном ECMAScript. Транспиляция между версиями ECMAScript. Сборка кода для браузера. DOM

**Модуль 9. Клиентские фреймворки (7 час.)**

Тема 9.1. Клиентские фреймворки: особенности (2 час)

Веб-фрейморк, типы. Компоненты в современном фронтэнде.

Тема 9.2. Примеры клиентских фреймворков (5 час.)

Создание приложений на React в браузере. Создание приложений на Vue в браузере

**Модуль 10. Внебраузерный JavaScript. Прототипирование серверного приложения (8 час.)**

Тема 10.1. Внебраузерный JavaScript (4 час.)

Фронтэнд и бэкэнд на JavaScript. Моноязыковая платформа разработка. Примеры работы с файловой системой

Тема 10.2. Прототипирование серверного приложения (4 час.)

Создание простейшего веб-сервера с помощью модуля http

Итоговая аттестация (1 час.)

**Описание практико-ориентированных заданий и кейсов**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Номер темы/модуля** | **Наименование практического занятия** | **Описание** |
| **1.1** | **Модуль 1. Тема 1.** | Современная экосистема JavaScript | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  *Пример задания.* Анализ версий JavaScript и инструментов. Составление таблицы.  Практические задания на определение версии программного обеспечения  *Пример задания (по видеолекции).* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – intro-001\_ (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите версию движка V8, которая используется в Node.js 14.13.0. Пример оформления:  // здесь ваше решение из одной строки, например, X.Y.ABC.DE-node.MN |
| **2.1** | **Модуль 2. Тема 2.1** | 1.Низкоуровневый синтаксис  2.Структура программы: элемент script | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Отработка основных понятий JavaScript  Практические задания на отработку умения использовать синтаксис экранирования символов Юникода  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – low-001\_ (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите строковое выражение из 8 символов, которое с помощью синтаксиса экранирования символа Юникода задаёт восклицательный знак из набора ASCII. Пример оформления:  \/u0201\  Практические задания на отработку умения подключать сценарий к веб-странице  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – low-002\_ (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите элемент script так, как он должен был бы выглядеть в разделе head веб-страницы, чтобы подключать файл https://kodaktor.ru/j/myscript. Пример оформления:  <script from=”https://kodaktor.ru/j/myscript”><\script> |
| **2.2** | **Модуль 2. Тема 2.2** | 1.Подключение зависимостей  2. Система npm | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой*. Практические задания на отработку умения создавать файл описания проекта по умолчанию  *Пример задания.* В отдельной пустой папке (на своём компьютере, на платформе Node.js) выполните команду npm init -y и посмотрите содержимое созданного файла package.json. Вам нужно выяснить, какой ключ был автоматически создан в разделе "scripts". Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, электронный адрес для отсылки Вам письма с результатом выполнения задания (если оно не содержит фатальных ошибок), ключ (код) задания – dep-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей в виде строки найденную вами версию пакета – например, в теле функции будет написано return 'abc'.  Практические задания на отработку умения подключать npm-пакет  *Пример задания.* В отдельном проекте (на своём компьютере, на платформе Node.js) установите пакет nodemon версии 1.18.3 (проверьте отдельно, какая версия является самой свежей) и узнайте, от какой версии пакета chokidar он зависит. Вы найдёте ответ на вопрос в папке node\_modules. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, электронный адрес для отсылки Вам письма с результатом выполнения задания (если оно не содержит фатальных ошибок), ключ (код) задания – dep-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей в виде строки найденную вами версию пакета – например, в теле функции будет написано return '34.8.12'. |
| **3.1** | **Модуль 3. Тема 3.1** | 1.Работа с объявлением переменных  2.Работа с присваиванием значения переменной | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения объявлять переменную  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – parse-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), тело которой содержит строку с объявлением переменной z без присвоения значения с помощью ключевого слова let. В данном случае функция ничего не должна возвращать. Пример оформления:  function task(x) {  // здесь ваше решение из одной строки  }  (Т.е. в общей сложности описание функции task состоит из 3 (трёх) строк, в первой объявление функции и открывающая фигурная скобка, в третьей закрывающая фигурная скобка, вторая строка представляет собой ваше решение задания.)  *Пример задания.*  Практическое задание на отработку умения присваивать значение уже объявленной ранее переменной. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – parse-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), тело которой содержит строку с объявлением переменной z без присвоения значения с помощью ключевого слова let. В данном случае функция ничего не должна возвращать. Пример оформления:  function task(x) {  // здесь ваше решение из одной строки  }  (Т.е. в общей сложности описание функции task состоит из 3 (трёх) строк, в первой объявление функции и открывающая фигурная скобка, в третьей закрывающая фигурная скобка, вторая строка представляет собой ваше решение задания.) |
| **3.2** | **Модуль 3. Тема 3.2** | 1. Значения null и undefined  2. Логические значения | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения работать со значениями undefined и null.  *Пример задания*. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – types-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который использует тернарный оператор для того, чтобы выдать ответ в виде строки, является ли предложенное значение значением null (выдаёт 'null'), либо undefined (выдаёт 'undefined'), либо NaN (выдаёт 'not a number'). В остальных случаях должно возвращаться значение 'Other'. Например, для значения '#' должно выдаваться 'Other'.  Практическое задание на отработку умения работать с логическими значениями  *Пример задания.* Практическое задание на отработку умения работать с логическими значениями. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – types-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который возвращает логическое отрицание аргумента x с помощью оператора «восклицательный знак» |
| **3.3** | **Модуль 3. Тема 3.3** | 1.TypeScript  2.Flow | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения работать с объявлениями типов в TypeScript  *Пример задания.*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который возвращает в виде строки ключевое слово TypeScript, которое используется для объявления интерфейса.  Практические задания на отработку умения устанавливать flow  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который возвращает в виде строки полное название npm-пакета, содержащего официальный пресет транспилятора babel для работы с flow. |
| **4.1** | **Модуль 4. Тема 4.1** | 1.Взвращение значения функции  2.Контекст this | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения работать с логическими значениями  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – func-a (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), которая по строке – аргументу x – возвращает эту строку, преобразованную целиком в верхний регистр.  Практические задания на отработку умения работать с контекстом function-функции  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – func-b (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), которая по числу – аргументу x – возвращает произведение этого числа на квадрат контекста. |
| **4.2** | **Модуль 4. Тема 4.2** | 1.Сущности первого класса  2.Коллбэки | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения работать с функцией как сущностью первого класса  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который возвращает функцию от двух аргументов, возвращающую их сумму.  Практические задания на отработку умения работать с функцией как с вызываемым типом данных (коллбэком)  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания func\_012 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей результат работы функции, переданной ей в качестве аргумента x и строку 'no' если типом x не является функция. |
| **5.1** | **Модуль 5. Тема 5.1** | Ветвление | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умений работать с операторами ветвления  *Пример задания*  1. В этом задании вам нужно продемонстрировать знакомство с конструкцией ветвления (if) в JavaScript.  2. Перейдите по адресу  <https://node-server.online/r> и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – if-001 (выбрать из выпадающего списка) – и пароль (nodetasker).  3. Введите код, выполняющий задание:  тело функции task, которая по аргументу x, который содержит некоторую строку, возвращает значение true, если строка x имеет длину 5 и код её первого символа равен 0, и false, если строка x имеет длину 3 и код её первого символа равен 1.  Пример оформления решения:  function task(x) { /\* решение \*/ } |
| **5.2** | **Модуль 5. Тема 5.2** | Циклический алгоритм | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения осуществлять вычисление с помощью циклического алгоритма  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который возвращает сумму всех аргументов, фактически переданных функции.  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания cycl-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей строку 'prime', если ей в качестве аргумента x передано простое число, и строку 'nonprime' если составное число. Рассматриваются целые числа от 3 до 10000 включительно. В остальных случаях предусмотрено значение 'incorrect'. |
| **6.1** | **Модуль 6. Тема 6.1** | Методы строк  Цепочка методов | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения осуществлять операции над строками  *Пример задания*  Практическое задание на отработку умения осуществлять операции над строками. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания str-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей true, если строка x является палиндромом и false в противном случае.  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания str-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке сумму десятичных кодов её символов. Например, по строке AZ должен получиться результат 155 = 65 + 90 |
| **6.2** | **Модуль 6. Тема 6.2** | Поиск по шаблону  Валидация строки | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения применять регулярные выражения  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания reg-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий true, только если в качестве аргумента x передана строка, которая содержит корректный IP-адрес версии 4 без маски.  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания reg-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий true, только если в качестве аргумента x передана строка, которая начинается и кончается заглавными латинскими буквами, а между ними находятся только цифры в количестве от 4 до 8, иначе false. |
| **7.1** | **Модуль 7. Тема 7.1** | Создание и обработка массивов | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения применять методы создания массивов  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания arr-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей массив из числа x, переданного ей в качестве аргумента, и ещё 9 подряд идущих за x чисел, т.е. [x, x + 1, …, x + 9]  Практические задания на отработку умения применять методы обработки массивов  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания arr-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей сумму элементов массива, переданного ей в качестве аргумента x |
| **7.2** | **Модуль 7. Тема 7.2** | Создание объектов  Работа со свойствами объектов | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения применять методы создания объектов  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания obj-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей JSON в виде правильной строки; у исходного объекта должно быть свойство love со строковым значением javascript и свойство year с числовым значением 2020 и свойство arg со значением x.  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания obj-002 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей массив собственных перечислимых ключей объекта, переданного ей в качестве аргумента x. |
| **7.3** | **Модуль 7. Тема 7.3** | Цепочка прототипов | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения работать с конструкторами объектов  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей имя конструктора объекта, переданного в аргумент x  Практические задания на отработку умения работать с прототипами объектов  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей число ненулевых прототипов у объекта o3, доступного по адресу  https://kodaktor.ru/j/protochain или https://kodaktor.ru/j/protochain2 либо npm-пакета goss\_proto |
| **8.1** | **Модуль 8. Тема 8.1** | Конструкторы объектов  webpack-dev-server | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения работать с конструкторами объектов  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания dom-000 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей имя тега элемента, переданного в аргумент x  Практические задания на отработку умения работать с девелоперским сервером webpack  *Пример задания*  Установите в локальной папке webpack версии 4.44.1, webpack-cli версии 3.3.12 и webpack-dev-server версии 3.11.0. Создайте в этой же папке файл index.html с любым содержимым, затем создайте подпапку src и в ней пустой файл index.js. Запустите webpack-dev-server и перейдите по адресу http://localhost:8080/error. Скопируйте первые два слова, которые выдаются на странице браузера левее слэша. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей в виде строки эти два слова, скопированных вами ранее. |
| **9.1** | **Модуль 9. Тема 9.1** | Веб-компоненты | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой*.  Практические задания на использование веб-компонентов  *Пример задания*  1. В этом задании вам нужно продемонстрировать знакомство с правилом именования веб-компонента в HTML5.  2. Перейдите по адресу  <https://node-server.online/r> и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – compo-001 (выбрать из выпадающего списка) – и пароль (nodetasker).  3. Введите код, выполняющий задание:  тело функции task, которая по аргументу x, который содержит некоторую состоящую из строчных латинских букв строку, возвращает результат применения метода createElement объекта document к строке x с правильно присоединённым к ней суффиксом el, так чтобы в результате получалось корректное имя кастомного HTML5-элемента.  Пример оформления решения:  function task(x) { /\* решение \*/ } |
| **9.2** | **Модуль 9. Тема 9.2** | Создание приложений на React в браузере  Создание приложений на Vue в браузере | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения работать с элементами React  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания react-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей созданный с помощью функции createElement объекта React (он будет доступен в функции и используется версия 16.13.1) новый элемент h1 без свойств и с единственным потомком в виде текста, заданного аргументом функции x.  Практические задания на отработку умения работать с элементами Vue  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания vue-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей созданный с помощью вызова инстанциатора конструктора Vue (он будет доступен в функции и используется версия 2.6.12) новый экземпляр Vue-приложения со значением свойства el, указанного в аргументе x. |
| **10.1** | **Модуль 10. Тема 10.1** | Создание сценария, считывающего данные из файловой системы | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения работать с файловой системой с помощью node.js  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания node-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей содержимое файла в виде utf8-строки, считанного по пути, указанном в аргументе x |
| **10.2** | **Модуль 10. Тема 10.2** | Создание простейшего веб-сервера | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения создавать простейшее серверное веб-приложение с помощью JavaScript  *Пример задания*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания http-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей экземпляр http-сервера, созданный с помощью встроенного Node.js-модуля http (он будет доступен в функции). Этот сервер не должен быть запущен / слушать какой-либо порт. Используйте только res.end (не нужны заголовки или теги HTML). Он должен обрабатывать два маршрута:  (1) /challenge – в ответ на такой запрос он должен возвращать x  (2) /api/rv/abc  где abc – произвольная строка длиной не менее 1 символа, состоящая только из строчных латинских букв.  В ответ на такой запрос сервер должен возвращать перевёрнутую строку.  Например: https://kodaktor.ru/api/rv/abc  Для других маршрутов предусмотрен ответ No |

**8.Оценочные материалы по образовательной программе**

**8.1. Вопросы тестирования по модулям**

|  |  |  |  |
| --- | --- | --- | --- |
| **№ модуля** | **Вопросы входного тестирования** | **Вопросы промежуточного тестирования** | **Вопросы итогового тестирования** |
| **0 (Входное тестирование)** | 1. К какому году относится стандарт JavaScript, который включил в себя оператор возведения в степень?   1. 2016 2. 2019 3. 2013 4. 1995   2. Какая кодировка используется для представления символов Юникода в языке JavaScript?&nbsp;   1. UTF-16 2. UTF-6 3. КОИ8-Р 4. Windows-1251   3. Назовите стандартное имя файла (с расширением) дескрипторов пакетов, в котором задаются зависимости для современного веб-приложения на JavaScript   1. package.json 2. package.js 3. manifest.txt 4. depend.json   4. Какое ключевое слово JavaScript (ECMAScript 2015) позволяет создать переменную, которой запрещено переприсваивать значение?   1. const 2. var 3. arr 4. dim   5. К какому объекту, предоставляемому браузером или платформой Node.js относится метод log, позволяющий выводить данные?   1. console 2. printer 3. terminal 4. logger   6. Что возвращает для значений null и undefined сравнение их между собой двойным равенством?   1. true 2. false 3. undefined 4. yes   7. Какой тип данных из перечисленных является в JavaScript примитивным?   1. Boolean 2. Object 3. Function 4. Array   8. Какое значение приобретает переменная, которая была объявлена, но которой не было присвоено значение?   1. undefined 2. null 3. not defined 4. unknown   9. Какой созданный в Facebook инструмент, конкурент языка TypeScript, работающий на основе Babel-транспиляции, позволяет добавлять в текст программы аннотации типов?   1. flow 2. typing 3. facepalm 4. npm   10. Какое слово позволяет досрочно завершить работу функции и вернуть значение в выражение вызова?   1. return 2. goback 3. print 4. call   11. Какая функция, доступная в браузере и на платформе Node.js, позволяет передать ей коллбэк и однократно отложить его выполнение на указанное количество миллисекунд?   1. setTimeout 2. setTimeoff 3. pause 4. delay   12. Какая инструкция реализует в JavaScript обыкновенный цикл со счётчиком?   1. for 2. count 3. var 4. iterate   13. Какое свойство строки позволяет получить количество символов в ней?   1. length 2. count 3. many 4. symbols   14. Какой символ обозначает начало и конец литерала регулярного выражения?   1. / 2. \* 3. ? 4. #   15. В массиве ['a', 'b', 'c'] каково значение элемента с индексом 1?   1. 'b' 2. 'a' 3. 'z' 4. 1   16. У объекта { name: 'Ivan' } что является ключом?   1. name 2. Ivan 3. name: Ivan 4. name, Ivan   17. Какое ключевое слово в JavaScript обозначает ссылку на текущий объект или контекст?   1. this 2. self 3. that 4. me   18. Как называется функция, вызываемая при создании нового объекта?   1. конструктор 2. деструктор 3. инициализатор 4. присваиватель   19. Как называется современный транспилятор, позволяющий использовать экспериментальные версии JavaScript и преобразовывать их в ES5?   1. Babel 2. Loader 3. Webpack 4. Flow   20. Какая функция используется для подключения зависимостей на платформе Node.js пока не реализованы модули ECMAScript2015?   1. require 2. demand 3. include 4. switch | **-** | **-** |
| **1** | **-** | 1. Кто является создателем языка JavaScript? Brendan Eich  Bill Gates Alan Turing Niklaus Wirth 2. В каком году был создан язык JavaScript? 1995 1990 1985 2000 3. Как называется стандартизованная версия языка JavaScript? ECMAScript Java Jscript ActionScript 4. Как называется текстовый формат обмена данными, основанный на JavaScript? JSON XML TOML MarkDown 5. Какой разработчик ассоциируется с созданием языка JSON? Douglas Crockford Ryan Dahl Evan You John Resig 6. Как называется версия стандарта JavaScript, следующая после ES5? ES2015 ES.More EcmaNext ECMA++ 7. Как называется движок JavaScript, на основе которого работают браузер Google Chrome и платформа Node.js? V8 G8 JSEngine2015 JavaScriptByGoogle 8. Какой оператор был введён в JavaScript в стандарте 2016 года? возведения в степень извлечения корня Набла сцепления строк 9. Что из перечисленного является транспилятором? Babel React ESLint Nodemon 10. Как называется компилятор в составе V8? Turbofan SpiderMonkey Elixir Spring Boot | **-** |
| **2.1** |  | 1. Что означает последняя цифра в версии пакета nodemon 2.0.4? -исправление ошибок -новая функциональность - изменение лицензии - переход на другой движок 2. Для символа ə как выглядит представление на языке JavaScript? - '\u0259' - '/u0259' - '\u\0259' - '\u0259” 3. Какая система кодирования используется в JavaScript? - Unicode - КОИ8-Р - EuroSystem-2015 - v8Encoding 4. Как верно записывается инструкция JavaScript для объявления переменной без присваивания? - let - LET - Let - const 5. Какой ключ при вызове npm или yarn для инициализации проекта используется для заполнения полей в package.json значениями по умолчанию? - y - defaults - values - package 6. Как называется линтер для современной версии ECMAScript? - eslint - jschecker - esvalidator - ecmalinter 7. Что в выражении 2 \* 3.3 является оператором? - \* - 2 - 3 - . 8. С помощью какого атрибута в элементе script указывается адрес сценария для подключения? -src - scr - location - address 9. Какой тип содержимого указывается при добавлении к веб-странцие сценария на языке JavaScript с помощью элемента script? - text/javascript - program/javascript - script/java - code/javascript 10. В выражении console.log('1); какой символ используется ошибочно и должен быть устранён для вывода в консоль числа? -' - . - ; - ( |  |
| **2.2** | **-** | 1. В папке с каким названием содержатся зависимости JavaScript-проекта на платформе Node.js? - node\_modules  - nodejs-modules - js\_dependencies - packages\_folder 2. Какое поле файла package.json предназначено для хранения информации о зависимостях, которые включаются в состав проекта? - dependencies - devDependencies - node\_modules - js\_packages 3. Какое поле файла package.json предназначено для хранения информации о зависимостях, которые используются только на этапе разработки ? - devDependencies - dev-modules - development\_packages - v8Encoding 4. Как называется менеджер зависимостей Node.js? - npm - composer - apt - yum 5. Какой ключ при вызове npm для инициализации проекта используется для установки пакета глобально? - g - common - system - j 6. Какая команда npm в сокращённой форме означает запуск установки пакета? - i - in - s - se 7. Какая команда npm предназначена для инициализации проекта и заполнения файла package.json значениями по умолчанию? - init - initialize - default - values 8. При инициализации с помощью команды init менеджера npm с указанием ключа y какое поле оказывается первым в файле package.json? - name - version - main - license | **-** |
| **3.1** | **-** | 1. Как корректно записывается инструкция создания переменной с обязательным присваиванием значения в JavaScript? - const  - let - var - val 2. Какой оператор означает присваивание значения имени переменной? - = - == - :=  - > 3. Какой символ начинает блок в сценарии на JavaScript? - { - [  - <  - ` 4. Как называется оператор удаления переменной в JavaScript? - delete - remove - kill - erase 5. Какая инструкция объявления переменной стала частью стандарта ES2015? - let - var - dim - private 6. Какой тип ошибки характеризует попытку получить в сценарии JavaScript значение переменной, которая не была объявлена? - ReferenceError - SyntaxError - TypeError - VarError 7. Какой оператор позволяет проверить факт того что переменная не связана ни с каким значением (кроме undefined), без риска возникновения ошибки ReferenceError и без влияния на эту переменную? - typeof - delete - checkexistence - wasdeclared 8. Какое значение окажется связанным с переменной z после выполнения инструкции-выражения var z = typeof z == 'undefined' - true - undefined - false - null | **-** |
| **3.2** | **-** | 1. Какое из нижеследующих значений примитивно? - null  - [1] - () => {} - { a: 2 } 2. Что возвращает для значений null и undefined проверка двойным равенством? - true - false - null  - undefined 3. Какой символ начинает массив в сценарии на JavaScript? - [ - { - <  - ` 4. Как называется оператор проверки типа переменной в JavaScript? - typeof - remove - kill - erase 5. Какой тип объектных данных стал частью стандарта ES2015? - Map - Function - Object - Array 6. Какой тип ошибки характеризует попытку вызвать в качестве функции то, что не является функцией? - TypeError  - SyntaxError - ReferenceError - VarError 7. Какой оператор позволяет проверить факт того что переменная ссылается на экземпляр того или иного типа объектов? - instanceof - typeof - checkclass - isoftype 8. Из какого количества символов состоит тернарный оператор? - 2 - 3 - 4 - 1 | **-** |
| **3.3** | **-** | 1. В каком году был предложен TypeScript? - 2012  - 1992 - 2018 - 1995 2. Какая компания начала и поддерживает разработку TypeScript? - Microsoft - Adobe - Yandex  - Google 3. Какая команда используется для запуска компилятора TypeScript? - tsc - types - tscript  - tscompiler 4. Какое расширение имеют файлы, содержащие сценарии на TypeScript? - .ts - .typesc - .jst - .bin 5. Наличие  какого файла в каталоге указывает на то, что каталог является корнем проекта TypeScript? - tsconfig.json - tspackage.json - typescript.yml - ts.js 6. Какой тип ошибки TypeScript характеризует некорректную попытку присвоить одному типу другой тип, например let n: number = 'a'? - TS2322  - ES2015 - TS2321 - ERRTYPE2333 7. Что определяется как именованный тип, задаваемый перечислением списка его членов и их типов? - интерфейс - класс - кортеж - ассоциативный массив 8. Что из перечисленного ниже задаёт тип-пару двух чисел (tuple)? - [number, number] - {number > number} - \number: number\ - [[number]: [number]] | **-** |
| **4.1** | **-** | 1. Что возвращает функция, заданная как () => {}? - undefined  - object - null - nothing 2. Какой символ открывает тело функции, задаваемой с помощью слова function? - { - [ - >  - ` 3. Какое слово обозначает контекст функции, задаваемой с помощью слова function? - this - self - me  - that 4. Какая последовательность символов задаёт зависимость результата от аргументов в стрелочной функции? - => - /> - --> - <= 5. Что возвращает оператор typeof для () => {}? - function - object  - anonymous - arrow 6. Какое ключевое слово используется для преждевременного завершения работы кода в теле функции? - return - goto  - end  - callback 7. Какая из функций возвращает объект? - () => ({}) - () => {} - () => {[]} - () => () 8. Что возвращает функция, в теле которой отсутствует инструкция return? -undefined - null - nothing - {} | **-** |
| **4.2** | **-** | 1. Назовите метод класса Function, позволяющий вызвать функцию, переопределив её this-контекст и передав её массив аргументов - apply  - invoke - return - run 2. В выражении setTimeout(func, ms, args) что является коллбэком? - func - setTimeout - ms - args 3. Среди перечисленных выражений что является примером IIFE (немедленно вызываемого функционального выражения)? -(() => {})() - (() => {})[] - (() => {}){} - /(() => {})/ 4. Что должна возвращать функция f, чтобы выражение f()() могло быть выполнено без ошибок?  - функцию - массив - число - множество 5. Какое свойство функции позволяет выяснить число аргументов, с которыми она была объявлена? - length - size  - args - number 6. Что возвращает свойство name для анонимной функции? - пустую строку - undefined  - null  - nothing 7. Что нужно передать в качестве аргумента x при вызове функции x => x(), чтобы она могла выполниться без ошибок? - функцию - массив - null - undefined 8. Как называется функция в составе объекта? - метод - интерфейс - оператор - дескриптор | **-** |
| **5** | **-** | 1. У какого из перечисленных выражений есть побочный эффект? - a = 3  - a == 3 - a \* 3 - !a 2. Какая из этих инструкций приведёт к появлению единицы в консоли? - if ({}) console.log(1) - if () console.log(1) - if {} console.log(1) - if ('') console.log(1) 3. Среди перечисленных выражений что является инструкцией вызова функции? - (() => {})() - (() => {})[] - (() => {}){} - /(() => {})/ 4. Какая инструкция проверяет свой аргумент на идентичность одному из нескольких перечисленных случаев-значений ?  - switch - for - when - with 5. Какая из инструкций создаёт бесконечный цикл ? - while({}); - while{}; - while(); - while; 6. Какая инструкция позволяет обернуть код проверкой на возникновение в нём отлавливаемых ошибок? - try…catch - try…return  - fetch…catch  - find…error 7. Какие ошибки не могут быть отловлены с помощью обёртывания кода проверкой на наличие отлавливаемых ошибок ? - SyntaxError - TypeError - ReferenceError - RangeError 8. Какая инструкция позволяет «бросить», т.е. сгенерировать исключительную ситуацию в сценарии? - throw - catch - push - drop | **-** |
| **6.1** | **-** | 1. Какое из нижеследующих значений примитивно? - 'abc'  - () => {} - [] - {} 2. Какой оператор осуществляет конкатенацию строк? - + - . - conc - $ 3. Какое свойство строк позволяет получать количество символов в них? - length - size - number - characters 4. Если один из двух аргументов операции + является строкой, к какому типу будет приведён второй аргумент?  - String - Object - Any - Number 5. Какой знак используется как префикс для выражений в фигурных скобках в составе шаблонных строк ES2015? - $ - # - & - @ 6. Что является результатом вызова для строки 'abc' метода substring с аргументами (1, 2)? - b - a  - ab  - bc 7. Как в шестнадцатеричном виде выглядит код начальной буквы слова «веб», полученный с помощью вызова метода charCodeAt? -432 - 234 - 123 - 321 8. Каков результат применения метода parseInt к строке «веб» ? - NaN - 3 - 0 - Infinity | **-** |
| **6.2** | **-** | 1. Какой символ используется для ограничения с обеих сторон литерала регулярного выражения ? - /  - $ - \* - \ 2. В выражении 'ababua'.replace(/a/, 'c') какой один символ следует вставить между слэшем и запятой, чтобы все вхождения буквы a были заменены на букву c? - g - i - o - m 3. В выражении 'ababua'.replace(/./, 'c') какой один символ следует поместить между точкой и слэшем, чтобы вся строка оказалась заменена на одну букву c? - \* - $ - # - @ 4. Если строке 'Saint-Petersburg' глобально сопоставить регулярное выражение /\w+/, какой будет длина получившегося массива?  - 2 - 22 - 222 - 0 5. Какому классу принадлежит метод match, который осуществляет сопоставление строки регулярному выражению? - String - Set - Array - Map 6. К какому типу с точки зрения оператора typeof относится /.\*/? - object - regularexpression  - string  - null 7. Какая строка соответствует регулярному выражению /^\d{1,2}$/ ? - '01' - '011' - 'ab' - 'a1' 8. Что из перечисленного является корректным комментарием ? - /\*\*/ - /.\*/ - /\* /.\*/ \*/ - /\.\*/ | **-** |
| **7.1** | **-** | 1. Что является результатом конкатенации строки и массива? - строка  - массив - объект - undefined 2. Что является результатом сравнения пустого массива и значения false с помощью оператора двойного равенства? - true - false - null - undefined 3. Каково значение свойства length выражения [[]]? - 1 - 0 - -1 - NaN 4. Какое из нижеследующих значений не примитивно? ?  - [] - 'a' - null - 0 5. В инструкции инициализации литерала массива квадратными скобками определите и выпишите символ, завершающий литерал массива   let r = ['first', 'second', 'third']; // массив r - ] - / - ; - r   1. Пусть массив m есть [3,2]; Сколько различных элементов в массиве [1,2, ...m, 1]? - 3 - 4  - 5  - 6 2. Какова сумма элементов массива, получаемого в результате вычисления выражения [1,2,3].map(x=>x\*x)? - 14 - 11 - 6 - NaN 3. Сколько элементов получится в массиве, который будет являться результатом вызова метода split без аргументов по отношению к строке 'abc' ? - 1 - 0 - 3 - NaN | **-** |
| **7.2** | **-** | 1. Что является результатом конкатенации строки и объекта? - строка  - массив - объект - undefined 2. Что является результатом сравнения с помощью оператора двойного равенства выражения !{} и значения false ? - true - false - null - undefined 3. Каково значение свойства length выражения {}? - undefined - 0 - -1 - NaN 4. Какое из нижеследующих значений не примитивно? ?  - {} - 'a' - undefined - 1 5. В инструкции инициализации литерала объекта фигурным скобками определите и выпишите символ, завершающий литерал объекта.   let o = {'first': 1, 'second': 2, 'third': 3}; // объект o - } - / - ; - o   1. Пусть объект o есть {a:5, a: 4, a:3, a:2}; Каким будет значение свойства a объекта o? - 2 - 3  - 4  - 5 2. В выражении {}+[] чем является часть до символа «плюс»? - блоком - объектом - массивом - строкой 3. Какой оператор используется для перебора свойств объекта ? - in - of - off - from | **-** |
| **7.3** | **-** | 1. Что является прототипом объекта, созданного инициализатором литерала объекта? - Object.prototype  - { Prototype } - null - undefined 2. Как создать объект без прототипа? - Object.create(null) - Object.create() - Object.create(undefined) - Object.create({}) 3. Назовите бинарный оператор метапрограммирования JavaScript, позволяющий проверить, что левый операнд является экземпляром второго операнда-как-класса - instanceof - classof - constructor - isclassof 4. Что является конструктором для () => {}?  - Function - Object - Map - Array 5. Что является конструктором для /.\*/ ?   - RegExp - Object - Array - null   1. Что является типом для /.\*/ ? - object - RegExp - Comment  - null 2. Что является типом для null?  - object - null - undefined - number 3. Какой метод позволяет узнать прототип объекта ? - Object.getPrototypeOf - Object.constructor - findPrototype - showProto | **-** |
| **8.1** | **-** | 1. Каким должно быть значение поля type в файле package.json, чтобы в проекте можно было использовать импорт и экспорт в рамках стандарта ECMAScript? - module - ecmascript - require - object 2. Какое ключевое слово ES2015 позволяет обозначить экспортируемую сущность как доступную для импорта по умолчанию? - default - imported - accessible - exported 3. Какой метод DOM позволяет получить первый из элементов, удовлетворяющих указанному CSS-селектору? - document.querySelector  - document.firstElement - window.selector - browser.getElementBySelector 4. К числу каких зависимостей проекта относится webpack ? - devDependencies - dependencies - runtimeDependencies - devModules 5. Какой пакет устанавливается вместе с webpack чтобы обеспечить управление сборкой проекта из командной строки?  - webpack-cli - webpack-command - webpack-api - webpack-manager 6. Для версий 7.8.x как называется пакет экосистемы транспилятора babel, содержащий функции для управления ядром из командной строки?  - @babel/cli - babel-core-cli - babel-commands - transpiler-cli-package 7. Какой ключ команды вызова транспилятора babel позволяет указать надстройки для преобразований, например, @babel/plugin-proposal-nullish-coalescing-operator ?   - plugins - addons - custom - rules   1. Как по умолчанию называется файл, создаваемый сборщиком webpack при соединении воедино файлов проекта ? - main.js - index.js - result.js  - bundlefile.js | **-** |
| **9** | **-** | 1. Какая компания инициировала разработку библиотеки React? - Facebook - Adobe - Yandex - Abbyy 2. Какое свойство React-компонентов позволяет обратиться к атрибутам тега, который создаёт экземпляр компонента? - props - attrs - values - keys 3. Какой язык шаблонизации, используемый в React-компонентах, позволяет записывать теги без кавычек внутри сценария на JavaScript? - JSX  - pug  - handlebars  - twig 4. К какой библиотеке экосистемы React относится метод render? - ReactDOM - ReactRenderer - runtimeReact - reactiveLib 5. Какая функция библиотеки React позволяет программно создать новый элемент, не прибегая к тегам JSX?  - createElement - createReactComponent - insertComponent - addCustomTag 6. Как называется разработанный в Facebook пакет, автоматизирующий создание современных приложений на React?  - create-react-app - react-facebook-creator - modern-react-composer - react-app-adder 7. Какая технология в React обеспечивает быструю работу с деревом элементов, позволяя вносить изменения по мере необходимости и рассчитывая их в оперативной памяти?   - Virtual DOM - Shadow DOM - DOM 3D - VRDOM   1. Какое свойство в описании экземпляра Vue-приложения отвечает за указание элемента веб-страницы, к которому подмонтируется экземпляр? - el - entry - mountPoint  - src | **-** |
| **10** | **-** | 1.Какой инструмент позволяет перезапускать приложения при внесении изменений в исходный код?  - nodemon  - eslint  - prettyjson  - moment  2. Выберите верное утверждение  - Веб-приложение на Node.js может включать в себя функциональность веб-сервера  - Веб-сервер – это приложение на Node.js  - Веб-сервер – это приложение, которое отвечает на HTTP-запросы, отправляя код ответа 200 на любые запросы  - Веб-приложение на Node.js не позволяет детально описать реакцию на HTTP-запрос и не позволяет определить, каким методом запрос был отправлен  3. Какой модуль экосистемы node.js обеспечивает базовую функциональность веб-сервера?  - http  - net  - fs  - util  4. Какое событие генерируется, когда прослушивающий HTTP-порт сервер на Node.js на базе встроенного модуля http получает запрос от клиента?  - request  - response  - query  - question  5. Если функции Server или createServer модуля http передаётся обработчик-коллбэк с двумя параметрами r1 и r2, в который из них при обработке запроса будет передан объект ответа?  - r2  - r1  - такой объект вообще не передаётся обработчику запроса  - ни в один из этих двух, нужен третий параметр  6. Какой метод объекта res, вызываемый до метода end, позволяет отправить код состояния и заголовки клиенту?  - res.writeHead  - res.send\_headers  - res.goAhead  - res.helmet  7. Какие типы данных или экземпляры каких классов можно отправлять с помощью метода res.write?  - строку  - буфер  - объект  - null  8. Экземпляром какого класса является объект request?  - IncomingMessage  - ReceivedEvent  - ClientSignal  - ReadingStream | **-** |
| **Итоговое тестирование** |  |  | Примеры вопросов (тест формируется из базы тестовых вопросов)  1. Выберите стандарт JavaScript, в который был введён оператор возведения в степень.   1. ES2016 2. ES5 3. ES1.1 4. LiveScript   2. Какие символы экранируют Unicode-последовательность в JavaScript?   1. \u 2. \s 3. /\* 4. ?!   3. В файле какого формата описываются зависимости для современного веб-приложения на JavaScript?   1. JSON 2. XML 3. DOCX 4. PY   4. Какое ключевое слово в современном стандарте JavaScript (ECMAScript 2015) позволяет создать переменную, которой разрешается переприсваивать значение?   1. let 2. const 3. do 4. dim   5. Выберите объект, доступный в браузере и на платформе Node.js, к которому относится метод log, позволяющий выводить данные?   1. shell 2. console 3. tty 4. module   6. Чему равно значение, возвращаемое оператором typeof для значения null?   1. 'object' 2. false 3. undefined 4. 0   7. Выберите значение, которое оператор typeof возвращает для значения NaN   1. 'number' 2. Object 3. NaN 4. 0   8. Если переменная b не объявлялась и не присваивалась (и вообще не упоминалась в программе до этого), какое значение вернёт выражение typeof b?   1. undefined 2. null 3. not defined 4. error   9. Какой командой инициализируется проект на TypeScript?   1. tsc --init 2. yarn --init 3. npm --init 4. types -install   10. Какая аббревиатура обозначает совмещение выражения объявления функции и её вызова?   1. IIFE 2. JSON 3. RTFM 4. ECMA   11. В инструкции **const** f = () => console.log('done'); setTimeout(f, 5000); что является коллбэком?   1. setTimeout 2. f 3. console.log 4. ничего   12. Какая инструкция реализует в JavaScript цикл с предусловием?   1. while 2. forEach 3. forWhile 4. iterate   13. Какой метод строк позволяет дополнить строку слева нужным количеством повторений указанного символа?   1. padStart 2. padLeft 3. repeatLeft 4. addLeft   14. В выражении 'ababua'.replace(/a/, 'c') какой один символ следует вставить между слэшем и запятой, чтобы все вхождения буквы a были заменены на букву c?   1. g 2. ^ 3. ! 4. @   15. Какова сумма элементов массива, получаемого в результате вычисления выражения [1,2,3].map(x=> x \* x)   1. 14 2. 22 3. 0 4. infinity   16. Что является прототипом объекта, созданного инициализатором литерала объекта?   1. Object.prototype 2. undefined 3. null 4. constructor   17. Какое ключевое слово в JavaScript обозначает ссылку на текущий объект или контекст?   1. this 2. self 3. that 4. me   18. Какое понятие определяется как множество объектов, наследующих одному и тому же прототипу?   1. конструктор 2. класс 3. массив 4. интерфейс   19. Как называется npm-пакет, который содержит ядро транспилятора Babel?   1. @babel/core 2. @babel/main 3. @babel/cli 4. babel-nuclear   20. Какое ключевое слово ES2015 позволяет обозначить экспортируемую сущность как доступную для импорта по умолчанию?   1. default 2. module 3. exported 4. required   21. Выберите верное утверждение  1. Веб-приложение на Node.js может включать в себя функциональность веб-сервера ☑️  2. Веб-сервер – это приложение на Node.js  3. Веб-сервер – это приложение, которое отвечает на HTTP-запросы, отправляя код ответа 200 на любые запросы  4. Веб-приложение на Node.js не позволяет детально описать реакцию на HTTP-запрос и не позволяет определить, каким методом запрос был отправлен  22. Если функции Server или createServer модуля http передаётся обработчик-коллбэк с двумя параметрами r1 и r2, в который из них при обработке запроса будет передан объект ответа?   1. r2 2. r1 3. такой объект вообще не передаётся обработчику запроса 4. ни в один из этих двух, нужен третий параметр |

* 1. **описание показателей и критериев оценивания, шкалы оценивания.**

Шкалы оценивания

* 1. Входное тестирование

**Шкала критериев оценивания по входному тестированию**

|  |  |  |
| --- | --- | --- |
| **Шкала оценивания** | **Критерии** | **Описание** |
| 70%-100% верных ответов на вопросы входного теста | **зачтено** | Потенциальный обучающийся успешно выполнил входной тест и может быть допущен к обучению по программе |
| 69% и менее верных ответов на вопросы входного теста | **не зачтено** | Потенциальный обучающийся не выполнил успешно входной тест и не может быть допущен к обучению по программе |

* 1. Промежуточное тестирование по темам/модулям

**Шкала критериев оценивания по промежуточному тестированию**

|  |  |  |  |
| --- | --- | --- | --- |
| **Шкала** | **Критерии** | **Описание** | **Уровень сформированности компетенции** |
| 80%-100% верных ответов на вопросы промежуточного теста | зачтено | Обучающийся успешно выполнил промежуточный тест по теме/модулю | Продвинутый уровень |
| 70%-79% | зачтено | Обучающийся успешно выполнил промежуточный тест по теме/модулю | Базовый уровень |
| 50%-69% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно промежуточный тест, необходимо повторное назначение теста и консультации преподавателя | Начальный уровень |
| Менее 50% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно промежуточный тест, необходимо повторное назначение теста и консультации преподавателя | Не владеет, компетенция не сформирована |

* 1. Итоговое тестирование по программе

**Шкала критериев оценивания по итоговому тестированию**

|  |  |  |  |
| --- | --- | --- | --- |
| **Шкала** | **Критерии** | **Описание** | **Уровень сформированности компетенции** |
| 80%-100% верных ответов на вопросы промежуточного теста | зачтено | Обучающийся успешно выполнил итоговый тест по программе | Продвинутый уровень |
| 70%-79% | зачтено | Обучающийся успешно выполнил итоговый тест по программе | Базовый уровень |
| 50%-69% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно итоговый тест по программе | Начальный уровень |
| Менее 50% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно итоговый тест по программе | Не владеет, компетенция не сформирована |

* 1. **примеры контрольных заданий по модулям или всей образовательной программе.**

**Пример контрольного задания по программе**

1. В этом задании вам нужно продемонстрировать умение запустить простейший веб-сервер на сервисе Heroku на основе GitHub-репозитория.

2. Перейдите по адресу  <https://node-server.online/r> и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – final-001\_ (выбрать из выпадающего списка) – и пароль (nodetasker).

3. Введите в нижнем поле

начинающийся с https адрес (URL) приложения (на Heroku), которое:

(а) запущено на Node.js версии 10.19.0

(б) по маршруту /day выдаёт номер текущего дня (относительно текущего месяца) – число от 1 до 31

(в) по маршруту /v8 выдаёт версию движка v8 этой версии Node.js

(г) по маршруту /node выдаёт версию этой версии Node.js

(д) по маршруту /mirror?x=XYZ выдаёт XYZ

например по маршруту /mirror?x=Ilya выдаёт Ilya

(е) по маршруту /package.json выдаёт соответствующий файл

(ё) в остальных случаях возвращает имя автора решения

(ж) выдаёт заголовки, разрешающие кроссдоменный доступ в ответ на любой запрос

Пример оформления решения:

/\* https://final--01.herokuapp.com/ \*/

* 1. **тесты и обучающие задачи (кейсы), иные практико-ориентированные формы заданий.**

Примеры тестов представлены в таблице 8.1.

В программе используются преимущественно задания с автоматизированной проверкой, которые являются обязательными для выполнения по каждой теме. Дополнительно для организации самостоятельной работы используются задания со свободным ответом (в том числе проблемные), которые слушатели выполняют самостоятельно.

**ПРИМЕРЫ ЗАДАНИЙ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ**

**Модуль 1. Тема 1**

Запишите выражение по дереву (без пробелов; значок ^ здесь обозначает возведение в степень):

**Модуль 2. Тема 2.1**

Задание 1

Используя справку от node по объекту console и его методу log (https://nodejs.org/api/console.html#console\_console\_log\_data\_args), преобразуйте введённую инструкцию так, чтобы слово world оказалось в кавычках после запятой, а внутри первой пары кавычек был подстановочный символ («гнездо» для строки).

Используя справку по командной строке вашей операционной системы, найдите способ передать в запускаемый файл временное значение какой-нибудь переменной окружения.

Задание 2.

Найдите способ передать в запускаемый файл параметр(ы) командной строки.

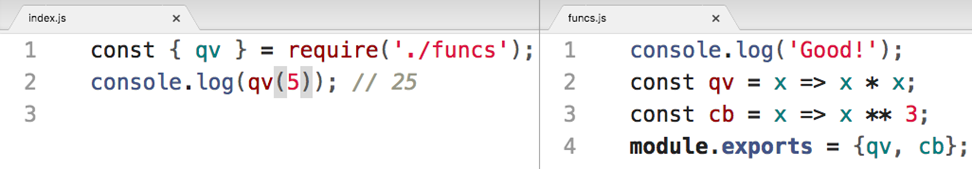
Задание 3.

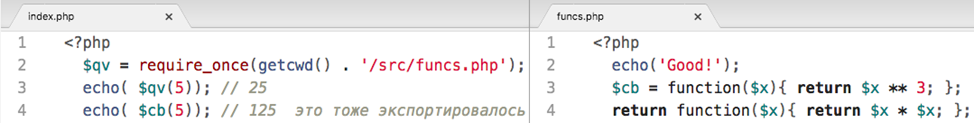
Создайте исполнимый файл на JavaScript, который можно запускать по его имени из любого места файловой системы, как если бы это была встроенная команда, который выводит в консоль текущую дату

**Модуль 2. Тема 2.2**

Задание 1.

Сравните работу require с require и require\_once языка PHP





Задание 2

Установите зависимость jsdoc, оформите любой пример (func-jsdoc.js) по соглашениям jsdoc и сгенерируйте для него документацию.

/\*\* @const {string} \*/ const hiWord = 'Hello';

/\*\*

\* возвращает приветствие

\* принимает строку или ничто, в случае ничто возвращает безымянное приветствие

\*/

function sayHello(name = 'Nameless') {

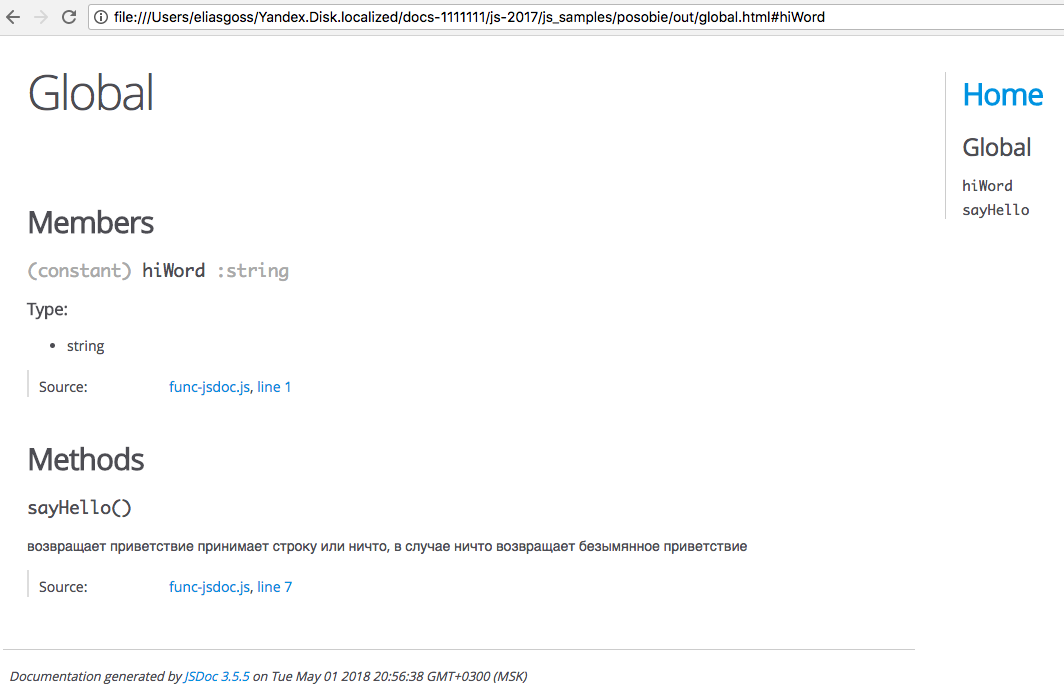
  return `${hiWord}, ${name}!`;

}

console.log(sayHello());

console.log(sayHello());

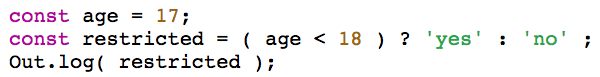
Выполнение npx jsdoc func-jsdoc.js приведёт к созданию папки out с некоторым сайтом, включающим страницу:



**Модуль 3.**

Задание 1

Рассмотрите код в строках 11-13 на странице https://kodaktor.ru/ternary



И доработайте его так, чтобы переменная restricted принимала не одно из двух, а одно из трёх различных значений: (а) значение yes при значении переменной age меньше 18 (б) значение notsure при значении переменной age равном 18 (в) значение no в противном случае

Указание. Это нужно сделать вкладыванием одного тернарного оператора в другой.

Задание 2

Исполнитель [алгоритмов] называется полным по Тьюрингу, если с его помощью можно реализовать любую вычислимую функцию, т.е. если он совпадает «по диапазону» задач с машиной Тьюринга (и прочими формализмами)

Строго говоря, ни один вычислитель не будет полным без бесконечного объема памяти, однако этим требованием можно пренебречь, если есть средства для обращения к памяти, не ограниченные ее объемом – например, «бесконечное» наращивание доступной памяти по мере требования.

Рассмотрим «шесть волшебных символов»: Шесть волшебных символов: []()+!

(пара квадратных скобок, пара круглых скобок, плюс и восклицательный знак).

Можно ли создать полный по Тьюрингу язык, используя только эти символы?

Пусть нам известно, что:

* ![] есть false
* !false есть true
* +true есть число 1
* x+y есть сумма чисел x и y
* пробел может иметь значение

Теперь запишите, пожалуйста, число 2 с помощью символов [ ] + !

Задание 3

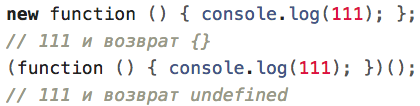
Определите, как с помощью объекта process и его свойств argv и env получать доступ к переменным окружения и параметрам вызова командной строки. Чем являются переданные значения с точки зрения типа данных?

**Модуль 4.**

**Тема 4.1**

Задание 1

Объясните поведение кода:



Задание 2

Вызовите функцию без использования круглых скобок.

Задание 3

Создайте функцию, которая возвращает трёхкомпонентный цвет CSS (например, 'rgb(123, 32, 12)' по трём переданным ей числам; по умолчанию для каждого компонента генерируется случайный цвет от 0 до 255.

**Тема 4.2.**

Спроектируйте способ создания IIFE, при котором выражение вызова писалось бы не справа от выражения функции, а слева

**Модуль 5.**

Задание 1

Так как в JavaScript существуют значения, которые нестрого равны друг другу при неявном приведении типов к логическому (они приводятся к false и называются falsy, «ложностные»), а одно из этих значений ещё и не равно самому себе, то нужен способ отличать их друг от друга.

Функция isNaN тоже занимается неявным приведением. Так, значение true возвращается вызовами isNaN() и isNaN('e').

(при этом отметим, что Math.sqrt(-1) не приводится к NaN, а в точности есть NaN, так же как литерал значения NaN, выглядящий в программе как NaN).

С использованием *операторов* напишите тернарный оператор, возвращающий:

'=NaN', если тестируемое значение в точности есть NaN,

'=null, если если тестируемое значение в точности есть null,

'=undefined, аналогично,

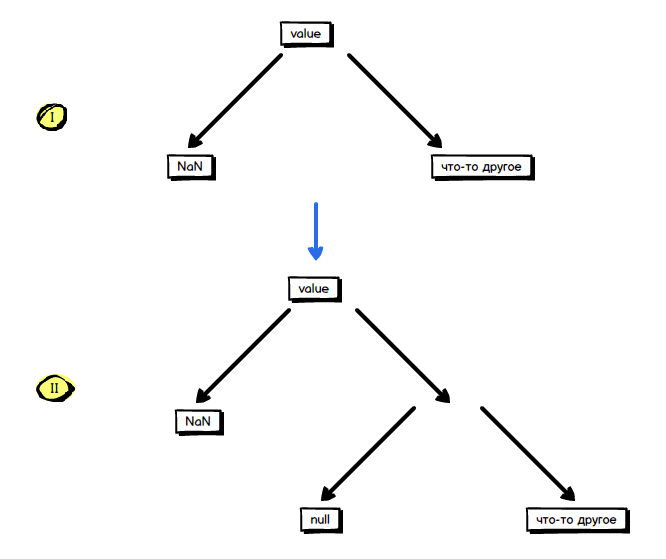
'=0', аналогично.

'=""', в случае пустой строки

и

'=false' в случае значения Boolean False.

Для этого поэтапно спроектируйте дерево вида:



Примечание.

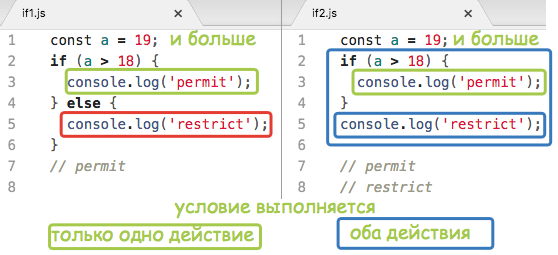
Как и оператор присваивания (см. выше) тернарный условный оператор в JavaScript правоассоциативен. Т.е. выражение  a = b = c эквивалентно a = (b = c), а не  (a = b) = c.

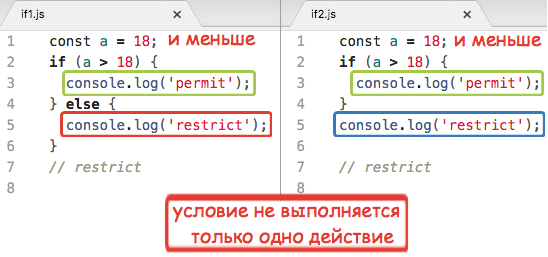
В то же время в таких языках как PHP можно встретиться с примером левоассоциативности, и именно в случае тернарного оператора. Чтобы добиться такой же работы условного оператора, как в JavaScript, в PHP необходимо вно расставлять порядок выполнения скобками.

Императивным аналогом тернарного оператора является традиционная инструкция ветвления if, а её специфическим частным случаем является инструкция выбора одного из вариантов switch.

Императивная инструкция не возвращает значения сама по себе, она только переключает поток выполнения в ту или иную ветку.

В зависимости от постановки задачи нам может оказаться нужной только одна ветка или обе. Если нам важно, чтобы какие-то действия выполнялись только при выполнении условия (либо только при его невыполнении, равнозначном выполнению отрицания этого условия), и не важно, что будет происходить в противоположной ситуации, мы можем обойтись только одной ветвью, т.е. сокращённой формой инструкции. Тогда (только) при выполнении условия произойдёт указанное действия и то, что следует дальше.





Варианты структурирования ветвления

**Модуль 6.**

Задание 1

С помощью цепочки методов toString и toUpperCase переведите десятичное число десять в шестнадцатеричную систему счисления

Задание 2

Решите обратную задачу с помощью метода parseInt и с помощью литерала шестнадцатеричного числа

**Модуль 7.**

Задание 1

Отсортируйте одномерный числовой массив по возрастанию числовых значений, используя метод sort и лямбду-предикат.

Задание 2

Передайте в функцию

const summer = (a, b, c) => a + b + c;

массив, содержащий три числа для подстановки вместо a, b, c, используя метод apply.

Задание 3

Найдите максимальный элемент в массиве.

Задание 4

Как можно убедиться в том, что свойство или метод находится в прототипе объекта?

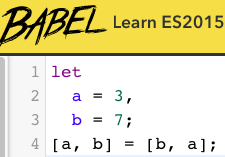
**Модуль 8.**

Задание 1

Создайте модуль с классом для системы npm и задокументируйте его.

Задание 2

Используя babel-preset-env, транспилируйте следующий сценарий:



**Модуль 9.**

Задание 1

Вот приложение на React, которое реализует простейший список задач с добавлением: [kodaktor.ru/react\_todo\_forvue](https://kodaktor.ru/react_todo_forvue). Оно загружает список из kodaktor.ru/j/tasklist список задач и создаёт массив элементов списка, используя метод fetch внутри асинхронного метода componentDidMount.

Перепишите его на Vue. Для элемента списка создаёте отдельный небольшой компонент. Для загрузки списка задач используйте метод created.

Задание 2

Перепишите на Vue приложение [kodaktor.ru/jsmixer\_src](https://kodaktor.ru/jsmixer_src), сделав бегунки (фейдеры), соответствующие трём цветам, компонентами. Разверните приложение на Heroku.

**Модуль 10.**

Задание 1

Создайте на Heroku простейшее серверное приложение на JavaScript, которое по корневому маршруту принимает в качестве query-параметра n число от 1 до 65535 в десятичной системе счисления, а возвращает в двоичной в виде набора из 8 цифр. Аналогичный пример для шестнадцатеричной системы: <https://kodaktor.ru/api/hex?n=45>

Задание 2

Создайте на Heroku простейшее серверное приложение на JavaScript, которое по маршруту /isapali принимает в качестве query-параметра s строку и возвращает 1, если она является палиндромом и 0 в противном случае.

**8.5.**  **описание процедуры оценивания результатов обучения.**

В процессе обучения слушатели должны выполнить практические задания с автоматизированной проверкой (в рамках практических занятий), задания самостоятельной работы и пройти промежуточную аттестацию по каждой теме/модулю – электронное тестирование. Промежуточное тестирование включает 8-10 вопросов по теме/модулю. Промежуточное тестирование считается успешно завершённым, если слушателем даны верные ответы на не менее чем 70% вопросов теста. Каждый слушатель имеет три попытки на выполнение промежуточного теста, время тестирования – 0,5 часа.

К итоговой аттестации допускаются слушатели, **выполнившие все практические задания с автоматизированной проверкой (в рамках практических занятий) и промежуточные тесты по темам/модулям (не менее 70% верных ответов по каждому тесту по теме)**, предусмотренные учебной программной. В ходе итогового электронного тестирования слушателю предлагается ответить на 20 вопросов. Тестирование считается успешно завершённым, если слушателем даны верные ответы на не менее чем 70% вопросов теста. Каждый слушатель имеет одну попытку на выполнение итогового теста, время тестирования – 1 час.

**9.Организационно-педагогические условия реализации программы**

**9.1. Кадровое обеспечение программы**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№**  **п/п** | **Фамилия, имя, отчество (при наличии)** | **Место основной работы и должность, ученая степень и ученое звание (при наличии)** | **Ссылки на веб-страницы с портфолио (при наличии)** | **Фото в формате jpeg** | **Отметка о полученном согласии на обработку персональных данных** |
| **1** | **Государев Илья Борисович** | Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО». Доцент факультета программной инженерии и компьютерной техники  Кандидат педагогических наук, доцент | https://kodaktor.ru/g/gosudarev |  | **+** |

**9.2.Учебно-методическое обеспечение и информационное сопровождение**

|  |  |
| --- | --- |
| **Учебно-методические материалы** | |
| Методы, формы и технологии | Методические разработки,  материалы курса, учебная литература |
| Методы: проблемно-развивающие методы; исследовательский, объяснительно-иллюстративный, метод проблемного изложения, частично-поисковый, методы контроля и самоконтроля  Формы: самостоятельная работа по освоению теоретического материала; самоконтроль через ответы на вопросы по видеолекциям и выполнение практических заданий для самоконтроля; промежуточный контроль (тесты по темам для промежуточной аттестации), итоговый контроль (итоговое тестирование по курсу). | 1. Государев, И.Б. Введение в веб-разработку на языке JavaScript [Электронный ресурс]: учебное пособие / И.Б. Государев. — Электрон. дан. — Санкт-Петербург: Лань, 2019. — 144 с. — Режим доступа: <https://e.lanbook.com/book/118648> 2. Современный открытый учебник [Электронный ресурс ]. Режим доступа: <http://javascript.ru/> |

|  |  |
| --- | --- |
| **Информационное сопровождение** | |
| Электронные  образовательные ресурсы | Электронные  информационные ресурсы |
| [Онлайн-редактор h1 («Кодактор») — авторская разработка; свидетельство о регистрации в Роспатенте (Федеральной службе по интеллектуальной собственности) №2014613612 от 01.04.2014)](https://kodaktor.ru/images/svid1.png)  Авторская система автоматизированной проверки заданий на языке JavaScript  **node-server.online/r** | 1. Выразительный JavaScript https://karmazzin.gitbooks.io/eloquentjavascript\_ru/content/index.html  2. Блог на коллективном ресурсе разработчиков <https://habr.com/ru/hub/javascript/> |

**9.3.Материально-технические условия реализации программы**

|  |  |
| --- | --- |
| Вид занятий | Наименование оборудования,  программного обеспечения |
| Лекционные (видеолекции) | Операционная система семейств Linux Mac Os, Windows.  Браузер Google Chrome  Открытая система управления виртуальными машинами VirtualBox  Рекомендуется скорость канала от 2 Мбит в секунду |
| Практические | Операционная система семейств Linux Mac Os.  Браузер Google Chrome, Canary  Открытая система управления виртуальными машинами VirtualBox  Рекомендуется скорость канала от 1 Мбит в секунду |
| Самостоятельная работа | Операционная система семейств Linux Mac Os.  Браузер Google Chrome, Canary  Открытая система управления виртуальными машинами VirtualBox  Рекомендуется скорость канала от 1 Мбит в секунду |

**III.Паспорт компетенций (Приложение 2)**

Автономная некоммерческая организация дополнительного профессионального образования

«СофтЛайн Эдюкейшн»

.

ПАСПОРТ КОМПЕТЕНЦИИ

**Дополнительная профессиональная программа повышения квалификации**

**«Введение в веб-разработку на языке JavaScript»**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | Наименование компетенции | Способен ставить и решать прикладные задачи по созданию и совершенствованию программного кода с использованием специализированных программных средств на примере инструментов экосистемы JavaScript | |
| 2. | Указание типа компетенции | профессиональная | |
| 3. | Определение, содержание и основные сущностные характеристики компетенции | Под компетенцией понимается способность постановки и нахождения путей решения прикладных задач по созданию и совершенствованию программного кода с использованием специализированных программных средств на примере инструментов экосистемы JavaScript  Слушатель должен:  *знать*:  - синтаксис языка программирования JavaScript и особенности программирования на JavaScript;  - особенности выбранной среды программирования;  - стандартные библиотеки языка программирования JavaScript  - технологии программирования на языке JavaScript  - место JavaScript среди современных интерпретируемых языков и сценарных языков программирования;  *уметь***:**  - применять JavaScript для написания программного кода;  - использовать выбранную среду программирования (редактор кода) для создания сценариев;  - использовать возможности имеющейся программной архитектуры веб-приложения;  *владеть*:  - навыками создания программного кода на JavaScript в соответствии с спецификациями;  - методами совершенствования программного кода с использованием специализированных программных средств на примере линтеров JavaScript;  - способами определения и манипулирования данными на JavaScript и JSON;  - навыками размещения программного кода в страницах, созданных при верстке информационных ресурсов (в клиентской части веб-ресурса) | |
| 4. | Дескриптор знаний, умений и навыков по уровням | Уровни сформированности компетенции  обучающегося | Индикаторы |
|  | Начальный уровень | *Знает*: синтаксис JavaScript и отдельные стандартные библиотеки для JavaScript.  *Умеет*: применять JavaScript для написания программного кода при решении простейших линейных алгоритмических задач.  *Владеет*: навыками разработки и запуска простейших сценариев на JavaScript в браузере |
|  | Базовый уровень | *Знает*: особенности JavaScript и выбранной среды программирования; место JavaScript среди современных интерпретируемых языков и сценарных языков программирования и стандартные библиотеки для JavaScript.  *Умеет*: применять JavaScript для написания программного кода при решении алгоритмических задач ветвления и повторения; использовать выбранную среду программирования (редактор кода).  *Владеет*: навыками разработки и запуска сценариев на JavaScript в браузере, реализует алгоритмические структуры ветвления и повторения на JavaScript |
|  | Продвинутый | *Знает*: технологии программирования на JavaScript.  *Умеет*: применять JavaScript для написания программного кода при решении прикладных задач с использованием адекватных алгоритмических структур.  *Владеет*: методами совершенствования программного кода с использованием специализированных программных средств на примере линтеров JavaScript; способами определения и манипулирования данными на JavaScript и JSON |
| 5. | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | Для формирования данной компетенции слушатель должен владеть:   * компетенциями цифровой грамотности, в том числе навыками владения ПК на уровне среднего или продвинутого пользователя; * компетенции в области алгоритмизации (знание основных алгоритмических структур, умения выражать их на псевдокоде и в виде блок-схем) и основ программирования | |
| 6. | Средства и технологии оценки | Тесты, электронное промежуточное (по темам/модулям) и итоговое тестирование | |

**IV.Иная информация о качестве и востребованности образовательной программы** (результаты профессионально-общественной аккредитации образовательной программы, включение в системы рейтингования, призовые места по результатам проведения конкурсов образовательных программ и др.) (при наличии)

**V.Рекомендаций к программе от работодателей**: наличие не менее двух писем и/или подтверждения на цифровой платформе Государственной системы предоставления ПЦС от работодателей о рекомендации образовательной программы для реализации в рамках Государственной системы предоставления ПЦС на формирование у трудоспособного населения компетенций цифровой экономики с указанием востребованности результатов освоения программы в сфере деятельности соответствующих компаний и готовности к рассмотрению заявок наиболее успешно освоивших образовательную программу граждан на прохождение стажировки и (или) собеседования на предмет трудоустройства путем проставления отметки в профиле программы

**Загружено на платформу**

**VI.Указание на возможные сценарии профессиональной траектории граждан** по итогам освоения образовательной программы (в соответствии с приложением)

**СЦЕНАРИИ ПРОФЕССИОНАЛЬНОЙ ТРАЕКТОРИИ ГРАЖДАН**

|  |  |
| --- | --- |
| **Цели получения персонального цифрового сертификата** | |
| **текущий статус** | **цель** |
| **Развитие компетенций в текущей сфере занятости** | |
| работающий по найму в организации, на предприятии | сохранение текущего рабочего места |
| работающий по найму в организации, на предприятии | развитие профессиональных качеств |
| работающий по найму в организации, на предприятии | повышение заработной платы |
| работающий по найму в организации, на предприятии | смена работы без изменения сферы профессиональной деятельности |

**VII.Дополнительная информация**

**VIII.Приложенные Скан-копии**

Загружено на платформу:

1. Два рекомендательных письма

2. Скан утвержденной образовательной программы

3. Скан утвержденного паспорта образовательной программы

4. Паспорт образовательной программы в формате word

**Генеральный директор**

**АНО ДПО «СофтЛайн Эдюкейшн» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Э. Разуваев**

М.П.