Частное учреждение образования

«Колледж бизнеса и права»

|  |
| --- |
| УТВЕРЖДАЮ  Заведующий  методическим кабинетом  \_\_\_\_\_\_\_\_\_\_Е.В.Паскал  «\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ |

|  |  |
| --- | --- |
| Специальность:  2-40 01 01 «Программное обеспечение информационных технологий» | Дисциплина: «Скриптовые языки программирования» |

**Лабораторная работа № 6**

**Инструкционно-технологическая карта**

Тема: Условное ветвление. Тернарный оператор. Логические операторы.

Цель: научится разрабатывать алгоритмы с использованием условного ветвления, научится использовать тернарный оператор, использовать логические операторы.

Время выполнения: 2 часа

# Порядок выполнения работы

1. Изучить теоретические сведения к лабораторной работе.
2. Реализовать алгоритм решения задачи.
3. Отлаженную, работающую программу сдать преподавателю. Работу программы показать с помощью самостоятельно разработанных тестов.
4. Ответить на контрольные вопросы.

# Теоретические сведения

Иногда нужно выполнить различные действия в зависимости от условий.

Для этого нужно использовать инструкцию **if** и условный оператор **?**, который также называют оператором «вопросительный знак».

#### Инструкция «if»

Инструкция **if(...)** вычисляет условие в скобках и, если результат **true**, то выполняет блок кода.

Например:

let year = prompt('В каком году была опубликована спецификация ECMAScript-2015?', '');

if (year == 2015) alert( 'Вы правы!' );

В примере выше, условие – это простая проверка на равенство (year == 2015), но оно может быть и гораздо более сложным.

Если выполнить более одной инструкции, то нужно заключить блок кода в фигурные скобки:

if (year == 2015) {

alert( "Правильно!" );

alert( "Вы такой умный!" );

}

Рекомендуется использовать фигурные скобки {} всегда, когда используются инструкцию **if**, даже если выполняется только одна команда. Это улучшает читаемость кода.

#### Преобразование к логическому типу

Инструкция **if (…)** вычисляет выражение в скобках и преобразует результат к логическому типу.

Правила преобразование типов:

Число **0**, пустая строка "", **null**, **undefined** и **NaN** становятся **false**. Из-за этого их называют «ложными» («falsy») значениями.

Остальные значения становятся **true**, поэтому их называют «правдивыми» («truthy»).

Таким образом, код при таком условии никогда не выполнится:

if (0) { // 0 is falsy

...

}

…а при таком – выполнится всегда:

if (1) { // 1 is truthy

...

}

Также можно передать заранее вычисленное в переменной логическое значение в **if**, например так:

let condition = (year == 2015); // преобразуется к true или false

if (condition) {

...

}

#### Блок «else»

Инструкция **if** может содержать необязательный блок **«else»** («иначе»). Он выполняется, когда условие ложно.

Например:

let year = prompt('В каком году была опубликована спецификация ECMAScript-2015?', '');

if (year == 2015) {

alert( 'Да вы знаток!' );

} else {

alert( 'А вот и неправильно!' ); // любое значение, кроме 2015

}

#### Несколько условий: «else if»

Иногда, нужно проверить несколько вариантов условия. Для этого используется блок **else if**.

Например:

let year = prompt('В каком году была опубликована спецификация ECMAScript-2015?', '');

if (year < 2015) {

alert( 'Это слишком рано...' );

} else if (year > 2015) {

alert( 'Это поздновато' );

} else {

alert( 'Верно!' );

}

В приведённом выше коде JavaScript сначала проверит year < 2015. Если это неверно, он переходит к следующему условию year > 2015. Если оно тоже ложно, тогда сработает последний **alert**.

Блоков **else if** может быть и больше. Присутствие блока **else** не является обязательным.

#### Условный оператор «?»

Иногда нужно определить переменную в зависимости от условия.

Например:

let accessAllowed;

let age = prompt('Сколько вам лет?', '');

if (age > 18) {

accessAllowed = true;

} else {

accessAllowed = false;

}

alert(accessAllowed);

Так называемый «условный» тернарный оператор позволяет сделать это более коротким и простым способом.

Оператор представлен знаком вопроса **?**. Его также называют «тернарный», так как этот оператор, единственный в своём роде, имеет три аргумента.

Синтаксис:

let result = условие ? значение1 : значение2;

Сначала вычисляется условие: если оно истинно, тогда возвращается значение1, в противном случае – значение2.

Например:

let accessAllowed = (age > 18) ? true : false;

Можно опустить круглые скобки вокруг age > 18. Оператор вопросительного знака имеет низкий приоритет, поэтому он выполняется после сравнения >.

Этот пример будет делать то же самое, что и предыдущий:

// оператор сравнения "age > 18" выполняется первым в любом случае

// (нет необходимости заключать его в скобки)

let accessAllowed = age > 18 ? true : false;

Но скобки делают код более простым для восприятия, поэтому рекомендуется их использовать.

#### Несколько операторов «?»

Последовательность операторов вопросительного знака **?** позволяет вернуть значение, которое зависит от более чем одного условия.

Например:

let age = prompt('Возраст?', 18);

let message = (age < 3) ? 'Здравствуй, малыш!' :

(age < 18) ? 'Привет!' :

(age < 100) ? 'Здравствуйте!' :

'Какой необычный возраст!';

alert( message );

Поначалу может быть сложно понять, что происходит. Но при ближайшем рассмотрении видно, что это обычная последовательная проверка:

Первый знак вопроса проверяет age < 3.

Если верно – возвращает 'Здравствуй, малыш!'. В противном случае, проверяет выражение после двоеточия «:», вычисляет age < 18.

Если это верно – возвращает 'Привет!'. В противном случае, проверяет выражение после следующего двоеточия „:“, вычисляет age < 100.

Если это верно – возвращает 'Здравствуйте!'. В противном случае, возвращает выражение после последнего двоеточия – 'Какой необычный возраст!'.

Вот как это выглядит при использовании **if..else**:

if (age < 3) {

message = 'Здравствуй, малыш!';

} else if (age < 18) {

message = 'Привет!';

} else if (age < 100) {

message = 'Здравствуйте!';

} else {

message = 'Какой необычный возраст!';

}

# Контрольные вопросы

1. Перечислите значения, которые при преобразовании к логическому типу данных принимают значение true и false.
2. Опишите работу тернарного оператора.
3. К какому типу данных приводится значение внутри оператора if?
4. Перечислите логические опреаторы.

# Задания

**Задание 1**. Название JavaScript.

Используя конструкцию **if..else**, написать код, который будет спрашивать: Какое «официальное» название JavaScript?

Если пользователь вводит «ECMAScript», то показать: «Верно!», в противном случае – отобразить: «Не знаете? ECMAScript!»

**Задание 2**. Покажите знак числа

Использовать конструкцию **if..else**, написать код, который получает число через **prompt**, а затем выводит в **alert**: 1, если значение больше нуля, -1, если значение меньше нуля, 0, если значение равно нулю.

Предполагается, что пользователь вводит только числа.

**Задание 3**. Переписать конструкцию **if** с использованием условного оператора «**?**»:

let result;

if (a + b < 4) {

    result = 'Мало';

} else {

    result = 'Много';

}

**Задание 4**. Переписать **if..else** с использованием нескольких операторов «?».

let message;

if (login == 'Сотрудник') {

    message = 'Привет';

} else if (login == 'Директор') {

    message = 'Здравствуйте';

} else if (login == '') {

    message = 'Нет логина';

} else {

    message = '';

}

# Литература

**Диков, А.В.** Клиентские технологии веб программирования: JavaScript и DOM: учебное пособие / А.В. Диков. – СПб: Лань, 2020 – 124 с.

**Читанамбри, Кирупа.** Изучаем React / Кирупа Читанамбри – 2-е изд. – М.: Эксмо, 2019. 368 с.

**Хавербеке, Марейн.** Выразительный JavaScript. Современное веб-программирование. / Марейн Хавербеке – 3-е изд. – СПб.: Питер, 2019. – 480 с.:

Преподаватель Рогалевич А.В.

Рассмотрено на заседании цикловой

комиссии программного обеспечения

информационных технологий №10

Протокол № \_\_\_ от \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Председатель ЦК\_\_\_\_\_\_\_\_ В.Ю.Михалевич