**Вопрос 1** **Верно ли утверждение что 0, NaN и пустая строка “” при сравнении(==) с false будут вычислены в true?**

Да - согласно правил преобразования типов все вышеуказанные величины будут преобразованы в false

Частично - кроме NaN хотя правила преобразования типов гласят, что NaN преобразуется в false

Нет - компилятор не сможет корректно преобразовать все данные величины к одному типу

Частично - кроме 0 и “” компилятор не сможет корректно преобразовать 0 и «», так как они являются значениями типов

**Ответ:** Частично - кроме NaN хотя правила преобразования типов гласят, что NaN преобразуется в false

**Вопрос 2** **Какие из специальных числовых значений относятся к JavaScript?**

-Infinity

Infinity

NaN

**Ответ:** -Infinity   
Infinity   
NaN

**Вопрос 3** **Что увидим в консоли и почему?**

**console.log( null > 0 );**

**console.log( null == 0 );**

**console.log(null >= 0);**

⭘ false

true

true

Компилятор преобразует null в 0 и будет сравнивать два числа

⭘ false

false

false

null == undefined и поэтому ни одна из записей не вернет true

⭘ Данная запись не пройдет компиляцию

⭘ false

false

true

Потому что алгоритмы сравнения и проверки неравенства работают по-разному

**Ответ:** ⭘ false  
false  
true  
Потому что алгоритмы сравнения и проверки неравенства работают по-разному

**Вопрос 4** **Что такое метка и с какими директивами она применяется?**

Способ прервать, а за тем продолжить несколько вложенных циклов одновременно применяется с директивами break/continue

Способ прервать или продолжить несколько вложенных циклов одновременно применяется с директивами break/continue

Способ прервать/продолжить работу определенного цикла и является альтернативой директивам break/continue

Способ прервать или продолжить несколько вложенных циклов одновременно применение с директивами не предусмотрено, отрабатывает при обращении к метке по имени

**Ответ:** Способ прервать или продолжить несколько вложенных циклов одновременно применяется с директивами break/continue

**Вопрос 5** **При каком значении i цикл будет остановлен?**

**var i = 3;**

**while (i) {**

**console.log( i );**

**i--;**

**}**

i = минимальное значение

При i = 0

Цикл будет длиться вечно

**Ответ:** При i = 0

**Вопрос 6** **Какой код отработает без ошибок?**

🞎 alert('Привет'); alert('Мир');

🞎 alert('Привет')

alert('Мир')

🞎 alert("Привет Мир")

[1, 2].forEach(alert)

🞎 alert(3 +

1

+ 2);

**Ответ:** 🞎 alert('Привет'); alert('Мир');   
🞎 alert('Привет')  
alert('Мир')   
🞎 alert(3 +  
1  
+ 2);

**Вопрос 7** **Какое из неравенств будет вычисляться в true?**

console.log("Арбуз" < "Яблоко")

console.log(NaN == NaN)

console.log("Хочется" != "Колется")

console.log(3 < 2)

**Ответ:** console.log("Арбуз" < "Яблоко")   
console.log("Хочется" != "Колется")

**Вопрос 8** **Какой из примеров представленных ниже отработает правильно и выведет 5?**

**var a = 5;**

**alert( a );**

**alert( a );**

**var a = 5;**

⭘ В первом примере ошибки не будет, так как соблюдается порядок инициализации, во втором же случае будет ошибка, так как инициализация будет происходить во время присвоения после вывода переменной.

⭘ В первом и во втором будет ошибка, так как обращаемся к переменной раньше, чем присваиваем ей значения.

⭘ Во втором случае ошибки не будет, так как переменная находится в области видимости global object и будет принадлежать ему же.

В первом же случае будет выдана ошибка, так как переменная инициализируется типом var и соответственно не будет принадлежать global object.

⭘ Ошибки не будет, так как переменные будут проинициализированы компилятором до обращения к ним.

**Ответ:** ⭘ В первом примере ошибки не будет, так как соблюдается порядок инициализации, во втором же случае будет ошибка, так как инициализация будет происходить во время присвоения после вывода переменной.

**Вопрос 9** **Что будет выведено в консоль при обработке данного кода console.log(true ? 1 : 2); и почему ?**

Ничего - так как величина слева вычисляется в true

1 и 2 - так как величина слева вычисляется в true

1 - так как величина слева вычисляется в true

2 - так как величина слева вычисляется в true

**Ответ:** 1 - так как величина слева вычисляется в true

**Вопрос 10** **Есть ли разница между null и undefined? (выбрать все правильные варианты)**

🞎 undefined – значение, которое можно присвоить явно

null – значение, которое присваивается автоматически в переменную, которая его не имеет

🞎 Есть

null - пустое значение (явно присвоенное)

undefined - не присвоенное значение (по умолчанию) имеет тип "undefined"

🞎 Нет

🞎 Есть

null - это объект

undefined - имеет тип "undefined"

**Ответ:** 🞎 Есть  
null - пустое значение (явно присвоенное)  
undefined - не присвоенное значение (по умолчанию) имеет тип "undefined"   
🞎 Есть  
null - это объект  
undefined - имеет тип "undefined"

**Вопрос 11** **Как вычисляются операторы с равным приоритетом по умолчанию (например 1-2+1)?**

От большего значения к меньшему

Справа налево

Слева направо

Никак, так как отсутствует оператор присваивания

**Ответ:** Слева направо

**Вопрос 12** **Какие из нижеперечисленных вариантов являются операторами? (выбрать все правильные варианты)**

typeof

+

alert

>

**Ответ:** typeof   
+   
>

**Вопрос 13** **Что будет выведено в результате выполнения кода:**

**switch(5){**

**default:**

**console.log(0);**

**case 1:**

**console.log(1);break;**

**case 4+1:**

**console.log(4+1);**

**case 2:**

**console.log(2);**

**}**

0,1

5,2

Код не пройдет компиляцию

4,2,0

**Ответ:** 5,2

**Вопрос 14** **Что будет выведено в консоль и почему?**

**var apples = "2";**

**var oranges = "3";**

**console.log( +apples + +oranges )**

23 - бинарный оператор + производит конкатенацию и при этом преобразует переменные в числовые величины

Код не пройдет компиляцию из-за некорректной записи в console.log()

23 - так как бинарный оператор + выполняет конкатенацию строк

5 - так как унарный оператор + преобразует строки в числа

**Ответ:** 5 - так как унарный оператор + преобразует строки в числа

**Вопрос 15** **Что появится в консоли?**

**1**

**var i = 1;**

**console.log( 2 \* ++i );**

**2**

**var i = 1;**

**console.log( 2 \* i++ );**

⭘ 1 - 2

2 - 4

⭘ 1 - 4

2 - 4

⭘ 1 - 4

2 - 2

⭘ 1 - 2

2 - 2

**Ответ:** ⭘ 1 – 4  
2 - 2

**Вопрос 16** **Что из перечисленного являются модальным окном alert, prompt и confirm? (выбрать все правильные варианты)**

alert

Ни один из вышеперечисленных

prompt

confirm

**Ответ:** alert   
prompt   
confirm

**Вопрос 17** **Расположите операторы по возрастанию признак приоритет (где первый с самым низким приоритетом и так далее по возрастанию):**

\*

+

==

&&

||

**Ответ:** \*   
+   
==   
&&   
||

**Вопрос 18** **Какие синтаксические конструкции могут применяться с оператором - ?**

Любые - включая директивы continue и break

Все кроме continue и break

Только те, которые возвращают значения

**Ответ:** Только те, которые возвращают значения

**Вопрос 19** **Какие из нижеперечисленных вариантов могут быть именем переменной? (выбрать все правильные варианты)**

var name = 5;

var $ = 5;

var 12name\_ = 5;

var \_ = 5;

var $12 = 5

**Ответ:** var name = 5;   
var $ = 5;   
var \_ = 5;   
var $12 = 5

**Вопрос 20** **Что появится в консоли и почему?**

**a = (5,6)**

**console.log(а);**

Такая запись не пройдет компиляцию.

5 - потому что запятая является оператором, приоритет которого ниже оператора присваивания, поэтому все, первой величины компилятор отбросит.

6 - потому что запятая является оператором, приоритет которого ниже оператора присваивания, поэтому компилятор отбросит все величины, кроме последней.

7 - ввиду того, что используется устаревшая форма записи.

**Ответ:** 6 - потому что запятая является оператором, приоритет которого ниже оператора присваивания, поэтому компилятор отбросит все величины, кроме последней.

**Вопрос 21** **Что такое константа и можно ли ее изменять?**

Константа – это переменная, которая никогда не меняется

Константа – это переменная, которая изменяется с определенной периодичностью

Константа – специальная величина, заданная по определенной логике и постоянно изменяется

**Ответ:** Константа – это переменная, которая никогда не меняется

**Вопрос 22** **Для каких задач могут использоваться циклы в js? (выбрать все правильные варианты)**

Записи потока данных в файл

Выполнение одной и той же операции несколько раз

Для сокрытия выполнения повторяющихся операций

Для вывода строки на экран

Выполнение повторяющихся операций

**Ответ:** Записи потока данных в файл   
Выполнение одной и той же операции несколько раз   
Для вывода строки на экран   
Выполнение повторяющихся операций

**Вопрос 23** **Отработает ли код указанный ниже?**

**1 console.log(8 \* null)**

**2 console.log("5" - 1)**

**3 console.log("5" + 1)**

**4 console.log("пять" \* 2)**

**5 console.log(false == 0)**

Нет - величина в 1-м примере не сможет быть вычислена

Нет - величина в 3-м примере не сможет быть вычислена

Нет - величина в 2-м примере не сможет быть вычислена

Нет - величина в 4-м примере не сможет быть вычислена

Код отработает без сбоев

Нет - величина в 5-м примере не сможет быть вычислена

**Ответ:** Код отработает без сбоев

**Вопрос 24** **Что будет выведено на экран и почему?**

**1 console.log(null || "user")**

**2 console.log("Karl" || "Buddy")**

Ничего - ошибка компиляции.

1- ничего, 2 - "Karl" потому что при использовании оператора логического или, если правая сторона вычисляется в true, то в консоль будет выведена величина слева.

1- ничего, 2 - "Buddy" потому что при использовании оператора логического или, если левая сторона вычисляется в true, то в консоль будет выведена величина справа.

1- "user", 2 - "Karl" потому что при использовании оператора логического или, если левая сторона вычисляется в true, то в консоль будет выведена величина слева если величина слева false то выводится величина справа.

**Ответ:** 1- "user", 2 - "Karl" потому что при использовании оператора логического или, если левая сторона вычисляется в true, то в консоль будет выведена величина слева если величина слева false то выводится величина справа.

**Вопрос 25** **Что получится в результате выполнения этой операции – 0/0 ?**

Infinity

-Infinity

NaN

Будет ошибка

**Ответ:** NaN

**Вопрос 26** **Верно ли утверждение, что в цикле for можно объявить переменную, которую можно использовать за его пределами?**

Нет - такой код не компилируется

Да - величина будет видна за пределами тела цикла

Нет - так как область видимости этой переменной тело цикла и она будет интерпретирована в undefined

**Ответ:** Да - величина будет видна за пределами тела цикла

**Вопрос 27** **Какое из утверждений будет верным при подборе имени для переменной? (выбрать все правильные варианты)**

Использовать короткие имена только для переменных «местного значения»

Имя переменной должно максимально четко соответствовать хранимым в ней данным

Только английский

Каждое слово, из которого состоит переменная, пишется с большой буквы

**Ответ:** Использовать короткие имена только для переменных «местного значения»   
Имя переменной должно максимально четко соответствовать хранимым в ней данным   
Только английский   
Каждое слово, из которого состоит переменная, пишется с большой буквы